TECHNISCHE UNIVERSITÄT MÜNCHEN

Fakultät für Informatik
Lehrstuhl für angewandte Informatik / kooperative Systeme

Proactive Recommender Systems in Automotive Scenarios

Roland Bader

Vollständiger Abdruck der von der Fakultät für Informatik der Technischen Universität München zur Erlangung des akademischen Grades eines

Doktors der Naturwissenschaften (Dr. rer. nat.)

genehmigten Dissertation.

Vorsitzender:		Univ.-Prof. Dr. Uwe Baumgarten
Prüfer der Dissertation:		
	1.	Univ.-Prof. Dr. Johann Schlichter
	2.	Univ.-Prof. Dr. Claudia Linnhoff-Popien (Ludwig-Maximilians-Universität München)

Die Dissertation wurde am 09.01.2013 bei der Technischen Universität München eingereicht und durch die Fakultät für Informatik am 03.05.2013 angenommen.

Acknowledgements

There are so many who have made this work possible with long discussions, constructive feedback, invaluable ideas or just by being there.

First and foremost I would like to thank my advisor Prof. Dr. Schlichter. He gave me the chance to pursue my research at his institute as an external PhD candidate. We had a lot of discussions to push the work into the right direction and he gave me the freedom to follow my own ideas. Especially his reviews of this thesis were in great detail. Likewise, I want to thank Prof. Dr. Linnhoff-Popien for her valuable review.

The results in this thesis were mainly obtained at BMW Research and Technology in Munich. A special thanks goes to Robert Hein who gave me the chance and freedom to follow my research interests. He introduced me to the world and challenges of the automotive industry as the basement for this thesis. I had a wonderful time in his team. Many thanks goes also to my former teammates Mikael Vaaraniemi, Isaac Trefz, Stefan Feit, Axel Jansen, Carsten Isert, Dominik Gusenbauer, Andreas Winckler and Klaas Klasing. Furthermore, I had the chance to work with other great teams at BMW. Thank you for your valuable input. I also want to thank the students who contributed to this thesis with their own great work: Hermann Stünz, Timo Schmidt, Axel Jansen, Eugen Neufeld, Sebastian Hoffmann, Jonathan Dees, Thomas Trummer, Andreas Karitnig and Konrad Vogt.

Being an external PhD candidate at a university has the advantage to be part of two great teams. Prof. Dr. Schlichters group of applied informatics and cooperative systems made it easy for me to integrate myself. Its members gave me some of their valuable time for constructive discussions which helped to find my direction. Having the chance to contribute to the regular feedback seminars greatly helped to focus my research. A special thanks goes to Dr. Wolfgang Wörndl. He introduced me to the wonderful world of recommender systems and gave me constructive feedback to my research activities. Many thanks goes also to: Halgurt Bapierre, Dr. Michele Brocco, Friedl Bunke, Dr. habil. Georg Groh, Dr. Robert Eigner, Dr. Florian Forster, Rene Frieß, Claudius Hauptmann, Dr. Jan Herrmann, Walter Kammergruber, Niklas Klügel, Hubert Kreuzpointner, Alexander Lehmann, Marlene Müller, Dr. Vivian Prinz and Marouane Sayih.

The process of obtaining a PhD involves many ups and downs. There are people along the way who are not involved in this process directly but are affected. First, I want to thank my girlfriend Vanessa Kunze to be one of these people. She helped me to get through the downs, enjoyed with me the ups and did not complain about far too less time together. My parents Josef and Margarete Bader also deserve a great deal of thanks. They showed me the right path in my childhood, gave me the freedom I needed and always supported my decisions. Finally, I want to thank all of my friends. They showed me that there is a life apart from my research and let me tease them with my user studies.

Abstract

Modern driver information systems are able to provide a large amount of data to the driver. If all these data is delivered unfiltered to the driver, information overload becomes a serious problem. Recommender systems are a promising approach to reduce information overload. However, they are mainly designed for desktop systems or mobile devices. Therefore, they do not take the restrictions of the automotive environment into consideration. We believe that proactively delivered recommendations are the most promising solution to handle these restrictions. Such systems process and deliver information automatically without explicit user request. The goal of this thesis is to investigate the design of proactive recommender systems in an automotive environment.

Proactive behavior requires an intelligent information processing. In case of proactive recommender systems, the system provides recommendations in the right situation and adapts user preferences to situational circumstances. We propose a framework for situation awareness which is tailored to proactive in-vehicle recommendations. The framework is based on human situation awareness which is represented by well-known machine learning methods. A study for gas station recommendations with artificial user data shows the applicability of our framework.

Intelligent information processing also involves selecting the right items for the users. Besides user preferences, the recommender system should also consider the context of the users in an automotive environment. Context integration into recommender systems became a popular research area in recent years. We apply a context integration method that exploits well-known paradigms from multi-criteria decision making (MCDM). The advantage of using MCDM methods is to be able to handle multiple heterogeneous context and user information. In combination with the context integration paradigms of prefiltering and postfiltering, the MCDM methods can be applied efficiently in a dynamic automotive environment. For evaluation, we compare the performance of different MCDM approaches in a user study based on gas station recommendations. Comparative models show better results than linear models or dominance-based filters.

As proactive recommendations are delivered without explicit user request, comprehensibility is an important issue. Adding explanations is a promising method to improve the comprehensibility of a recommendation. We investigate two explanation methods for proactive recommendations. The first method justifies the selected items. It uses the resulting items of our context-aware recommender system and generates arguments for explanations. Our user study with gas station recommendations shows that the users are mostly satisfied with the generated explanations. The second method justifies the reason of a recommendation. We propose a method to explain the reasoning in the Bayesian network that we used in our situation awareness framework and show example explanations.

The individual investigation of important components of a proactive recommender system is not enough to tell whether the integrated system is designed properly. Therefore, we want to know whether the drivers would accept a proactive recommender system consisting of the presented components. To investigate user acceptance, we implement an in-vehicle prototype of a gas station recommender system based on our previous results. To make the prototype easy to use, it is equipped with an adequate user interface that is validated with expert interviews. We investigate user acceptance with the technology acceptance model (TAM). The results show the importance of choice and a clear presentation of information. They also indicate that drivers tend to accept such a proactive recommender system inside their cars. Therefore, we conclude that our system design is adequate to provide proactive recommendations in automotive environments.

Kurzfassung

Heutige Fahrerinformationssysteme bieten dem Fahrer eine Vielzahl von Informationen. Wenn diese Informationen ungefiltert an den Fahrer weitergeleitet werden würden, dann kann es zu Informationsüberflutung kommen. Zur Vermeidung von solcher Informationsüberflutung haben sich Empfehlungssysteme (Recommender Systeme) etabliert. Empfehlungssysteme sind in erster Linie für den Desktopbereich konzipiert worden und berücksichtigen deshalb keine Einschränkungen der Interaktion und der menschlichen Informationsverarbeitung im Fahrzeug. Es existieren bereits Ansätze um diese Einschränkungen aufzuheben. Der Einsatz von proaktiven Empfehlungssystemen ist dabei einer der aussichtsreichsten Möglichkeiten. Diese Systeme arbeiten selbstständig und präsentieren Empfehlungen ohne Benutzeranfrage. Der Schwerpunkt dieser Arbeit liegt darin den Aufbau von proaktiven Empfehlungssystemen im Fahrzeug zu konzipieren.

Proaktives Verhalten erfordert eine intelligente Verarbeitung von Informationen. In Kombination mit Empfehlungssystemen bedeutet dies, dass das System Informationen in einer passenden Situation präsentiert und Benutzerpräferenzen an die jeweilige Situation anpasst. In dieser Arbeit wird ein Framework zur Modellierung von Situationsbewusstsein für proaktive Empfehlungssysteme beschrieben. Das Framework basiert auf menschlichem Situationsbewusstsein, da dieses Gebiet in der Forschung bereits weitreichend untersucht wurde. Menschliches Situationsbewusstsein wird dabei mit Hilfe von Methoden aus dem Bereich des maschinellen Lernens abgebildet. Zur Evaluation werden Daten für ein proaktives Tankstellenempfehlungssystem erhoben und die Anwendbarkeit des Frameworks gezeigt.

Neben der Wahl einer passenden Situation, spielt auch die Wahl von geeigneten Empfehlungen eine entscheidende Rolle. Dies beruht bei Fahrern nicht nur auf ihren Präferenzen, sondern auch auf ihrem Kontext. Um dem heterogenen Charakter verschiedener Kriterien einer kontextadaptiven Empfehlung gerecht zu werden, werden in dieser Arbeit Methoden aus der multikriteriellen Entscheidungstheorie eingesetzt (Multi-Criteria Decision Making (MCDM)). Diese werden mittels Vor- und Nachfilterung in den Empfehlungsprozess integriert. Die Evaluation vergleicht verschiedene MCDM Methoden auf Basis von Tankstellenempfehlungen. Vergleichende Methoden schneiden dabei besser ab als lineare Modele oder Methoden, die auf Dominanz basieren.

Da proaktive Empfehlungssysteme ohne Anfrage empfehlen, sollte ihr Verhalten für den Benutzer verständlich sein. Explizite Erklärungen sind eine gängige Methode um die Verständlichkeit eines Systems zu erhöhen. In dieser Arbeit werden zwei Methoden vorgestellt um proaktive Empfehlungen zu erklären. Die erste Methode rechtfertigt die Inhalte einer Empfehlung. Dabei werden passende Argumente auf Basis des kontextadaptiven Empfehlungsansatzes generiert. Bei einer Benutzerstudie mit Tankstellenempfehlungen waren die Benutzer in den meisten Fällen zufrieden mit den generierten Erklärungen. Die zweite Methode rechtfertigt mit Hilfe der aktuellen Situation das eigenständige Verhalten des Systems. Die Methode basiert auf der Modellierung des Situationsbewusstseins und wird mit Beispielen untersucht.

Die individuelle Betrachtung von Bestandteilen eines proaktiven Empfehlungssystems ist nicht ausreichend um die Anwendbarkeit des integrierten Systemdesigns abzuleiten. Deshalb wird in einer finalen Benutzerstudie in einem Fahrzeug die Benutzerakzeptanz eines Systems untersucht, das aus den vorgestellten Bestandteilen besteht. Hierfür wird eine angemessene Benutzerschnittstelle für das Fahrzeug entwickelt und mit Experteninterviews validiert. Als Grundlage der Studie dient das Technologieakzeptanzmodel (Technology Acceptance Model (TAM)). Die Ergebnisse zeigen, dass Auswahl und die klare Vermittlung von Informationen eine wichtige Rolle spielen. Ferner tendieren die Testfahrer dazu ein solches System zu akzeptieren. Somit kann daraus geschlossen werden, dass das vorgestellte Systemdesign für den Einsatz von proaktiven Empfehlungssystemen im Fahrzeug hinreichend ist.

Contents

List of Figures

List of Tables

1. Introduction

Modern cars have become a powerful digital device over the last couple of years. They are able to capture environmental information with sensors and access digitally available information from the Internet over broadband connections. Nevertheless, the primary task of a driver is to drive the car and maneuver through traffic situations (Tonnis et al. [TBK06]). Other tasks are even secondary including responses to the environment (e.g., a honk or a turn signal) or tertiary. Tertiary tasks are decoupled from driving and aim to provide more convenience (e.g., turning on the air conditioner).

Interacting with **in-vehicle information systems** (**IVIS**) represents a tertiary task. An example for IVIS is searching for a **point-of-interest** (**POI**). POIs comprise services such as restaurants or gas stations. They are described by their location and additional information like opening times or available facilities such as ATMs or shops. In general, POIs are stored in a database inside the car. With broadband Internet connection, the database can be updated and additional information for POIs can be downloaded. This increases the amount of potentially interesting information for the driver. The usual way of searching a POI is by querying the local database and browsing through the results. Finding the right POI and formulating queries may impose cognitive load on the driver.

IVIS for POIs usually display information on a small **central information display** (**CID**). The interaction with visual IVIS may cause driver distraction because the glance of the driver is off the road. Furthermore, displaying large amounts of POIs on small screens is a challenge inside a car. Besides, the number of IVIS that might have relevant information for the driver increased over the last couple of years. This is due to the increase of computational power on small chips for lower prices, new high-resolution displays and affordable cellular broadband connection. As a consequence, solutions have to be found to avoid information overload like depicted in Figure 1.1.

Intelligent selection of POIs by means of recommender systems (Resnick and Varian [RV97]) is a possible solution. Recommender systems comprise a group of methods to support an active user with suggestions to find relevant items. The idea is to assist like-minded users with "Word of Mouth" knowledge. Recommendation methods use item details, user preferences, demographic information and most recently the context of the user to select suitable items. Recommender systems became popular by commercial services such as Amazon.de (e-commerce), Last.fm (music) or Netflix (movie rental). However, available recommender systems are designed for desktop computers or mobile devices. Due to interaction limitations, they should be designed differently for cars.

Figure 1.1.: Information overload inside a car due to many in-vehicle information systems (IVIS) (Tretten [Tre11])

1.1. Mobile Information Need

Classical in-car information need comprises status information of driver assistance systems (e.g., the speed indicator) and warnings (e.g., low current gas level warning). Infotainment systems extend the scope of relevant information. They belong to the group of IVIS that provide entertaining information in addition to warnings and status information. Infotainment systems are designed to fulfill the information need of a driver.

There are lots of sources of relevant information to cover mobile information need. Today, so-called apps for smartphones provide an easy way to access several information sources. According to App of the day [App11], Apple's app store offered 300.000 apps by end of 2010. Among others, the categories navigation (e.g., DB Navigator, Navigon), travel (e.g., Tripadvisor, PrimoSpot Parking, Gasbuddy, Expedia) and lifestyle (e.g., Filmstarts, Yelp, Qype, Post app) make up approximately 15%. Content provider used in these apps offer large amounts of POIs. Next to static information such as location, address or business hours, additional dynamic information of interest such as user ratings, movie critics or gas prices are offered. However, apps are designed for smartphones. They do not consider interaction restrictions inside a car and are not connected to in-car sensory. These sensors provide valuable context information about the driver, the car and the environment.

Several user studies investigated what kind of information people want to access while they are on the go. A diary study of mobile information need by Sohn et al. [SLGH08] reveals what is of interest for mobile users. One quarter of mobile information need is related to either discovering a POI or navigate to one. Trivia information like the birthday of Bob Marley also appears in 18% of the diary entries. Furthermore, business hours and news are also of interest. 72% of information is context related, e.g., location (35%), activity (24%) or time (28%). Vrcek et al. [VBB09] confirm that mobile users seem to be most interested in navigational systems. Such systems include routing and traffic services and information systems for parking opportunities or tourist information. Alt et al. [AKS+10] confirm with an online survey that news, weather and nearby sights are relevant for the drivers. An interesting observation of Sohn et al. is the reason why information need is not addressed. Only 45% of needed information is accessed at the desired time and 25% later. 28% of the users who access information later mention the involvement in driving or biking as reason.

1.2. Problem Description

The general problem of accessing information inside a car is driver distraction. Information sources like POIs comprise much information of relevance for the user, e.g., their position on a digital map. Therefore, POIs are mainly displayed in the CID of the car like a navigation system. With much information, the risk of information overload is high. This may lead to long phases of glances away from the street. The drivers are also subject to cognitive load because they have to handle this information. Furthermore, the drivers interact with the CID to find relevant information. These aspects may lead to driver distraction. Driver distraction should be avoided before it becomes a serious risk.

There is need for information inside a car but driving often leads to postpone or skip the consumption of this information. Several approaches are possible to solve the problem of in-car information access. Some researchers distinguish traffic situations according to the load they impose on the driver. Stationary situations, e.g., standing still at a traffic light, impose low cognitive effort. For instance, Alt et al. [AKS+10] detect standstill times at a traffic light and push information in that time slot. Their approach depends on infrastructure. Furthermore, important information might be missed, if too few standstill times occur. Another approach to solve the problem is to improve the access to large amounts of information. Either the visualization on small displays or the interaction with large amount of information can be improved. However, this could be challenging for large sets of POIs. Furthermore, most of the information is not relevant for the drivers in their situation. The third solution is to reduce the amount of information. Intelligent agents (e.g., Maes [Mae94]) are an abstract class of methods to work out tasks such as information filtering. They decide which information is filtered based on knowledge and models of the user, the domain and the environment. The main challenge for intelligent selection is to infer on the right information at the right time.

The research area of artificial intelligence offers several methods for intelligent filtering. This work focuses on recommender systems as a solution of intelligent filtering to avoid information overload. However, classical recommender systems are designed for the users who are able to focus their full attention to the system. In an automotive environment, we have to approach recommender systems from a different viewpoint. In contrast to desktop systems and mobile devices, information access is restricted because of the primary task of driving. Interaction with visual IVIS to enter search queries or browse catalogs should be reduced to avoid driver distraction. Another aspect of classical recommender systems is that they focus on the accuracy of delivered items considering user preferences in general. However, mobile information need is mainly defined by context and the situations of a user. This information should be considered when providing suggestions.

1.3. Goals

The goal of this thesis is to investigate proactive recommender systems as a solution of integrating recommender systems into the automotive environment. Proactive behavior involves the freedom of a system to react on an input on its own (Covey [Cov89]). It allows the system to support the driver intelligently with the right information at the right time. Our overall research question is:

> How should proactive recommender systems be designed inside a car to assist the drivers receiving relevant items for their current situation at the right time?

We apply the research methodology of design science in this work by following the guidelines of Hevner et al. [HMPR04] to design science in information systems. Our proposed approach contains four main artifacts in system design: inferring on situations to make a recommendation, selecting appropriate items, explaining the decisions of the system and delivering the results to the driver. We show the relevance of the problem of information overload while driving by means of literature review in automotive research. The artifacts are evaluated observationally with offline and online user studies. Additionally, experiments are carried out with common metrics from the field of intelligent systems and recommender systems to make the results comparable. A review of related work in proactive recommendation systems shows why other similar approaches could not be applied to our problem. A comprehensive description of relevant methods from intelligent systems, recommender systems and decision theory builds the groundwork for the design of our artifacts. The results are communicated to the research community by several publications on conferences and workshops.

Our proposed approaches for the artifacts are investigated with the simple example of a proactive gas station recommender. This is a comprehensible and obvious use case that concerns all drivers. Although some of the artifacts are more complex than it might be needed for a gas station recommender, literature shows that they are important for a proactive recommender system. The goal of the selected example is to increase the comprehensibility of the work. We do not aim to propose an adequate solution for a gas station recommender.

To be able to answer the research question, we divide the question in four sub-questions. The first question we answer is: How to infer from the situation of the user on the usefulness of recommendations? Human situation awareness involves sensing of information that defines a situation, the comprehension of the current situation as well as the anticipation of future situations (Endsley [End00]). Our goal is to transfer human models of situation awareness into computational models. We assume that there are sensors that deliver information about the situation of the user. The focus of our research is to establish a situation awareness framework for decision making towards recommendations using data from information sources such as sensors or databases. The framework should be able to incorporate existing inference methods such as Bayesian networks or fuzzy logic to represent dependency of situations and information need.

The second question is: How to select useful items in the situation of the user? Classical recommender research focuses on recommendation accuracy instead of usefulness in situations. The major difference is that usefulness is especially determined by the context of the user. Context-aware recommender systems became popular (e.g., Adomavicius and Tuzhilin [AT08]) recently. Our goal is to propose a flexible model of context integration that is able to handle different types of heterogeneous context and user information. It should not be specialized to classical methods such as collaborative filtering or content-based filtering but should be able to integrate such methods. This problem can be modeled as a **multi-criteria decision making** (**MCDM**) problem. Decision theory provides well-defined methods for MCDM that can be used for that. Our goal is to exploit these MCDM methods in combination with common context integration methods for recommender systems such as prefiltering or postfiltering.

The third question is: How to make intelligent behavior in proactive recommender systems comprehensible to the user? In contrast to classical recommendation systems, a proactive system chooses its query for items itself. As the users do not request any information, they might not understand why the system provides recommendations. In research, comprehensibility is either out of scope of proposed proactive recommenders, implicitly solved by design or domain-specific, e.g., persuading a shopper to buy. Explanations are a promising method to make decisions of intelligent systems more comprehensible. Applied in a proactive recommender, explanations are able to justify the decisions of the system in order to convince the driver that the recommended items are useful in that situation. Our goal is to design and evaluate methods that are able to generate suitable explanations for proactive recommendations.

The forth question is: Do the drivers accept proactive delivery of items inside a car? The answers to the other three questions only comprise results for specific parts of a proactive recommender system. However, an automotive environment is different to usual application areas of recommender systems such as desktop computers and mobile devices. Our goal is to investigate whether the drivers would accept such a system in their cars. User acceptance comprises **ease of use** (**EOU**) and **usefulness** (**U**) (Davis [Dav89]). EOU requires an adequate user interface that follows principles of usability.

1.4. Road Map

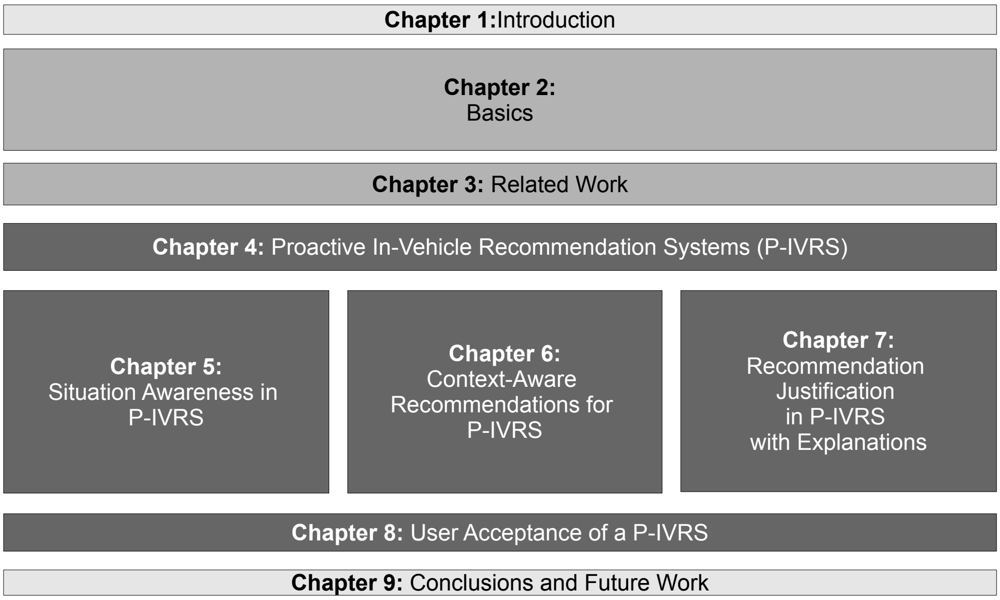


Figure 1.2.: Structure of the thesis

Figure 1.2 shows the structure of the thesis. We distinguish general chapters (1, 9), literature review (2, 3) and our own contribution (4, 5, 6, 7, 8).

Chapter 1 gives an introduction to the topic of proactive recommender systems in automotive scenarios. It shows that there is need for information inside a car, what the problem of information access is and what goals we pursue to solve the problem.

Chapter 2 establishes basics to understand our approach and models for proactive recommender systems. First, the conditions of the automotive environment are derived from literature. Second, we give a definition of our understanding of the terms context and situation. Third, we describe methods of decision making and intelligent systems, we used for our models. Fourth, we introduce recommender systems in general and especially context-aware recommender systems and explanations in recommender systems. Finally, we describe properties of proactive assistance in general.

Based on this information, Chapter 3 details related approaches of proactive information systems. The chapter is divided in approaches for the desktop, for mobile devices and for cars. Furthermore, each section distinguishes approaches which either trigger information delivery by the task of the user, the environment or user preferences.

Chapter 4 classifies our approach of **proactive in-vehicle recommender systems** (**P-IVRS**). It defines the scope of P-IVRS compared to similar recommender systems, potential use cases, requirements and the design of a P-IVRS. As a central component, we detail how the system makes proactive decisions on its own.

The next three chapters describe our generic approach for each system component and its evaluation. Chapter 5 details our framework for situation awareness in proactive recommender systems. For evaluation, we describe the collection of training data and three implementations of the core of the framework.

Chapter 6 comprises our approaches for context-aware recommendations. First, we show the generic parts of our proposed model including prefiltering, postfiltering and a multi-criteria decision making (MCDM) recommender to assess the items. Next, we focus on detour as context because it is one of the most important context information in our use case of gas station recommendations. Finally, a user study is described which compares different kinds of MCDM methods for item assessment and postfiltering and our location-based filter for prefiltering.

The third system component to generate explanations is described in Chapter 7. First, findings from a preliminary study about explanations for gas station recommendations are detailed. Second, we describe a general approach for an explanation interface for P-IVRS. Third, our method for generating item explanations and its evaluation with an offline study is described. Finally, our method for situation explanations is detailed.

The last Chapter 8 of our contribution comprises a user acceptance study. We design an adequate user interface for proactive in-vehicle recommendations. It is used together with the implementation the situation awareness model, the context-aware recommender and the explanation component, to carry out the user acceptance study inside a car.

We close the thesis in Chapter 9 with a summary of our major findings and conclusions. Based on the findings in this thesis and our experience with proactive in-vehicle recommender systems, we give an outlook to further research.

2. Basics

In this chapter, we investigate characteristics of the automotive environment, recommender systems and proactive systems and describe basic notations for our problem. The limitations of the automotive environment are important because our proactive recommender is embedded into the car as an information system. Additionally, incorporating the context of the users and their situations is also necessary for drivers. We describe our understanding of context and situations. As recommender systems lead to a final choice, we introduce human decision making and mathematical methods for multi-criteria decision making (MCDM). To make the system able to act intelligently, we also introduce fuzzy logic and Bayesian networks and describe how Bayesian networks can be made comprehensible with explanations. We use these methods later to build our proactive recommender system. Along with classical recommender systems, we introduce **context-aware recommender systems** (**CARS**) and explanations in recommender systems. Finally, we describe general characteristics of proactive assistance and proactive information systems.

2.1. Automotive Environment

Cars have developed in recent decades more and more into electronic devices assisting the driver in various ways. Especially the numbers of **advanced driver assistance systems** (**ADAS**) have increased significantly over the years. ADAS comprise additional systems in a car to support the driver. Some ADAS are supplemented by in-vehicle information systems (IVIS). IVIS either deliver relevant information for the driving task, e.g., a warning, or assist the user with other kinds of information, e.g., music controls (infotainment). Telematics systems such as a navigation system are the most popular IVIS. In the context of designing information delivery as well as the interaction for IVIS, designers have to be aware of driver distraction issues.

2.1.1. Advanced Driver Assistance Systems (ADAS)

Driving Task Classification

The classification of ADAS depends on the **task of the driver** which ADAS want to assist and how the driver handles these tasks. The three level human performance model

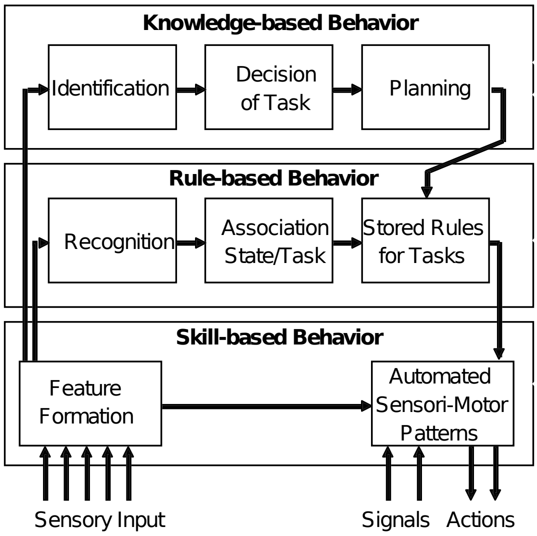


Figure 2.1.: Three levels of performance of skilled users (Rasmussen [Ras83])

of Rasmussen [Ras83] (Figure 2.1) describes **goal-oriented behavior** based on **cognitive load** of a task. It is a psychological model that is widely used for ADAS because driving itself is a goal-oriented task. The highest level of the model is **knowledge-based behavior** for unexpected or untrained tasks inside a car. The driver identifies input information and creates a plan by weighting alternative actions based on current goals and prior knowledge. It is the intention of the driver to follow this plan. Decisions are stored in memory to shorten future reaction on similar input information. Knowledge-based behavior is demanding for the driver, therefore Förster [F8̈7] recommends minimizing **intentional behavior** while driving. On the second level, patterns in memory are recognized based on input information and behavior rules for these patterns are retrieved (**rule-based behavior**). Recognition and retrieval of rules may become a routine during a learning phase in which the process is often repeated. It leads to a subconscious course of actions which is called **skill-based behavior**. The more skill-based an action is executed, the more cognitive resources for other tasks are available. It strongly depends on the **experience** of a driver or how often a traffic situation was experienced respectively. A novice driver starts using the controls of a car such as the steering wheel or the pedals on a knowledge-based level. On this level, the drivers have to think about every action before they perform it. After some time, the user or driver becomes more familiar with the controls of the car (skilled-based).

Donges [Don82] describes a three-layer **driving task oriented model** to complement the human performance oriented model of Rasmussen. All layers of the model contain parts which can be assigned to levels of Rasmussen's model. The highest layer is the **planning** or **strategic layer**. For instance, the drivers choose a route among alternatives depending if the area is unknown (knowledge-based level), known (rule-based level) or if it is routine (skill-based level). The next layer is the **maneuvering** or **tactical layer**. It comprises the control of the car on the road and among other traffic participants. The lowest layer is the **operational** or **control layer** to stabilize the car. The layers are subject to human, car and environmental influences.

Donges discusses components of the driving task itself and makes no difference how much they contribute to the driving task. Bubb [Bub03] classifies **tasks inside the car** by their contribution to the fulfillment of the driving task. Tasks which are directly associated to the driving task are **primary tasks**. **Secondary tasks** comprise responses to the environment, e.g., to honk or to indicate a turn, and reactions on the current situation, e.g., turning on the windshield wipers or the lights. **Tertiary tasks** are decoupled from the actual driving task and make driving or being in the car more convenient, e.g., searching for points-of-interest (POIs) or turning on the air conditioner.

Resource-based Information Processing

The model of Rasmussen is derived from classical **human information processing**. Abendroth [Abe01] distinguishes between sequential and resource-based models for human information processing. A **sequential model** assumes that the next stage of the process can only be executed if the previous stage is finished. A **resource-based** model implies that people only have limited resources available to perform tasks. Abendroth and Bruder [AB09] mention visual, acoustic, haptic and vestibular recognition as important human resources for driving, whereas the **visual sense** is the most import one. If tasks claim different resources their **interference** is lower but not fully unrestricted. The resource-based approach is traced back to Wickens' [Wic84] model of how people process information to perform tasks. It contains the reception of information by human sensory, the retrieval and the storage in the human memory and the choice of an action based on a risk-benefit analysis. For driving, a zero or very low willingness to risk an accident in the choice of an action is assumed. How efficiently an action can be performed by the driver depends on the **attention** of the driver, i.e., how many resources are available and how many resources the action needs.

Prediction and Preview

One of the most important capabilities of people for driving is prediction and preview (for example, see Sheridan [She87]). **Preview** is the ability, mainly visually, to recognize situations that are relevant for the driving task. **Prediction** is the estimation of future situations that influence the driving task. The ability of prediction and preview is strongly influenced by the experience of the driver. The **anticipation time** for predicted situations is different depending on the level a task has in Donges' [Don82] model, i.e., the time the driver needs to choose an appropriate action before a critical point. It ranges from a long horizon of some hours on the strategic layer, e.g., for route selection, to some seconds to stabilize the car. For instance, guidance systems deliver instructions to choose a maneuver some minutes before the critical point.

ADAS Classification

There are many ways to **classify** ADAS. Naab's [Naa04] classification distinguishes the systems according to their contribution to security and comfort. Complex driving situations lead to high workload. This may decrease the performance of the driver. This is a **security** risk where ADAS can deliberately intervene (for example, collision avoidance or warning, emergency brake or curve speed control). ADAS can also deliver higher **comfort** by taking over parts of the driving task (for example, **adaptive cruise control** (**ACC**), parking assistant, lane keeping assistant or a navigation system). In the last couple of years a third group of assistance systems became more and more important which classifies systems according to their contribution to **energy efficiency** (for example, stop and go systems). ADAS can be further classified by distinguishing between active and passive systems (for example by Kosch [Kos05]). **Passive safety** systems react in case an accident occurs to reduce negative consequences of a crash, e.g., airbags. **Active systems** react in case of a dangerous situation (generally just before an accident) to prevent the accident at the last minute (e.g., **electronic stability program** (**ESP**)).

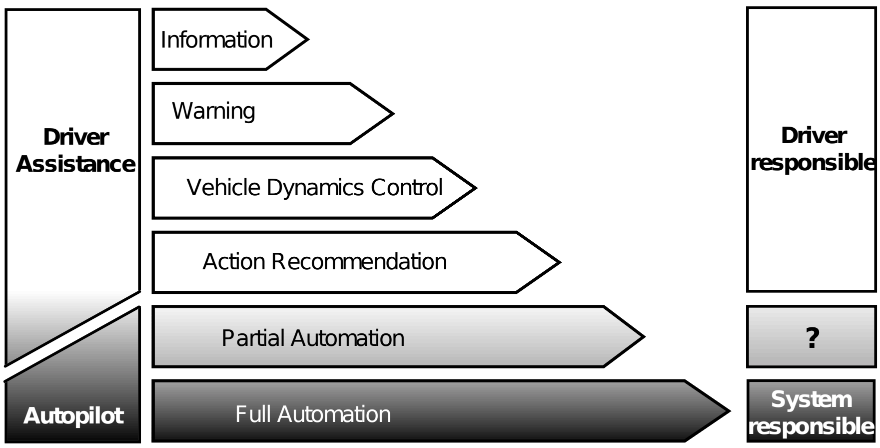


Figure 2.2.: Categories of ADAS implementations according to the level of automation (Donges [Don99])

Donges [Don99] makes a classification according to the **level of automation** (Figure 2.2). It is driven by the actual implementation of the assistance system. A system ranges from simply showing information, over making a warning or recommending actions to taking over parts of the driving task. A proactive recommender system is on the same level as a warning system. It delivers recommendations instead of warnings but item recommendations are no action recommendations. As a simplification, there are systems which inform the driver (in-vehicle information systems (IVIS)), others which act and some which do both.

It is important to note that for all assistance systems the driver has always to be in **full control**. In case tasks are performed by ADAS, the driver should always be able to take back the control. In general, the costs of **supervision** should not lead to more conscious behavior of the driver, i.e., a skill-based task should not become a rule-based or knowledge-based task.

Sensory systems

The more automation is involved, the more a system needs the capability to interpret environmental, human or car information. This calls for powerful **sensory systems** deployed in the car. Sensors either capture the state of the car with integrated sensors (for example, the current gas level or the temperature of the motor), the driver (for example, with cameras or driver interaction behavior) or the environment (for example, with cameras, lidar, radar or ultrasonic). Knowledge about the individual situation of a car can be augmented by sensor information from other cars. **Car2X** (**C2X**) communication (Car2Car or Car2Infrastructure) is a new technology to enable short-range exchange of information between cars (for example, Rockl et al. [RRFS07], Strassberger [Str07] or Kosch [Kos05]). Sensor data fusion aggregates pieces of sensor data to comprehend the situation and enable to choose an appropriate action.

2.1.2. In-Vehicle Information Systems (IVIS)

IVIS Classification

In-vehicle information systems (IVIS) complement ADAS by providing information to the driver. In the 1990s where assistance systems came up inside cars, information systems comprised **informing** and **warning** the driver and to make **action recommendations** (Naab [NR98] or Donges [Don99]). Informing the drivers comprises either to show information relevant to the driving task, e.g., current speed, or to show the status of an assistance system, e.g., detected lane information. The drivers receive information to augment their situation awareness. This information would be hidden without the system. With the fast development in consumer electronics, another category of IVIS becomes more present inside cars. **Infotainment** systems are a mix of entertainment and information systems and are not related to the driving task, i.e., they do not intervene in the driving dynamics (Strassberger [Str07]), but can elsewhere be useful inside a car. Examples of infotainment systems are POI systems and systems which provide information about the weather or news.

Human Machine Interface (HMI)

IVIS use **channels of output and input** to interact with the driver, called **human machine interface** (**HMI**). The goal of the HMI is to make in-car systems (technology) usable (ergonomy) by means of the interface (design). Generally, the design follows a separation of display and control. For **displaying information**, the central information display (CID), the instrument cluster and the head-up display are common channels. Information in the head-up display is generally more urgent and is needed continuously by the driver (e.g., current speed, route guidance, gas level warning). Until the head-up display was introduced into cars, the instrument cluster was responsible for this kind of information. The CID is used for less urgent information such as the map view of the navigation or POI search. In general, visual output is the closer to the glance direction of the driver, the more urgent displayed information is. The CID is **controlled** in many current cars by a central controller like the iDrive controller. Function-specific controls are mounted around the steering wheel. Additional output channels, e.g., acoustic (e.g., park distance control), audio (e.g., route guidance) or haptic (e.g., lane departure warning), are also in use.

Warning Systems

The design of **warning systems** is related to proactive recommender systems as both act without user request. Hoffmann and Gayko [HG09] state that a warning should lead the **attention** of the driver to the right situation, for example, to avoid a crash, and the driver should be supported in making a decision among **alternative actions**. Warning elements are distinguished by their **urgency** and type. The choice of the **output channel** influences the effectiveness of the warning. Visual warnings have a high information rate and can be recognized quickly. Haptic warnings can be recognized even more quickly but are only able to provide low information. Acoustic warnings are only able to be recognized moderately and provide moderate information. The **timing** of a warning system can be early, medium or late. It depends on the amount of information, coverage (availability of resources) and pardonableness (false positives) which timing to choose. Elements with low coverage should warn early to have the possibility to repeat the warning later. The higher the amount of information is, the later an element can warn. With more information, the reaction time is shorter. A warning is less disturbing if it is more excusable, therefore it can be given earlier. Finally, warning elements which indicate an action (e.g., to break) cannot warn early, because they have a low pardonableness.

Information Flow

A challenge for IVIS is **information overload**. Information overload has emerged since cars contain more and more information systems. Additional systems can be applied

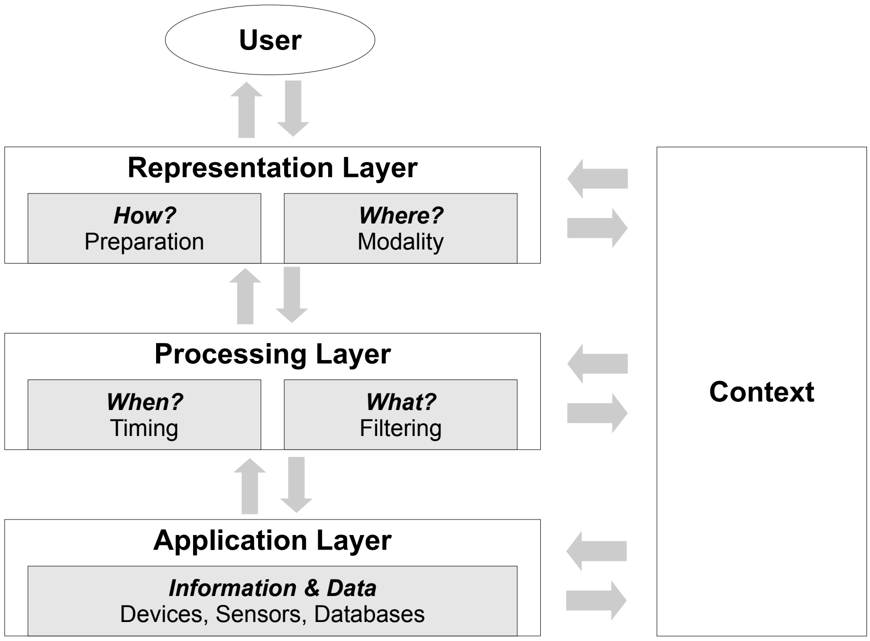


Figure 2.3.: Information flow inside a car (according to Ablassmeier [Abl09])

to regulate the information along the flow of information as described by Ablassmeier [Abl09] (Figure 2.3). Heterogeneous information of driver information systems is on the **application layer**, e.g., sensor data or a POI database. Information is **processed** by deciding which information is relevant for the driver (*What?*) and when to deliver it (*When?*). The **representation layer** prepares information for the output (*How?*) and chooses an adequate channel and modality of output (*Where?*). With the HMI, the driver is able to give feedback to the delivery. The feedback may propagate down the layers. All of the layers may be influenced by **contextual information** which helps to achieve each of the goals of the layer. For instance, a gas level indicator captures the current gas level from the sensor inside the tank on the application layer. On the processing layer, rules trigger a warning or simply forward the gas level. On the representation layer, either the warning is shown in the instrument cluster accompanied by an acoustic output or the current gas level.

Telematics Systems

A special kind of IVIS is a **telematics** system. The term itself is a composition of telecommunication and informatics. It describes information systems that make use of telecommunication technologies for information gathering. Although the term covers many systems such as fleet management or e-learning, we are only interested in in-car telematics. The most popular application of telematics inside a car is a navigation system. **Navigation systems** comprise both the actual routing from one point to another in a street network and services around the routing such as POI search or traffic information. Static information about the environment of the driver is stored in local or distributed **databases**. The most important databases are the digital street map and the storage of POIs. Current development considers making use of navigation systems as **sensor** to support **long-term preview and prediction** capabilities. For instance, Ress et al. [REHK05] describes an "**electronic horizon** (**EH**)" which is a subset of navigation information along the route provided on the internal bus system.

A navigation system consists of several components. The **positioning** system uses the current position of the car to provide navigational services. An absolute position is generally delivered by a **global navigation satellite system** (**GNSS**) like the **geographical positioning system** (**GPS**) with an uncertainty of about 10 to 15 meters (depending on the system and environmental circumstances). Cars also use in-car sensors to improve the accuracy with relative positioning, called dead reckoning. The **route planner** calculates a route based on user input, databases and the position of the car. Common algorithms use modifications of Dijkstra's algorithm [Dij59] to determine the shortest path between two nodes in the street network based on a cost model, e.g., time or length. A **wireless communication** component assists the route planer by providing dynamic information about the current traffic situation, for example, in form of a **traffic message channel** (**TMC**) message. On the go, the **map-matching** component matches information from the positioning system to the street database to show the drivers the current position and guide them to their destination. The **guidance** assists the drivers by selecting next maneuvers. Besides the acoustical or visual output of the guidance, information such as the current position, POIs or the street map can also be shown in a **digital map display**. More detailed information about in-car navigation systems can be found in Kleine-Besten et al. [KBKPS09].

2.1.3. Driver Distraction

Cause

The drivers interact with IVIS by consuming the output or putting information into the system. Output as well as input is a tertiary task concurrently performed with the primary task of driving the car. Therefore, designer of IVIS should take driver distraction into regard. Although distraction is also caused by sources such as eating or talking to a passenger, we are nevertheless interested in how distraction is caused by an information system. For a more comprehensive view, Kircher [Kir07] provides an exhaustive review of driver distraction literature.

The danger of distraction arises when two different tasks are executed at the same time. However, according to Wickens' [Wic84] interference model, a **concurrent task** not necessarily lowers the task (driving) performance if two different human resources are required, e.g., haptic for one task and visual for another. Ranney et al. [RMGG01] distinguish visual distraction (short or long glances away from the road), auditory distraction (voices or environmental noise), physical distraction (hands and feet are doing something else than driving) and cognitive distraction (thoughts not related to the driving task). Cognitive distraction may cause "tunnel vision" (Victor et al. [VHE05]) which lets the drivers stare on the road but ignore environmental circumstances important for the driving task. Research on distraction by visual and interaction-demanding systems such as navigation systems (for example, in Tijerina et al. [TPG98]) shows that visual and physical distraction cause higher performance degradation than auditory and cognitive distraction.

Compensation

The drivers have different conscious and unconscious abilities to **adapt driving behavior** to compensate the demand of the situation (Haigney et al. [HTW00]). On the strategic level, they can decide to postpone and avoid distracting tasks, e.g., interacting with an in-car system. On the operational level, the reaction can be to reduce speed. Chiang et al. [CBW01] found that the drivers reduce speed when they enter a destination into a navigation system. Driver distraction occurs when there are **not enough resources** left to share attention and compensate the demand of a concurrent task by adaptation. It can also occur if the driver focuses on an object, event or person not related to driving (Tasca [Tas05]). If the drivers cannot adapt to the distraction, a crash, misses in navigation or low driving performance, e.g., frequent lane crossings, can be the **outcome of distraction**.

Consequence

Characteristics of the driver, the complexity of the driving task and the current task determine the **level of distraction**. The state of the driver is one of the characteristics. Kopf [Kop05] distinguishes long-term (e.g., experience, load-bearing capacity, personality or disabilities), mid-term or enduring the ride (e.g., fatigue, drowsiness, health or alcohol/drugs) and short-term (e.g., attention, emotion and vigilance) **driver states**. In this context, Shinar et al. [STC05] show that solving a mathematical question representing a complex concurrent task has more negative influence on driving than making an emotional call. Next to characteristics of the driver, also characteristics of the driving task have an influence. For instance, different kinds of traffic situations are differently complex for the driver (Fastenmeier [Fas95]). Other tasks than driving are the third influence factor. If a concurrent task involves other devices, their **design** is important. Design properties are among others, integration into the car HMI, kind of interaction (e.g., iDrive Controller, touch screen) or colors and size of text information. Many studies mention the danger of using a mobile phone while driving (a review is provided by Young and Regan [YR03]). The design as well as legislature may lower potential driver distraction (e.g., prohibit the usage of mobile phones while driving).

The danger of distraction increases as soon as the driver **interacts** with IVIS. Tijerina et al. [TPG98] show that putting information into a navigation system requires long completion time, long eyes off the road and frequent glances on the device. Dingus et al. [DMH$^+$95] found that a voice-based turn-by-turn guidance has lower demand of the driver compared to map-based guidance, especially for complex maps. Therefore, Green [Gre99] proposes a 15-second completion rule for navigation-related tasks involving visual display and physical control.

2.2. Context and Situation

Research in the area of proactive information retrieval has shown that the usefulness of such systems depends on their capabilities to retrieve information depending on context (e.g., Rhodes [Rho03] and Leake and Scherle [LS01]). Nevertheless, it is difficult to define what context is at all. Bazire and Brezillon [BB05] found 150 definitions of context in their literature search and Adomavicius and Tuzhilin [AMK11] add that every application domain of context uses its own tailored definition. In automotive research, authors such as Ablassmeier et al. [APRR07] emphasize the role of context for information delivery inside a car. In this section, we define our understanding of context for this work. Especially in the automotive area, the situation of the driver influences the consumption of information. Situation and context share some characteristics. Similar to context, the term situation is often used with different meanings or even equivalent to context. We also define how we use situation in this work.

2.2.1. Context

Definition

Context is defined in mobile systems as information that is additionally used to enhance mobile applications. In early definitions, researchers listed information that belongs to context. **Location** is one of the most applied context types in the mobile environment but not the only relevant one (Schmidt et al. [SBG99]). Schilit and Theimer [ST94] state that **user context** (profile, location, people around and the social situation), **physical context** (light, noise, traffic and temperature) and **computational context** (connectivity and nearby resources, e.g., printer) are the most important aspects of context. Brown et al. [BBC97] add the **time** of day or the **season** of the year. Ryan et al. [RPM98] additionally use the **environment** and the **identity** of the user and Dey et al. [DAS01] add the **emotional status** and the **attention** of a user. Wörndl and Groh [WG07] emphasize the usefulness of physical and **social** context in case of recommender systems.

Dey et al. [DAS01] note that "the notion of context is still ill defined" by that time. They define context in a **generic** way:

> Context is any information that can be used to characterize the situation of entities (i.e., whether a person, place, or object) that are considered relevant to the interaction between a user and an application, including the user and the application themselves.

The definition of Dey et al. refers to context in computer systems with a user and an application. Information comprising context in the interaction is kept open but should be relevant for the situation of an entity. Winograd [Win01] refers to the work of Dey et al. and notes that "context is an **operational** term" because "something is context because of the way it is used in interpretation, not due to its inherent properties". That means that the same information can be context in one case and no context in another. For instance, the identity of a user is generally no context, but if demographic information about the users is known to the system, it becomes context. The interpretation of information as context depends on the entity it is interpreted for. Kosch [Kos05] emphasizes that context is always connected to an entity and never occurs detached from it. Schmidt et al. [SBG99] state more precisely what context for an entity is. They refer to context as the "knowledge about the user's and IT device's **state**". Similarly, Strang et al. [SLPF03] refer to **context information** as part of context that "characterizes the state of an entity concerning a specific aspect". They explain **aspect** as a specific dimension of the state space with a semantic meaning. An aspect is characterized by a **scale** of possible values that it adopts. Finally, they define context itself as the **set of context information** relevant for the **task** of the user if the entity is a user. If this task is executed by means of interaction between the user and the application, the view on context of Strang et al. and Dey et al. overlap.

Classification

Research in **context-aware systems** mainly focuses on context related to mobile devices (e.g., see a review of applications in Chen and Kotz [CK00]). We are especially interested in context concerning a **driver**. Hoch et al. [HSAR07] describe the principle idea of BMW's "ConntectedDrive" vision based on three major sources of context relevant in the automotive environment (Figure 2.4). The **environment context** comprises the surroundings of the car (e.g., traffic participants, weather, road surface conditions and facilities and buildings around). The **car context** tells us something about the car itself and its components (e.g., current pressure of tires, oil or gas level, position or velocity). The **driver context** includes the mood of the drivers, their physical condition (e.g., fatigue), what the drivers are doing or where they are looking. Together, these three types of context form the so-called "situational context" (Hoch et al. [HSAR07]) as an abstract construct.

Figure 2.4.: Automotive context (Hoch et al. [HSAR07])

For context modeling and incorporation, it is common to differentiate **context types** (see Strassberger [Str07]). Dey et al. [DAS01] distinguish between **low-level** and **high-level** context. Sensors directly observe low-level context and high-level context is derived from observed context. Furthermore, Dey and Abowd [DA99] refer to **primary** context as information that directly characterizes the situation of an entity (e.g., its name) and leads to **secondary** context (e.g., its email address). Strassberger [Str07] further distinguishes **raw sensor information** and its mapping to the scale of an aspect as **abstract sensor information**. Both are low-level context. The fusion of a set of sensor information that refers to the same aspect is called **aggregated information**, e.g., several sensors for the speed of a car. **Derived information** is received by interpreting low-level information. Aggregated as well as derived information are high-level. Mapped on the understanding of context of Dey and Abowd, raw sensor information is primary context and the rest is secondary context.

Different to the classification in types, Dourish [Dou04] makes a distinction between a representational and an interactional view. In the **representational view** the type of context to be processed is known in advance and its current value has to be captured during the run-time of the application. The **interactional view** assumes that the behavior during interaction depends on context, but context itself does not have be known by the system itself. A bidirectional view of context influencing actions and vice versa is assumed.

Incorporating Context

For the incorporation of context into applications, we distinguish the terms awareness, adaptivity, dependency and sensitivity. **Awareness** means to process context in some way. One of the ways is **adaptivity**. It comprises a change of behavior. We distinguish two kinds of behavior change. A system is **dependent** if its functionality only works when context is known. It is **sensitive** if it changes its output based on context, e.g., the position of an item in the ranking. This definition is consistent to Dey et al. [DAS01]. The authors state that "a system is context-aware if it uses context to provide relevant information and/or services to the user, where relevancy depends on user's task". Chen and Kotz [CK00] distinguish **active context awareness** where an application changes its behavior due to context and **passive context awareness** where an application only presents a new context but does not process it.

Context Modeling

To process contextual information, it has to be **modeled** in an adequate and computer understandable way. In a recent survey by Bettini et al. [BBH$^+$10], further information about context modeling can be found. Strang and Linnhoff-Popien [SLP04] conclude on an exhaustive survey of context modeling methods that **ontologies** are the most expressive method to describe context. They also list characteristics of context processed by a system. The **quality** of context depends on the sensor that captures the information and therefore may differ based on the sensor itself and the environment. Information that is sensed can be **incomplete** or **ambiguous**. Depending on the application, different level of **granularity** may be required.

The work of Baldauf et al. [BDR07] investigates several software **frameworks** for context-aware systems. A typical structure is a **hierarchical** infrastructure with a layered architecture. The highest layer comprises the application itself. It communicates with a central broker or manager of context information that typically supports a heterogeneous context model. On the lowest level, resource services (e.g., sensors) deliver information and context recognizers provide functionality to process information.

2.2.2. Situation

Definition

We define a situation as an **abstract** construct that does not exist without context and cannot be measured by sensors. Anagnostopoulos et al. [ANH07] state that "situations are viewed as logically aggregated pieces of context". In contrast to context, a situation occurs **detached** from an entity (Kosch [Kos05]). Entities are rather involved in situations. Multiple entities can share the same situation but need not to have exactly the same context. For instance, several cars can be in a congested traffic situation at the same time but their speed context is not necessarily the same. It may differ slightly. Even if two contexts have the same value, they belong to separate entities. Change of context information of one entity does not affect the other. Regarding modeling of context and situation, a context object belongs to exactly one entity object and several entity objects may share a situation object. If a situation changes all entities are affected. This is consistent to Dey et al. [DAS01]. The authors state that context "characterize the situation of entities". It follows a **direction of causality**. Situations are caused by changes of individual context of entities. Situation changes do not cause the change of context. A situation **exists** as long as at least one entity is in that situation. An entity **enters** a situation if the situation already exists and context information of the entity takes similar values as context information of other entities in that situation.

The definition of situation of Strang et al. [SLPF03] is "the set of all known context information". They mention *all* context information in contrast to only relevant context information. The consequence would be that only **one situation** exists. Similar to the definition of context that should be relevant for the interaction (Dey et al. [DAS01]), we define a situation by distinguishing between context information that is more relevant and such that is less relevant for a specific situation. Hence, **more than one situation** exists. For instance, the speed of a car makes a strong contribution to the situation "We are in a traffic jam", whereas the volume of the music system inside a car has nearly no contribution to "We are in a traffic jam" but to the situation "It is noisy". We only need to measure the set of context information that sufficiently defines a situation.

A situation depends on the specific **understanding** of a user or a system of the underlying context information. Barwise [Bar81] describes a situation as an excerpt from the reality that people perceive and live in.

Characteristics

In contrast to context, we do not distinguish hierarchical structure of granularity and situation information and situation. Situations can only be derived from other situations. Furthermore, context has no semantic information. On the other hand, individual context information describes so-called aspects which are semantic information (Strang et al. [SLPF03]). In our understanding, situations include **semantic information**. Based on these observations, we treat situations as abstract context information with special characteristics. Similarly, Anagnostopoulos et al. [ANH07] emphasize that "situation awareness is considered as the particular kind of context awareness".

Temporal aspects are an important characteristic of situations. Situations appear in a range of time, whereas context is considered independent from time. Situations that occur over a range of time are not disjoint and therefore have temporal dependencies. According to Pfennigschmidt and Voisard [PV09], a situation comprises situation characteristics itself and an interval $[t_b, t_e]$ in which a situation **begins** and **ends**. There are also relationships among situations in time depending on different granularity of time (e.g., minute, hour, day, month). The authors call a set of non-overlapping situations over time a **situation sequence**. From the field of **temporal reasoning** by Allen and Ferguson [AF94], we adopt the relationships which situations can have. Situations X and Y in a sequence use the relations: X meets Y, X takes place before/after Y or X starts/ends Y. Additionally, situations in different sequences can use: X during Y or X overlaps with Y. Temporal reasoning is especially valuable for generating plans which contain actions in a sequence.

Situation Awareness

Research in context and context awareness focuses on the exploitation of context in computational systems. **Situation awareness** is driven, mainly in cognitive science, by the concept which people use for performing tasks, e.g., in surgery or aviation. Billings [Bil95] describes situation awareness as "an abstraction that exists within our minds, describing phenomena that we observe in humans performing work in a rich and usually dynamic environment". Endsley [End00] simply says that "situation awareness is knowing what is going on around you". She emphasizes that the more data that is produced does not necessarily mean that more information is needed. We do not need to know everything but depending on our goals and the nature of our decision task, we need to understand things around us. For example, a surgeon and a pilot need to know different things because they have different goals. This work follows the definition of situation awareness by Endsley's [End00]:

> The perception of the elements in the environment within a volume of time and space, the comprehension of their meaning, and the projection of their status in the near future.

Endsley [End00] defines a **general model** for situation awareness (Figure 2.5). Situation awareness is in between a system and a user included in the process of performing a task. **Individual factors** such as goals, **personal characteristics** such as the memory and **system characteristics** mainly influence situation awareness. Situation awareness is used to make **decisions** that lead to actions. **Actions** are part of a **plan**. Anagnostopoulos et al. [ANH07] emphasize that "uncertainty plays a central role in the area of situation awareness" because context is subject to imperfection. Even if situation awareness is entirely perfect, incorrect decisions can be made with an incorrect decision model. The core of situation awareness contains three levels:

- 1. Level: The **perception** of information the situation awareness is based on is fundamental. The more information is missing or inaccurate the worse the picture of the environment becomes which may lead to wrong decisions. A system perceives its environment by means of sensors and through communication with other systems.

- 2. Level: The **comprehension** of the situation combines pieces of information according to their relevance towards the decision task. Not all information that is perceived needs to be interpreted but only a subset of relevant information. The interpretation involves a subjective (awareness) and objective (situation) component because every comprehension is a subjective interpretation of how things really are.

- 3. Level: On the highest level, a **prediction** of future situations is done. The prediction is done based on the comprehension of current situations (Level 2) or based on perceived states of the environment. Both the situation and its implication can be part of a forecast. Forecasts allow for timely decisions.

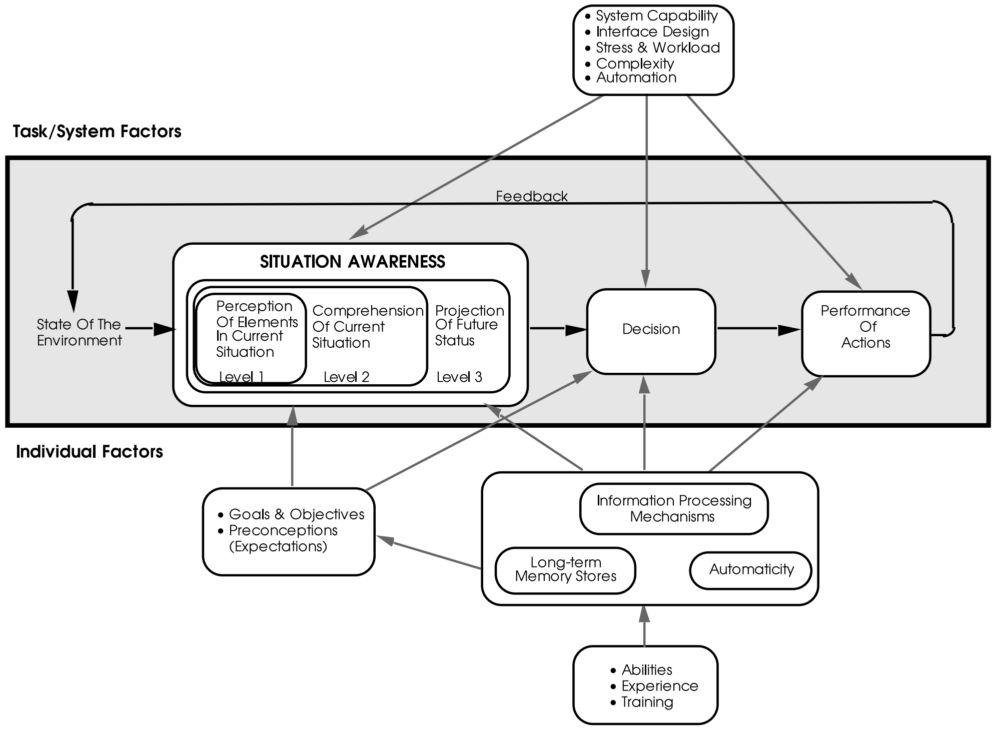


Figure 2.5.: Model of situation awareness in dynamic decision making (Endsley [End00])

Time plays an important role on Level 2 and 3 of the model. The system has to understand how much time it has to make a decision or perform an action. It also should measure how soon projected situations have an impact on decisions.

McGuinness and Fox [MF00] propose a fourth level called **"Resolution"** on Endsley's model. It decides about an action from a set of available actions. The analogy is to ask what shall we do, after we know the facts (Level 1), what is going on (Level 2) on and is going to happen (Level 3). To make such cognitive concepts of situation awareness applicable for computation systems, models for data fusion, e.g., the JDL model (Llinas and al. [LBR+04]), were proposed (see Esteban [ESW+05] or Salerno et al. [SHBB03] for a review).

2.3. Decision Making

Situation awareness provides fundamental information for making good decisions during a task. Recommender systems assist the user with the task of finding appropriate items. This involves making a decision towards one or more items. Jameson [Jam11] emphasizes the role of results from psychosocial research in decision making for recommender systems. Incorporating these results may lead to better results in the usability of the system. A proactive system may also take over some of the decisions to assist the user. To do this adequately, the same decision strategies should be used by the system as are used by the user. This section comprises general aspects of decision making, which strategies are applied by people and which methods exists for computational decision making.

2.3.1. Background

Decision Strategies

The researchers in utility theory assumed that people make decisions by **maximizing benefit** and **minimizing costs** which is called a **rational** decision. This view was extended and refined by findings in the last couple of decades. The prospect theory model of Kahneman and Tversky [KT79] describes how decisions are made if probabilities (prospects) are assumed with alternatives. Such **uncertainty** is common for real world decisions. The model assumes that people tend to have an aversion against lost. Simon [Sim97] found that decisions are subject to bounded rationality. Decision making is **limited** by available information and its certainty, the cognitive ability of people and the amount of time to make a decision. The result is that decisions are often made without taking all criteria into account but more out of a **"gut feeling"** (e.g., Gladwell [Gla05] or Gigerenzer [Gig07]). Simple rules and **heuristics** are applied to make decisions. Gigerenzer [Gig07] shows that the quality of these decisions is often better than with exhaustive assessment. Payne et al. [PBJ93] refer to the principle of least effort by Zipf [Zip49] to explain the usage of simple decision strategies such as heuristics. People choose the decision strategy that provides **enough accuracy** for the current task with **minimal cognitive effort**.

Payne et al. [PBJ93] emphasize that the selection of a **decision strategy** is adapted to context and situation of a decision. Influencing factors are the nature of the decision problem (the task itself and its context), the person (cognitive abilities, prior knowledge) and the social context. **Task complexity** comprises the number of alternatives and attributes and if time pressure is involved. Miller [Mil56] shows that the maximum number of items such as alternatives or attributes that can be processed at once is **7 plus minus 2**. **Context effects** comprise such aspects like the similarity of alternatives, the quality of the alternative set or framing effects. Framing effects mean that the decision makers only use the information that is displayed and the way it is displayed (Slovic [Slo72]). As a result, **inconsistency in preferences** can occur if alternatives are presented in a different kind (Kahneman and Tversky's prospect theory [KT79]).

Choice

The process of decision making results in a final **choice**, e.g., to choose an item among alternatives from a recommendation. Early studies in psychology demonstrated that intrinsic motivation and task performance increase with choice (see Deci and Ryan [DR85]). According to Iyengar and Lepper [IL00], this only means that choice is better than no choice but does not tell anything about the number of alternatives. In fact, early studies incorporated only small sets of alternatives. Iyengar and Lepper show that **less choice** leads to a higher probability of purchasing products and higher satisfaction with the selection compared to choosing among large sets of alternatives. With a more complex decision space, people tend to switch to simple heuristics and rules or avoid choosing at all (e.g., see Shafir [SST93]). Faced with large decision spaces, people tend to create so called **consideration sets** as a trade-off between decision costs and benefits of a choice (Hauser and Wernerfelt [HW90]). They risk to leave an alternative out which should belong to the set or add one which is of poor quality (Keller and Staelin [KS87]). Methods to reduce the set of alternatives are to ignore dominated alternatives (Keeney and Raiffa [KR76]) or to incorporate user preferences. However, these methods require making trade-offs and making trade-offs imposes effort on people (Luce et al. [LPB99]).

People want to **justify** their choice to themselves and others. They seek for reasons to make a decision (Shafir [SST93]). These reasons can be rational and bound to criteria but can also be decoupled from criteria.

Time of Outcome

Another influence factor for decisions is the **time of outcome** of a decision, e.g., the decision to save money with the outcome of a new car in some years. The benefit of an event is perceived as higher if the event takes place sooner than later (Löwenstein et al. [LÖ3]). A disadvantage of immediate benefit maximization is that more decisions have to be made over time.

2.3.2. Human Decision Making Strategies

Classification

People use several decision strategies flexibly (Payne et al. [PBJ93]). **Compensatory** strategies evaluate every relevant attribute of the alternatives and make trade-offs. Bad attributes of an alternative can be compensated by good attributes. **Non-compensatory** strategies only regard a subset of attributes. In contrast to compensatory strategies, they avoid conflicts. Compensatory strategies impose in general more cognitive load. These strategies differ in the **amount of information** processed. Either every relevant attribute is evaluated or the amount is reduced, e.g., to lower cognitive load. Furthermore, attributes can either be processed across alternatives (**attribute-based**) or within each alternative separately (**alternative-based**). According to Russo and Dusher [RD83], attribute-based strategies impose less cognitive load. The strategies also differ in the **formulation of alternatives**, for example, by using an overall evaluation metric like a score. Finally, strategies may calculate metrics to compare alternatives (**quantitative**) or alternatives are compared directly (**qualitative**).

Strategies

The **weighted additive** (**WADD**) strategy (e.g., in Keeney and Raiffa [KR76]) is a rule that considers all relevant attributes of each alternative and the preference weights of the decision maker to determine an overall evaluation metric. To simplify the WAAD strategy, Dawes [Daw79] suggests a heuristic called **equal weight** (**EQW**) that ignores weights or probabilities but evaluates all alternatives and attributes. Similar to WAAD, Tversky [Tve69] describes the **additive difference** (**ADDIF**) strategy. The difference of each attribute is weighted and summed in a pairwise comparison.

Simon [Sim55] noticed that decision maker often do not optimize the output of a decision (rational) but are satisfied if an alternative is found which is "good enough". The heuristic is called **satisficing** (**SAT**). Alternatives are evaluated one after another and in case an alternative is found which fulfills the preference requirements, then the process is stopped. The outcome depends on the order or processing. Variants of the strategy involve all attributes or a subset for the assessment. The heuristic is often used when the set of alternatives is large.

Another heuristic that can be applied in case of large choice sets is **elimination-by-aspect** (**EBA**) (Tversky [Tve72]). It orders the most important attributes (aspects) and a predefined cut-off level eliminates all alternatives that do not exceed the level.

Ordering of attributes is also done by the **lexicographic** (**LEX**) heuristic. Hereby, attributes are ordered according to the preferences of the decision maker (Fishburn [Fis74]). Equal alternatives concerning the most important attribute are compared on the second most important attribute and so on.

Another group of strategies counts good and bad attributes of alternatives. Russo and Dosher [RD83] describe the **majority of confirming dimensions** (**MCD**) heuristic. Alternatives are evaluated pairwise on each attribute and the alternative with the most winning attributes is compared to the next. This is similar to ADDIF but without user weights. Alba and Marmorstein [AM87] describe a similar heuristic called **frequency of good and bad features** (**FRQ**). A metric for grouping attributes in a binary set of good and bad is chosen, e.g., a cut-off level, and the frequency of attributes in these groups are counted for each alternative. It corresponds to a type of **voting** for alternatives.

2.3.3. Multi-Criteria Decision Making (MCDM) Methods

The challenge of comprehensive strategies is to aggregate attributes of alternatives to a final assessment. Research in multi-criteria decision making (MCDM) defines a set of mathematical tools to aggregate multiple criteria or to select alternatives. Criteria can be either attributes or high-level information based on attributes. The methods are developed for complex decisions tasks, e.g., buying a new car or choosing a financial product. Fülöp [F̈05] groups the methods in **multi-attribute utility theory** (**MAUT**) (Keeney and Raiffa [KR76]) and **Outranking**. There are also methods to select alternatives based on **dominance** of criteria. For all methods, alternatives and criteria can be set up as a decision matrix:

	Criteria			
Alternatives	C_1	C_2	$\cdots$	C_N
	W_1	W_2	$\cdots$	W_N
A_1	a_{11}	a_{21}	$\cdots$	a_{1N}
A_2	a_{21}	a_{22}	$\cdots$	a_{2N}
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
A_M	a_{M1}	a_{M2}	$\cdots$	a_{MN}

A multi-criteria decision problem is the selection of the best alternative A_i from a decision matrix $M \times N$. In case of recommendation systems, an alternative is an item and a_{ij} is the characteristic of a criterion C_j of this item. Every criterion has a weight W_j which is either relative with $\sum W = 1$ or absolute, depending on the method. The weights correspond to user preferences.

MAUT

MAUT methods maximize a function. The main characteristic of MAUT is that bad performing items on one criterion can be compensated by good performing criteria. The **weighted sum model** (**WSM**) is the simplest MAUT method and calculates the weighted mean score S_i over every criterion for all alternatives with the Equation 2.1.

$$S_i = \sum_{j=1}^{N} a_{ij} \times w_j \tag{2.1}$$

This allows ranking the alternatives according to S_i but fails if the units of the criteria are not equal. The method also disregards the range of the values within a criterion. To cope with heterogeneous information, the **weighted product model** (**WPM**) approach can be used. It multiplies instead of sums up the criteria and powers instead of multiplies the weights to calculate the score S_i (Equation 2.2).

$$S_i = \prod_{j=1}^{N} a_{ij}^{w_j} \tag{2.2}$$

WPM has to be applied carefully to attributes that can be 0 because the output would always be 0 independent from other dimensions. Contrary to WSM and WPM, the **analytic hierarchy process** (**AHP**) (Saaty [Saa80]) uses relative importance of alternatives and calculates assessments pairwise. It decomposes the decision problem into hierarchies by comparing each alternative in each criterion dimension separately. A pairwise comparison does not require the scale of the criteria to be ordered, e.g., a numeric scale, because the assessment is built by comparing two alternatives directly. First, the decision maker assesses all criteria with a relative importance. The importance scale used in the original AHP is:

1	Equal importance or preference
3	Moderate importance or preference of one over another
5	Strong or essential importance or preference
7	Very strong or demonstrated importance or preference
9	Extreme importance or preference.
0,2,4,6,8,10	Values in between

With n alternatives n^2 pairwise comparisons have to be made theoretically. As a_{ji} is equal to $\frac{1}{a_{ij}}$ and no alternative have to be compared with itself, we get $\frac{n*(n-1)}{2}$ comparisons finally. Pairwise comparison is done once for the criteria by building a $N \times N$ matrix and once for all alternatives by building N $M \times M$ matrices. In case the weights for each criterion are already known, e.g., because the user entered them explicitly, only the pairwise comparison for the alternatives is necessary. After the pairwise comparisons, the square matrices are reduced to a priority vector with N respectively M values. The original AHP calculates the maximum Eigen vector of the matrices. Calculating the Eigen vector involves the successive multiplication of the matrix with itself and building the normalized check sum of the rows. If the difference between consecutive vectors becomes very small, the process stops. The last step in AHP is to use other MAUT methods such as WSM or WPM to calculate the final assessment.

Outranking

In Outranking, alternatives are compared to the ideal alternative and the worst alternative. The first step is to construct an outranking relationship by determining the preference between two alternatives. The second step is to evaluate the relationship to construct a preferred set of alternatives or a ranking. In contrast to MAUT, Outranking methods can cope with weak as well as with strict preferences that often results from imprecise preferences models in reality. Among several existing methods, we select the **technique for order preference by similarity to ideal solution** (**TOPSIS**) (e.g., in Hwang and Yoon [HY81]) as Outranking method because it is more generic. Another popular method called ELECTRE only provides a set of preferred alternatives and no explicit ranking. The basic idea behind TOPSIS is to rank the alternatives by the shortest distance to the ideal alternative and the longest distance to the worst alternative. TOPSIS starts with converting M attribute values a_{ij} for each alternative into dimensionless values using the Equation 2.3.

$$r_{ij} = \frac{a_{ij}}{\sqrt{\sum_{i=1}^{M} a_{ij}^2}} \tag{2.3}$$

Next, a weighted normalized matrix V is constructed using normalized weights W with $\sum W = 1$. The ideal value A^* (Equation 2.4) and the worst value A^- (Equation 2.5) are taken from the weighted matrix V.

$$A^* = \{(\max v_{ij} | j \in J), (\min v_{ij} | j \in J^*) | i = 1, \cdots, M\} \tag{2.4}$$
$$A^- = \{(\min v_{ij} | j \in J), (\max v_{ij} | j \in J^*) | i = 1, \cdots, M\} \tag{2.5}$$

The ideal value is a virtual alternative with the "best" criterion from every dimension. Either the value is the minimum if the criterion j is cost oriented (J^*), e.g., detour, or it is the maximum if the criterion j is benefit oriented (J), e.g., cost saving. The worst value is the same but vice versa. Next, we calculate the distance between all alternatives and the ideal and the worst value by using the Euclidean distance (Equations 2.6 and 2.7).

$$S_{i*} = \sqrt{\left(\sum (v_{ij} - v_{i*})^2\right)}, i = 1, 2, 3, \cdots, M \tag{2.6}$$
$$S_{i-} = \sqrt{\left(\sum (v_{ij} - v_{i-})^2\right)}, i = 1, 2, 3, \cdots, M \tag{2.7}$$

The last step aims to calculate the relative closeness to the ideal solution. It should have the longest distance to the worst solution simultaneously. This is done by $C_{i*} = S_{i-}/(S_{i*} + S_{i-}), 0 \leq C_{i*} \leq 1, i = 1, 2, 3, \cdots, M$. The result is a vector $C = (C_i, ..., C_n)$ which enables us to rank the alternatives. To receive values in $[0, 1]$, all elements of the vector have to be normalized with $\sum C$.

Dominance Filtering

Instead of calculating single assessment scores from several criteria, alternatives can also be filtered directly. The **Skyline** operator is a technique from database engineering

(Boŕzsoňyi et al. [BKS01]) to classify a set of dominant items with multidimensional criteria. As there is no single solution in a multidimensional optimization problem, the dominance of an alternative $\vec{a} = (b_1, ..., b_n)$ over an alternative $\vec{b} = (b_1, ..., b_n)$ is defined with the so-called **Pareto dominance** in Equation 2.8.

$$\vec{a} >_D \vec{b} \leftrightarrow \forall i : a_i \leq b_i \wedge \exists k : a_k < b_k \tag{2.8}$$

The alternative $\vec{a}$ Pareto dominates alternative $\vec{b}$ if $\vec{a}$ is better in at least one dimension and equal in all others. The more the dimensions are correlated, the fewer elements are classified. The amount of alternatives in the Skyline set also increases with the number of dimensions. Skyline can be applied without having the exact utility function of an attribute. It has only to be known if the attribute is monotonically increasing or decreasing. For many attributes this can easily be derived, e.g., a cheap gas station has obviously more utility than an expensive one.

The regular Pareto dominance uses stringent comparison between the dimensions. Therefore, items such as $\vec{a} = (8.75, 7.5)$, $\vec{b} = (8.70, 7.55)$ and $\vec{c} = (1.25, 7.6)$ are all in a Skyline set. To reduce the set further, we make the Skyline comparison **fuzzy** with a fuzzy factor μ in $[0\%, 100\%]$. This excludes either $\vec{a}$ or $\vec{b}$ because they are too similar and $\vec{c}$ because it only dominates slightly in one dimension and is much worse in the other dimension. A fuzzy factor μ extends the comparison operators $>$ (Equation 2.10) and $=$ (Equation 2.9).

$$a_k =_\mu b_k \leftrightarrow |\frac{a_k - b_k}{r_k}| < \mu \tag{2.9}$$

$$a_k >_\mu b_k \leftrightarrow \frac{a_k - b_k}{r_k} > \mu \tag{2.10}$$

$\vec{a}$ μ dominates $\vec{b}$ if $\vec{a}$ is much better ($>_\mu$) in at least one dimension and equal, better or worse ($=_\mu$) in all other dimensions. If $\vec{a}$ and $\vec{b}$ are equal, better or worse in all dimensions, only one item is taken randomly. The value r is used for normalization and is the upper range border of the kth dimension.

A different way of reducing the Skyline set is to loosen the dominance requirement on k **dimensions** where $k < n$ for the number of dimensions n (Chan et al. [CJT$^+$06]). This can especially be useful if the number of dimensions is large because it is more unlikely that the Pareto dominance condition is fulfilled. Alternative $\vec{a}$ k-dominates $\vec{b}$ if it is better in at least one dimension and equal in k other dimensions. The k Skyline set is a subset of the Skyline set and becomes equal for $k = n$. It is also possible to incorporate user weights. User weights W indicate the relative importance of a dimension. This can be exploited in an extension of the k Skyline to the ω **Skyline** by using a threshold weight ω on k dimensions. Then, alternative $\vec{a}$ ω dominates $\vec{b}$ when the sum of weights of all regarded dimensions exceeds ω.

2.4. Intelligent Systems

Besides the selection of relevant items, a proactive recommender also triggers the delivery of recommendations automatically. To be able to decide the right moment for the delivery "intelligently", the proactive recommender needs a component that processes information towards this decision. Relevant information to process in this case is context information, situations of the users or user preferences. The research area of intelligent systems provides well-defined methods for information processing. However, real-world information involves uncertainty and even if we have perfect knowledge, the decision is subject to uncertainty. Therefore, only methods that cope with uncertainty are relevant in our case. In this section, we focus on fuzzy logic and Bayesian networks because these methods are applied in our system. There are many other approaches from the field of intelligent systems that could be used in our case but these approaches are out of scope for this thesis. We refer to basic literature in this field of research (e.g., see Russell and Norvig [RN02]). Furthermore, we describe basics of explanations in Bayesian networks to make the intelligence of the system comprehensible for the user. Finally, feature selection methods are used to investigate situations relevant for a decision.

2.4.1. Fuzzy Logic

Fuzzy Sets

Fuzzy logic is dated back to Lotfi Zadeh [Zad65]. It tries to cope with uncertain and vague information by mapping this information to a set. In contrast to first-order logic, a fuzzy variable can have more than one value at the same time. The term fuzzy logic comprises mathematical models as extension to set theory and an inference mechanism.

A **fuzzy variable** maps a **crisp** value like 15° to one or more fuzzy values like "mild" or "cold" with a certain degree of membership. The fuzzy values form a **fuzzy set** A which members are taken from a **universe** U. Like a discrete set, a fuzzy set contains a finite number of elements $u \in U$. Additionally, a total projection $\mu_A : U \rightarrow [0,1]$ called **membership function** defines the degree of membership of an element $u \in U$ to the set A (Equation 2.11).

$$A := \{(u, \mu_A(u)|u \in U, \mu_A(u) \in [0,1]\} \tag{2.11}$$

The interval $[0,1]$ is called **membership space** M. The set A corresponds to a regular discrete set if the membership function μ_A maps on the values 0 or 1 for all crisp values. Usually, there are fuzzy **linguistic terms** such as "low" or "warm" which are associated with the values u. We distinguish **primary terms** such as "low" and "warm" from terms with **modifiers** such as "very low" or "rather warm". Membership functions can be arbitrary but because of better computational processability and storage, simple functions such as triangular, trapezoidal, Gaussian, sigmoidal or singletons functions are used. These functions can easily be described in mathematical form. Instead of a continuous representation, membership functions can also be a discrete vector of values defining the degree of membership. Properties from set theory such as equality, empty set, subset, complement, union or intersection can be associated with fuzzy sets, too. A fuzzy set is **normalized** if the largest membership value is 1.0.

Fuzzy Inference

In the **fuzzification** phase, crisp values are mapped to the fuzzy set A by determining the degree of membership to one or more values in the set. In contrast to regular discrete sets, an input value can have more than one membership. This corresponds to human perception of terms such as "mild" or "cold". People perceive the transition as smooth rather than abrupt.

To make inference with fuzzy variables by means of a **fuzzy logic system**, a **rule base** is set up. The rules connect linguistic terms and are used in an inference mechanism. The terms are connected by **connectives** (AND, OR, IF-THEN or IF and ONLY IF) and can be negated. In a conjunction with AND, all terms need to be fulfilled, i.e., all terms need to have a degree of membership $\mu_A(u) > 0$. In a disjunction with OR at least one term needs to have a degree of membership $\mu_A(u) > 0$. We apply the most common definition of the operator AND, OR and NOT for fuzzy sets. If $p, q \in A$, we take the minimum membership in case of AND (Equation 2.12), the maximum in case of OR (Equation 2.13) and the complement for the negation (Equation 2.14).

$$\mu_A(q \text{ AND } p) \equiv \min\{\mu_A(p), \mu_A(q)\} \tag{2.12}$$

$$\mu_A(q \text{ OR } p) \equiv \max\{\mu_A(p), \mu_A(q)\} \tag{2.13}$$

$$\mu_A(\text{NOT } q) \equiv 1 - \mu_A(q) \tag{2.14}$$

Rules are constructed as conditional IF-THEN sentences with operators and terms as conditions on the left side (**antecedent**) and the result of inference on the right side (**consequence**). Two kinds of inference methods are distinguished. The **Mamdami-type method** (Mamdani [Mam77]) infers on the membership to a fuzzy set in the consequence. In the **Sugeno-type method** (Sugeno [Sug85]), the variable in the consequence is a linear or constant function. Mamdami is the more common and intuitive method. It is used in systems in which human input and understandability is required. Therefore, we apply this type of method in our research. Sugeno is more suitable for mathematical analysis.

The Mamdami-type inference consists of three steps. First, the **aggregation** of the terms in the antecedent is done by means of the operator MIN, MAX and NOT. Second, the **implication** of the antecedent to the consequence results in a membership $\mu_A^*(r)$ for the value $r \in A$ in the consequence. The Mamdami implication simply takes the minimum of the antecedent membership a and the consequence membership c: $a \rightarrow c = \min\{a, c\}$. The implication is done for every rule in the rule based. If rules are weighted with a certainty grade $w \in [0, 1]$, the implication is extended to $a \rightarrow c = \min\{a, c, w\}$. Third, all implications are aggregated in the **composition** phase with an operator. A commonly used operator is again the $\max_{i=1}^{n}\{\mu_i^*(r)\}$ operator for all n rules in a rule base. The result of the inference is a final membership function for the variable in the consequence.

The optional **defuzzification** phase converts the result in a crisp value again. A commonly used technique is the **center-of-gravity** (**COG**) method. The area defined by the membership and the discrete values represents the result in the consequence. The COG method calculates the centroid of the area and takes the crisp value belonging to the centroid as defuzzified value.

2.4.2. Bayesian Networks

Bayes' Rule

A popular method for reasoning under uncertainty is a Bayesian model. Introducing a **degree of belief** with probability makes decision under uncertain conditions feasible. The basic notation of probability is a **random variable** A that can have either a Boolean, discrete or continuous value range. An atomic event is an assignment of a value out of the value range of A, e.g., $A = true$. The **prior** or **unconditional probability** assigns a degree of belief p to every atomic event independent of other information: $P(A = true) = p$. A **probability distribution** $P(A) =< p_1, p_2, ..., p_n >$ describes the whole set of beliefs for a random variable. The probability of all events of a random variable sums up to 1: $\sum_{i=1}^{n} P(A = p_i) = 1$.

If more than one random variable is involved, $P(A, B, ...)$ describes the probabilities of all possible combinations of these variables, called **joint probability distribution**. For continuous valued random variables, it is not feasible to denote all the single probabilities. There, a **probability density function** $P(A = a) = F(a)$ is used instead. It is a parameterized function of a and can be, for example, normally or uniformly distributed.

Besides prior probability, we are also interested in **conditional or posterior probabilities**: $P(A = a|B = b) = p$. If we observe the event b, A is in state a with a belief of p.

$$P(A|B) = \frac{P(A, B)}{P(B)} \tag{2.15}$$

By means of the **product rule** in Equation 2.15, we calculate the posterior probability with prior probabilities if $P(B) > 0$.

$$P(A,B) = P(A|B)P(B) \tag{2.16}$$

$$P(A,B) = P(B|A)P(A) \tag{2.17}$$

If we reformulate the product (Equations 2.17 and 2.17), the Equation 2.18 of **Bayes' rule** can be derived.

$$P(B|A) = \frac{P(A|B)P(B)}{P(A)} \tag{2.18}$$

Bayesian Inference

We are now able to calculate posterior probabilities from observations (**evidence**). We assume that we have a full joint probability distribution of our environment $P(A, B, ...)$. If we want to know the probability of $P(A)$ out of the joint distribution, we simply sum up all the posterior probabilities over all other random variables B: $P(A) = \sum_B P(A, B)$ (**marginalization**). The same can be achieved if the joint probability is replaced by posterior probabilities according to the product rule: $P(A) = \sum_B P(A|B)P(B)$ (**conditioning**). The probability $P(A)$ resulting from marginalization or conditioning serves as normalization factor to make sure the probabilities of all values for A sum up to 1.

With the Equation 2.15, the **causal inference** calculates $P(A|B)$ with evidence for B. The problem with full joint probability distributions is that they do not scale well for many random variables and it is also impractical to estimate all probabilities in the distribution. Causal inference can be simplified by taking **independence of random variables** into regard. If variables A and B are independent from C, than the equation holds: $P(A, B, C) = P(A, B)P(C)$. Based on this equation, we derive that $P(C|A, B) = P(C)$ which reduces the number of probability estimations in the joint distribution.

In practice, the conditional probability as well as prior probabilities are often known or can easily be estimated. Bayes' rule in Equation 2.18 allows for **diagnostic inference** from the cause A to the effect B. If more than one cause variable is involved, the assumption of independence simplifies the calculation. In this case, we have a conditional independence $P(A, B|C) = P(A|C)P(B|C)$ or $P(A|B, C) = P(A|C)$ of A and B under C. C separates A and B because it is the cause of both of them. With these assumptions, the full joint probability distribution of one cause A influencing many effects $B_1, B_2, ..., B_n$ can be written as **naive Bayes model** (Equation 2.19).

$$P(A, B_1, B_2, ..., B_n) = P(A) \prod_i^n P(B_i|A) \tag{2.19}$$

Bayesian Network

If we construct the Equation 2.19 as a direct acyclic graph (DAG), we receive a **Bayesian network** like Figure 2.6.

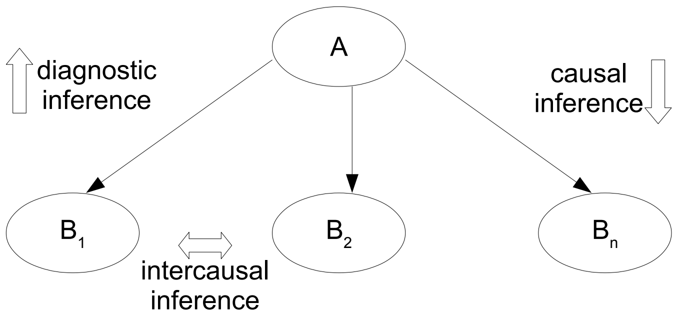


Figure 2.6.: Simple Bayesian network with three types of inference in the network

Random variables become nodes of the network that are connected by arrows from the cause to the effect. Each node is described by a **conditional probability table** (**CPT**) with $P(B_i|Parent(B_i))$. All distributions in the network together determine the full joint probability distribution under conditional independence (Equation 2.20).

$$P(B_1, B_2, ..., B_n) = \prod_i^n P(B_i|Parent(B_i)) \tag{2.20}$$

The nodes in a Bayesian network can be either discrete, continuous or a mix of both. For inference, we distinguish **evidence variables** which can be observed and **hidden variables**. The result of the inference is the posterior probability distribution of the hidden variables. The resulting state of the hidden variables is also called the belief. Bayes' rule in Equation 2.18 is used to marginalize the variables and to normalize them. Several exact and approximate methods for inference are proposed which can handle large networks (e.g., see Russell and Norvig [RN02]).

Learning Parameters

Because estimating probabilities in a network manually is often unfeasible, automated **learning** from data is used. To simplify learning in our case, we assume that complete training data and a predefined structure are known. In this case, we simply use a **maximum likelihood** (**ML**) estimator to determine CPTs. Otherwise, we have to apply methods such as **expectation maximization** (**EM**) with incomplete training data or a local search for structure learning. The ML estimator determines the maximum likelihood for an observation $x_i = j$ and $pa(x_i) = k$ by counting frequencies of the observation (Equation 2.21).

$$P(x_i = j|pa(x_i) = k) = \frac{P(x_i = j, pa(x_i) = k)}{P(pa(x_i) = k)} = \frac{\alpha + N_{ijk}}{\alpha + N_{ik}} \tag{2.21}$$

The number of occurrences of the parent N_{ik} normalizes the number of observations for a specific N_{ijk}. A prior α (e.g., $\alpha = 1.0$) is used because of unreliable estimations of unobserved events. There are several kinds of applying a learning method. We use a **10-fold cross validation** in our tests. The method divides the data in 10 subsets and runs the ML estimator with 9 subsets. The 10th subset is the testing set on which performance metrics such as accuracy are calculated. The procedure is repeated 10 times and each run leaves out another subset.

2.4.3. Explanations for Bayesian Networks

Explanations and Transparency

Research in intelligent systems shows that comprehensibility and transparency are desirable for systems that help the users to make decisions (e.g., in Southwick [Sou91] or Richards [Ric03]), especially because intelligent systems are **black boxes**. The meaning of the term "transparency" in this thesis should not be confused with its usage in the paradigm of "transparent computing" in ubiquitous computing. Authors such as Brown [Bro01] refer to transparent computing in case the actions of a system take place without the notice of the user. In this case, transparent means invisible. In combination with explanations, transparency describes that the user understands the behavior of a system. In this case, transparent means comprehensible. We use the term "transparency" exclusively to describe a comprehensible system behavior in this thesis.

Explanations are a common method to make intelligent systems more comprehensible. They are useful for the user to detect errors in the reasoning process, learn the capabilities of the system or when information is needed to make decisions (Gregor and Benbasat [GB99]). Furthermore, they may contribute to make the user trust the system, persuade the user to select and lead to more acceptance of the system. We focus our description on **Bayesian network explanations**. To explain **rule-based systems** such as a fuzzy logic system, we refer to other approaches such as Southwick [Sou91], Swartout et al. [SPM91], Richards [Ric03] or Nakatsu [Nak04]. In research, especially expert systems and decision support systems in medical diagnosis, e.g., MYCIN (Buchanan and Shortliffe [BS84]), are widely studied.

Classification

In general, explanations can be structured in **what** to explain (content), **how** the system interacts with the user (communication) and to **whom** the explanation is delivered (adaptation) (Lacave and Diez [LD03]). For Bayesian networks, methods that explain the content can be further grouped in three categories. We distinguish nodes in a network by variables that explain (explanatory variables H), variables that are explained (explanandum E) and observations O. The rest of the nodes are part of all available variables V, e.g., unobserved variables.

Explanations for the **evidence** focus on justifications of the evidence (observations) by analyzing the probabilities of unobserved variables (**abductive reasoning**). Abductive reasoning explains a subset of observed variables ($E \subseteq O$) with unobserved or observed variables ($H \subset V$). The explanandum E and explanatory variables H do not share any variables ($H \cap E = \emptyset$). Abductive reasoning is mainly used in domains like medical expert systems to derive a diagnosis about observed anomalies.

Another category of explanations explains the **model**. It makes the knowledge base transparent and helps experts to construct or debug a model or introduce novice users to the capabilities of a system.

The third category explains the **reasoning** of the system. It helps experts to evaluate a system by testing different kinds of evidence and analyzing the inference result. Explaining how the result was reached makes it easier to detect errors in the model. It can also explain why some expected results are not obtained or how the system would behave assuming some hypothetical input. Explaining the reasoning searches for the explanandum E in unobserved variables ($E \subseteq V \setminus O$). These variables are sometimes even unobservable, e.g., the class variable in a Bayesian network. The explanandum is explained by observations ($H \subseteq O$).

Explanation Methods

To make the proactivity of our recommender comprehensible, we use methods to explain the reasoning and abductive reasoning. We restrict our discussion to fundamental methods. A more detailed review of approaches can be found in Lacave and Diez [LD03]. The **most probable explanation** (**MPE**) in a Bayesian network (e.g., explained by Pearl [Pea88]) seeks for the set of assignments A for unobserved variables (explanandum E) which maximizes the joint probability $P(A, E)$ where E is the set of evidence (Equation 2.22).

$$A = argmax_A P(A|E) \tag{2.22}$$

We determine the MPE by repeatedly calculating the joint probability $P(A, E)$ with Equation 2.20. Using all observations for the explanation is called total abduction, otherwise partial abduction. The disadvantage of total abduction is that also irrelevant variables are included, i.e., variables which do not have an impact on the observation. This makes the explanation not compact.

Irrelevance is defined by means of statistical independence proposed by Shimony [Shi91]. If $P(E|A,X) = P(E|A)$, then X is irrelevant. For practical reasons, a less restrictive formulation with $|P(E|A,X) = P(E|A)| \leq \delta$ may be used. Furthermore, if the variable adopts its most "usual" (probable) state in the explanation, it is irrelevant (de Campos et al. [dCGM01]). Finally, the user or an expert may also explicitly determine irrelevant variables.

Another abductive reasoning method is to compare two explanatory variables H_1 and H_2 with the posterior ratio divided by the prior ratio of the explanations for every explanandum variable e (Equation 2.23).

$$Bayes\,factor = \frac{p(H_1|e)/p(H_2|e)}{p(H_1)/p(H_2)} = \frac{p(e|H_1)}{p(e|H_2)} \tag{2.23}$$

If the so called ***Bayes factor*** (Jeffreys [Jef61])is less than 1, then H_2 is preferred. If it is less than 3, then H_1 is slightly preferred and if it is between 3 and 12, then H_1 is strongly preferred. With the $Bayes\,factor$, also explanations for the reasoning can be tested by measuring the impact of an explanatory variable H_i to other explanandums E_1 and E_2, given already observed evidence e, i.e., $e \notin H_i$ (Equation 2.24).

$$Bayes\,factor = \frac{p(H_i|E_1,e)}{p(H_i|E_2,e)} \tag{2.24}$$

Thus, we can compare two explanatory variables H_i and H_j towards their impact on the explanandum E_1 and E_2.

A further method is to measure the influence of evidence by a **cost function**, e.g., the cross-entropy (Suermondt [Sue92]) in Equation 2.25.

$$H(P(H|e),P(H)) = \sum_i p(H_i|e)\log(\frac{p(H_i|e)}{p(H_i)}) \tag{2.25}$$

The amount of information is measured with evidence e by comparing the posterior distribution of H after e with the prior distribution. Furthermore, Suermondt computes with the cross-entropy the importance of the chain of reasoning to H for each evidence e to the inference result, i.e., the final posterior distribution of H.

More recent approaches are proposed by Flores [FGM05] and Nielsen et al. [NPE08] which build **trees** of arguments. The explanation is a path from the root to a leaf.

Their method is tailored to medical diagnosis explanations where the result may also include variables for which no evidence exists but are highly probable because of other variables.

2.4.4. Feature Selection

We investigate the influence of situations on the decisions of a user towards the relevance of a recommendation with feature selection methods. These methods allow assessing the impact of observed context (situations) to a class variable (recommendation).

Entropy-Based Metrics

The **mutual information** (**MI**) (also called **information gain**) of two random variables X and Y measures how much knowing one variable determines and reduces the uncertainty about the other. It is based on **Shannon's entropy** value that indicates the uncertainty of a random variable. For a discrete random variable X with n values, Shannon's entropy is depicted in Equation 2.26.

$$H(X) = -\sum_{i=1}^{n} p(x_i) \times \log(p(x_i)) \tag{2.26}$$

Furthermore, if a variable Y with m values is known, we can measure the remaining entropy of X by the conditional entropy in Equation 2.27.

$$H(X|Y) = -\sum_{i=1}^{n}\sum_{j=1}^{m} p(x_i, y_j) \times \log(\frac{p(x_i)}{p(x_i, y_j)}) \tag{2.27}$$

Based on these equations, the mutual information is composed of the joint distribution of X and Y and their individual distributions (Equation 2.28).

$$\begin{aligned} I(X;Y) = \sum_{y \in Y}\sum_{x \in X} p(x,y) \times \log(\frac{p(x,y)}{p_1(x) \times p_2(y)}) = \\ H(X) - H(X|Y) = H(X) + H(Y) - H(X,Y) \end{aligned} \tag{2.28}$$

In case of independence of X and Y the mutual information is 0 because $p(x, y) = 0$. Otherwise, if one variable fully determines the other or both variables are identical, the mutual information reaches its maximum. The mutual information is non-negative and symmetric. It can also be expressed by means of the entropy. The entropy or uncertainty of X is reduced by the uncertainty about X which remains if Y is known. By means of the mutual information, we measure the **redundancy** of a variable by knowing the other.

$$R = \frac{I(X;Y)}{H(Y) + H(X)} \tag{2.29}$$

If the normalized version of the mutual information in Equation 2.29 becomes zero, the two variables are independent, as $I(X;Y) = 0$. If R reaches its maximum, one of the variables is completely redundant. Equation 2.30 determines the maximum value of R.

$$R_{max} = \frac{min(H(Y), H(X))}{H(Y) + H(X)} \tag{2.30}$$

Two times the redundancy yields the **symmetric uncertainty** $SU = 2 \times R$. A problem with mutual information is that it prefers attributes with many values, because the logarithm with the same base can be higher for attributes with more values. The **gain ratio** (**GR**) solves this problem. It penalizes a multi-valued attribute by normalizing with the entropy of the attribute Y (Equation 2.31).

$$GR = \frac{I(X;Y)}{H(Y)} = \frac{H(X) - H(X|Y)}{H(Y)} \tag{2.31}$$

ReliefF (RFF)

The method of **ReliefF** (**RFF**) is based on Relief with the extension of more robustness against noise and the ability to handle more than two valued classes. The basic idea is to evaluate the quality of attributes by the similarity of instances in case of the same class and the difference of instances for different classes. Therefore, for an attribute A k nearest hits H (instances with the same class value) and k nearest misses M (instances with different class values) are searched for a random selected instance R. This is done for m instances and for every instance a quality estimation $W[A]$ is updated. $W[A]$ is decreased for different attribute values A of R and H and increased for different values A of R and M. Additionally, the k nearest instances can be weighted inversely proportional to their distance to R. We apply ReliefF with $k = 10$ (suggested by Kononenko [Kon94]) and with and without a weighting by distance. For weighting an exponential decrease with $\sigma = 2$ is used.

chi square (χ^2) (CHI)

The **chi square (χ^2) (CHI)** test in statistics evaluates a null hypothesis, i.e., an observed distribution is compared with a theoretical distribution. For feature selection, the independence of an attribute A with a class C can be tested with the null hypothesis of independence of the variables. For that, the observed joint distribution O of A and C is compared with the expected joint distribution E if A and C are independent. The last distribution can be derived from the observed distribution of C. Finally, a score with Pearson's chi squared test of independence is calculated (Equation 2.32).

$$\chi^2 = \sum_{i=1}^{n} \sum_{j=1}^{m} \frac{(O_{i,j} - E_{i,j})^2}{E_{i,j}} \tag{2.32}$$

For discrete variables, n is the number of values of A and m for the class C. The smaller the value of χ^2 is, the more independent the variables are.

2.5. Recommender Systems

Recommender systems are a special kind of intelligent systems used to select relevant items for a user. In this section, we describe characteristics and goals of recommender systems. Classical recommender systems are commonly distinguished by the technique they use to select items: content-based filtering, collaborative filtering, knowledge-based filtering and utility-based filtering. For proactive recommendations, context is important. We describe techniques to integrate context into recommender systems. Like for Bayesian networks, we use explanations to make recommender systems comprehensible for the user. Our approaches are evaluated with common metrics from research.

2.5.1. Background

Origin

Recommender systems originate in the research of **information retrieval (IR)**. IR is the process of "leading the user to those documents that will best enable him/her to satisfy him/her need for information" (Robertson [Rob81]). The user who searches for documents (mostly text-based) tries to accomplish a specific task. Searching involves a **query** that is entered in the system and the system provides a list of results. At the beginning of the 1990s, personal computers became more powerful and therefore other kinds of delivering information to the user have been investigated. One of this kind is **information filtering** (Belkin [BC92]). Information filtering is closely related to information retrieval because it has the same goal to provide relevant information but it focuses more on the need and the task of the user. This leads to the usage of **user models** as profiles to capture long-term preferences instead of short-term preferences with a query. The goal of filtering is to remove data, not to find or rank data like retrieval.

Definition

Recommender systems are a kind of information filtering with a special kind of user profile. They are grounded on the idea of people recommending information to others based on **"Word of mouth"** (Resnick and Varian [RV97]). Konstan [Kon04] describes recommenders systems as an automated way to use the opinions of a **community of people** to help people who are **like-minded** in finding information. Resnick et al. [RIS+94] mention the problem of **information overload** where people are overwhelmed by large sets of choice. Recommender systems are proposed as solution for information overload.

In the beginning, recommender systems mainly referred to the social aspect of finding information. Later the term was extended also to non-social aspects. Burke [Bur02] describes a recommender system as "any system that produces individualized recommendations as output or has the effect of guiding the user in a personalized way to interesting or useful objects in a large space of possible options". Hence, another important characteristic of recommender systems is **personalization**. According to Burke [Bur00], recommender systems distinguish from information retrieval by being individual to the user.

Recommender systems can be useful in many ways. Adomavicius and Tuzhilin [ASST05] state that recommenders are useful when the user does not have enough **domain-specific knowledge** to make a decision. For Belkin [Bel00], recommenders are useful because **users' behavior** in information seeking sometimes suggests that the users do not really know by themselves what they want. A recommender may give a helpful hint in this case. However, Quinn [TQ00] emphasizes that the users that are receiving recommendations should be **tolerant towards inaccuracies** in the result.

Recommenders are used for e-commerce (Ben Schafer et al. [SKR01]) such as *Amazon.com* (Linden et al. [LSY03]), movies such as *Netflix.com* (Bennett and Lanning [BL07]) or Movielens (Dahlen et al. [DKH+98]), news (e.g., Billsus et al. [BBE+02]), restaurants (e.g., Burke [Bur00]) or tourists (e.g., Ricci and Werthner [RW02]).

Calculating Recommendations

Recommender systems suggest items such as products or restaurants to an active user based on implicitly observed behavior or explicit information given by the user (e.g., ratings or preferences). The basic task of a recommender system is to assess a set of **items** I for a **user** U by predicting the item **ratings** R of the users with Equation 2.33.

$$I \times U \rightarrow R \tag{2.33}$$

Like Adomavicius and Tuzhilin [ASST05], we do not restrict recommendation systems to ratings from the users. Rating is defined in this work as any kind of user-specific information towards an item, e.g., utility.

Classification

There are several ways to classify recommender systems. Montaner [Mon03] and Rao [RT08] distinguish approaches by their **domain** in web recommenders, movie recommenders, product recommenders and news recommenders. These approaches can further be differentiated according to their **commercial deployment**, e.g., *Amazon.com*, *Last.fm* and *Netflix.com*, in contrast to a research prototype. Furthermore, the **receiver** of a recommendation can either be an individual, which is the case for most approaches, or a group (e.g., see McCarthy et al. [MSC$^+$06]). Jannach et al. [JZFF11] distinguish three kinds of **purposes** a recommender can follow. The recommender either predicts user preferences for items, interacts with the users to provide a "good feeling" about the system or makes the users to discover new information. Similar to Jannach et al., Herlocker et al. [HKTR04] distinguish systems by the **goals and tasks** of the user. Early systems such as GroupLens extend IR with **annotations in context**, e.g., ratings of the users, to recognize relevant items or **find good items** based on the predicted ratings. For items with high involvement like financial products, a recommender should **find all good items**. In domains where a lot of items are consumed in a short time or in sequence, the recommendation of a **sequence** can be helpful, e.g., music or paintings (e.g., see Bohnert et al. [BSZ09]). Finally, the users may just want to **play** around with the system.

Most authors use a **technical** classification for recommender systems. The simplest classification is in systems that are **collaborative** and consider other users in the process or only the **individual** user. Malone et al. [MGT$^+$87] describe three kinds of information filtering: cognitive (content-based), social (collaborative) and economic. Balabanovic and Shoham [BS97] list three kinds: content-based, collaborative filtering and hybrid. Burke [Bur02] adds demographic, knowledge-based and utility-based approaches and Kautz et al. [KSS97] investigate social network-based approaches. Manouselis et al. [MC07] and Adomavicius et al. [AMRT12] distinguish methods by how many item criteria they take into regard. **Multi-criteria decision making (MCDM)** recommenders use more than one criterion to evaluate items instead of a single criterion like most classical recommenders do.

Outcome

Recommendation system approaches also different in the focus of their output. Towle and Quinn [TQ00] note that classical recommenders operate in a **preference-based** manner based on user preferences. In **need-based** approaches, the need of a user is explicitly incorporated in the recommendation as a selection criterion (e.g., Herlocker and Konstan [HK01]). In case of preference-based approaches, we also distinguish if the system stores **persistent user profiles** like described by Montaner [Mon03] and if it is able to handle **ephemeral users** (Ben Schafer et al. [SKR01]) with short-term preferences as well.

The **output** of a recommender itself and its generation can also be different. Adomavicus and Tuxhilin [ASST05] distinguish **absolute value** recommendations and **relative order** of items, e.g., a ranking. Ben Schafer et al. [SKR01] distinguish whether this output was generated by a **push-based** mechanism, by a **pull-based** mechanism by giving a hint without retrieving the results or **passively** by embedding the recommender in the application like *Amazon.com*. If a recommendation is delivered push-based it can either cover **synchronous** information need with a notification or **asynchronous** information need, e.g., with an email. The **trigger** for pushing messages can be a need of information or preferences of the user. Pull-based delivery is less disturbing than push-based delivery but the probability that the user misses something is higher. Passive delivery is similar to push-based delivery but recommendations are shown constantly embedded in the search results.

2.5.2. Classical Recommender Methods

Content-based Filtering

Content-based recommender systems assume that the users rate items with similar features similarly. They originate from information retrieval and information filtering techniques (Belkin and Croft [BC92]) by incorporating user preferences. Recommendations are calculated by measuring the similarity of unrated items I_{NR} and already rated items I_R (Item-to-item correlation according to Ben Schafer et al. [SKR01]). The content for similarity measurement is extracted from rated items I_R as a set of **features**, e.g., keywords from text documents. The features have weights based on the ratings R the user gave the items. Unrated items I_{NR} with the highest ratings are recommended. Item features can be represented as database-like tables with predefined structures (e.g., restaurants), unstructured data (e.g., text documents or web pages) and semi-structured data with features as well as free text (Pazzani and Billsus [PB07]).

Most content-based recommenders are for documents, web pages, products or news. In general, short-term user profiles are built on search history and saved in the browser, e.g., in a cookie (**ephemeral user**). If this information is saved, e.g., if the user is logged in on a web page, **long-term profiles** can be built. Content-based recommendations are restricted to the features that are associated to an item (**description problem**). Two items with identical feature sets cannot be distinguished in case the user rates them differently. Furthermore, content-based recommendations can be **over-specialized** because only similar items to the ones already rated can be recommended. In news recommendations, this leads to the **portfolio problem** (Belkin and Croft [BC92]). Finally, content-based recommender systems are faced with **ramp up** problems if not enough ratings are available and the **new user** problem if the user has not rated enough items yet (**cold start**).

Collaborative Filtering

Content-based recommender systems only take the preferences of an individual user into regard. **Collaborative filtering** (**CF**) considers the whole community to calculate the rating for a single user. It assumes that users with similar taste rate items similarly. The idea is to capture "Word of Mouth" in computational systems. Early approaches try to achieve this by regarding ratings inside a group of stereotype users (Rich [Ric79]) or to allow the user to make free-text annotations (Goldberg [GNOT92]). The first automatic collaborative filtering approaches are Ringo for music by Shardanand and Maes [SM95] and GroupLens for news group articles by Resnick et al. [RIS$^+$94].

Methods for collaborative filtering are classified according to the underlying processing of data. Breese et al. [BHK98] distinguish two kinds of CF algorithms: Model-based and memory-based approaches. **Memory-based approaches** consider the whole database of rated items I_R and try to calculate some kind of similarity to evaluate an unrated item. Sarwar et al. [SKKR01] add **item-to-item** correlation to the older method of **user-to-user** correlation. If the recommendation problem is regarded as matrix of the users U as rows and the items I as columns and the ratings R as entries, then the user-to-user correlation calculates similarity row-wise to find similar users and item-to-item correlation column-wise to find similarly rated items. A common correlation metric is Pearson's correlation or cosine similarity. Memory-based methods in general do not scale well for large databases. **Model-based approaches** learn a user model from ratings and calculate the prediction or a recommendation based on the model. A popular recent method is to build matrix factorization models (Koren et al. [KBV09]).

Ben Schafer et al. [SKR01] state **conditions** under which a domain is suitable to be improved by a collaborative filter. First, many items and many ratings per item should be available, in general more ratings than items, and the users have to rate multiple items. Second, the nature of the domain should involve some kind of taste and items are assumed to be homogenous items, i.e., items are similar in objective criteria but differ in subjective criteria. Third, items should not change too often (e.g., news are challenging) and taste should not change too fast (e.g., clothing can be demanding). CF is good at finding **niche** items and it does **not require features** of an item such as content-based filtering. A major issue is **ramp up** until a critical mass of users is reached. Furthermore, **cold start** problems arise when a **new user** has to rate items first and a **new item** has to be rated some times. According to Claypool et al. [CGM$^+$99], 98% to 99% of the items do not have ratings if the available set of items is very large. It is difficult to find like-minded users in such a constellation (**sparsity problem**). A consequence is the "**Lemming effect**" described by Klahold [Kla09] where a popular item is often recommended if it is often positively rated.

Knowledge-based Filtering

Both content-based and collaborative approaches depend on ratings a user gives to an item. If not enough ratings are given, cold start and sparsity issues occur. **Knowledge-based recommender systems** are not based on ratings but on explicit models. The knowledge comprises models that match user preferences to item features (e.g., Towle and Quinn [TQ00], Burke [Bur00] or Chun and Hong [CH01]). Preferences are either entered by the user explicitly or are learned by means of machine learning methods. The approach requires initial effort by the engineer of the system, whereas for other recommender methods a user has to put in effort to get good recommendations. Knowledge-based recommenders are applicable in domains where there is a **need for a recommendation** (Towle and Quinn [TQ00]) instead of preference-based domains. For instance, restaurant recommendations may either be provided based on the users' taste or if they are hungry and need a restaurant. Furthermore, all items instead of only rated items can be taken into regard. These reasons make knowledge-based recommenders complementary to other approaches. They are often applied in **hybrid approaches** (e.g., by Burke[Bur00]).

Knowledge-based recommender systems are distinguished by the knowledge source. Burke [Bur02] defines catalog knowledge (item features and properties), functional knowledge (mapping of user preferences to item features) and user knowledge as main sources of knowledge. **Case-base reasoning** (Aamodt and Plaza [AP94]) is a popular method for knowledge-based filtering where the catalog knowledge is represented by cases. Burke [Bur00] proposes a restaurant recommender in which a restaurant is the case and Ricci [RW02] takes previous travel plans as cases in his travel recommender. Another approach is a **constraint-based** recommender system like described by Felfernig and Burke [FB08]. The user model (functional knowledge) is represented by constraints to the items. The more the recommendations satisfy these constraints, the better they are (constraint satisfaction [Tsa93]). Many approaches are also based on a **conversation** with the user to refine the knowledge of the system about the user (e.g., Burke [Bur00], Mirzadeh et al. [MRB05]), e.g., by critiquing the recommender (Smyth et al. [SMRM04]).

The knowledge base is also a valuable source to generate **explanations** for recommendations (Felfernig et al. [FB08]). However, the knowledge base has also disadvantages. The effort of setting it up is transferred from the user (ratings) to the **knowledge engineer** who has to build a system that captures user knowledge adequately. Knowledge-based recommenders are also not suitable to discover **niche** items for people with no ordinary taste. Another disadvantage is that selection in some domains rely less on **features** of an item. The users would have trouble to claim their preferences in these domains, e.g., for song features such as speed, kind of instruments and vocal.

Utility-based Filtering

Utility-based recommender systems are similar to knowledge-based recommenders. They may also be seen as a special kind of knowledge-based recommenders. The idea is derived from utility theory instead of knowledge-based systems. The knowledge source in the system is extended by **utility functions**. Utility functions represent knowledge to map user preferences to item features (functional knowledge). They can either be general for all users or user-specific. The main characteristic of a utility-based recommender is that it goes beyond the features of an item by interpreting them towards the **utility for a user**. This makes features comparable on an abstract level. MCDM methods (see Section 2.3.3) are used to predict user interests for an item. One of the first approaches applying MAUT in recommender systems is Guttmann's Tate-A-Tate [Gut98] recommender for products. It elicits preferences for item features manually and uses a weighted sum for prediction. The MAUT Machine of Schmitt et al. [SDB02] goes a step further and allows more sophisticated user preference elicitation with *OR* and *AND* operators. The approach of Price and Messinger [PM05] regards the utility of a whole set focusing on the diversity of the items and the uncertainty of user preferences. Manouselis and Costopoulou [MC07] combine MAUT and collaborative filtering in the domain of product recommendations.

Utility-based recommenders are applied in domains where the choice is made on **rational decisions**. The users should be able to formulate preferences, e.g., to prefer alternative A over B or indifference between the alternatives. Otherwise, the method shares its advantages and disadvantages with knowledge-based approaches.

Hybridization

Due to disadvantages of single recommendation approaches, researchers often combine several techniques to **hybrid recommenders**. Malone [MGT+87] early noticed that the combination is able to augment information filtering. For recommender systems, Burke [Bur02] defines several kinds of combining. The simplest way of combining several approaches is by **weighting** the predicted ratings of each method and combining them to a single rating, e.g., linear. This approach is quite static as weights are generally determined in advance. **Switching** between the outputs of several approaches allows to react on the quality of recommendations which a method is able to provide. For instance, if too few similar users are found, the method can switch to a content-based approach. This requires additional knowledge, e.g., about the quality of the output, to make a system-side decision. A **mixed** approach transfers the decision to the user by presenting the results of several approaches at the same time and let the user decide.

The predicted ratings are more like an explanation. The hybridization techniques so far work horizontally. **Cascading** of single recommenders works in a vertically way where the output of one recommender is the input of another. This requires setting up an order of application. The performance of an approach, its quality or coarseness may determine the order. In general, the set of items gets smaller after each step. **Feature augmentation** does not aim to shrink the set but to increase the quality of aspects taken into regard in the next step. For instance, a sparse rating matrix may be augmented with artificial ratings to make collaborative filtering applicable.

2.5.3. Context-Aware Recommender Systems (CARS)

Classical recommender systems such as collaborative filtering, content-based filtering or knowledge-based filtering aim to predict the rating, utility or preference for an item based on the attributes of an item or models describing user preferences for items. However, the consumption of items often depends on the context of the user. Individuals listen to different music while they are at home compared to being on the treadmill in the gym. They watch different movies with their partners compared to being alone. They are interested in visiting different restaurants depending on the weather. Adomavicius and Tuzhilin [ASST05] emphasize that context matters in recommender systems.

Calculating Recommendations with Context

In several publications concerning context-aware recommender systems (CARS), Adomavicius and Tuzhilin describe a general view of context incorporation into recommender systems ([AT08], [AT10] and [AMRT12]). The Equation 2.33 shows the traditional 2D view of predicting ratings R for an item I based on the user U. The extension with context C leads to a **multidimensional mapping function** in Equation 2.34.

$$I \times U \times C \rightarrow R \tag{2.34}$$

Context becomes a further dimension in the prediction. Independent from the acquisition of context in an explicit (by user input) or an implicit (with sensors) way, context is a set of explicit variables in this view. A challenge which all rating-based multidimensional methods share is that the users probably are unwilling to give ratings more than one time. This especially applies in case of different context. Relevant context for recommendation systems are the task of the user, physical or interaction context and also social context (Wörndl and Groh [WG07]).

Based on Equation 2.34, Adomavicius and Tuzhilin describe three algorithmic paradigms of incorporating context (Figure 2.7). Contextual prefiltering, postfiltering and modeling differ mainly in the order of context incorporation.

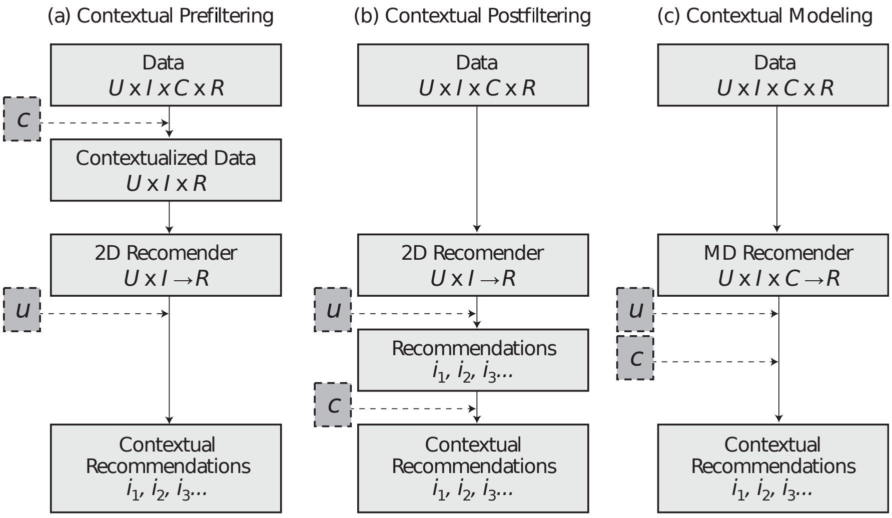


Figure 2.7.: Three paradigms of incorporating context into recommender systems according to Adomavicius et al. [AMK11].

Contextual Prefiltering

Contextual **prefiltering** (Figure 2.7 (a)) selects items based on context and evaluates only this subset with a classical recommender. Adomavicius et al. [AT05b] propose to use several prefilters and combine their output. The approach uses **loose coupling** of user preferences and context. Ratings are predicted with classical recommenders decoupled from the context. In prefiltering, items are cut off based on context constraints they have to fulfill, e.g., only movies on Saturday. We distinguish prefiltering by the type of the classical recommender. For rating-based approaches such as collaborative filtering or content-based filtering, the items which are in the subset are used to evaluate a new item, i.e., predict its rating. For approaches that do not rely on item ratings, the items in the subset are evaluated, e.g., movies which are only for kids.

In case of rating-based prefiltering, **exact prefiltering** can be too narrow leading to not enough ratings to make an accurate prediction. Therefore, Adomavicius et al. [AT05b] apply a **generalization** of context for prefiltering. Hierarchical generalization of context, e.g., "Saturday" → "Weekend", is used to increase the amount of prefiltered items. The result is a multidimensional matrix. The **reduction** of the matrix by context, e.g., cut off all items consumed during the week, aims to build a 2D matrix to be able to apply a classical recommender.

Baltrunas and Ricci [BR09] and Baltrunas and Amatriain [BA09] apply a different approach for rating-based methods by **splitting** items and user profiles. In item-splitting ([BR09]), virtual item groups are generated based on the context they are consumed in. In user-splitting ([BA09]), the user is represented by micro profiles for a specific context. After splitting, classical recommenders can be applied.

Wörndl et al. [WSW07] propose a **hybrid approach** to cope with the complexity of context. Content- and knowledge-based filtering is used to shrink the set of potentially relevant items and a 2D collaborative filter ranks the items.

Location is a powerful prefilter that is often used in research. Ahn et al. [AKH06] use the current location of the user to prefilter the search space for items in mobile advertisement (only items not closely located). Then, the authors apply a modified Pearson's correlation function which incorporates context to calculate the ratings for the items. For a restaurant recommender, Horozov et al. [HNV06] exploits the assumption that people who live in the same area are more likely to visit the same places. They calculate the similarity of the users after prefiltering them according to their home location. **Location-based services** (**LBS**) such as mobile guides filter out POIs which are close to the position of the user and apply different models of user interest prediction afterwards, e.g., CyberGuide [AAH+97], MapMobyRek [ARN08], CityGuide [DMM+04], GUIDE [CDM+00], COMPASS [vSPK04] or CRUMPET [PLM+01]). The approach of Kodama et al. [KIGI09] uses loose coupling of context and preferences by creating a prefiltered rank according to closeness. A Skyline filter based on the categories of the item features starts with the closest item (nearest neighbor query) and eliminates all other categories that are Pareto-dominated (see Equation 2.8) by this item. Dominance calculation is based on a user profile containing preferences for item categories. The method continues with the next neighbor item until all categories are eliminated or are in the Skyline set. The approach works fine for exactly one type of context information that needs to be ordered (in this case, the location). With more than one context information, some kind of ordering for this information is required.

Contextual Postfiltering

Contextual **postfiltering** (Figure 2.7 (b)) also uses **loose coupling**. A classical recommender evaluates the items and the output is adjusted to the context. Postfiltering either filters out items that are not relevant in context or adjust the ranking of items to context. Common methods to adjust the ranking use either specific item usage patterns in context or bring more diversity (McGinty and Smyth [MS03]) among the items into the ranking. Panniello et al. [PTG+09] investigate differences between postfiltering and prefiltering in the e-commerce domain. Their results suggest that the performance depends on the application itself.

We distinguish two general approaches of postfiltering. A **heuristic-based** postfiltering approach regards user preferences for attributes of items in context and analyzes the evaluated items. The heuristic-based context-aware ranking described by Agrawal et al. [ART06] uses an ordered preferences set where feature A is preferred to feature B in context C. Similarly, Stefanidis and Pitoura [SP08] also regard **contextual preferences** to calculate interest scores for items. Park et al. [PYC06] propose a context-aware music recommender with explicit user preferences for song attributes (genre, temp and song mood) depending on the physical mood. The preferences correspond to utility functions relative to the physical mood. A fuzzy Bayesian network infers on the physical mood. First, contextual information, e.g., temperature, noise, gender or age, is fuzzified with trapezoidal membership functions to cope with uncertainty and heterogeneous information. To avoid a loss of information in discretization for Bayesian networks, a fuzzy Bayesian inference method is used. Based on the posteriori probability for physical mood and the utility function, a final score is calculated for the items with a linear weighting model.

A **model-based** approach calculates the probability that a user would take an item with its attributes and in context. Yap et al. [YTP05] propose a model-based approach with a separation of the context module and the recommender system. The user gives a score to an item and the system learns user interests based on context information and item attributes. It mines relevant context parameters. In a later work, Yap et al. [YTP07] apply Bayesian networks in a restaurant recommender and show that it outperforms decision trees (similar to Palmisano et al. [PTG08]). Brunato and Battiti [BB03] describe PILGRIM, a mobile web site recommender that learns location-based usage patterns of Internet searching to adapt the ranking of search results.

Contextual Modeling

Contextual **modeling** (Figure 2.7 (c)) uses a multidimensional approach with **tight coupling** of preferences and context to calculate the ratings. A model based on user preferences in context is established and used for prediction. In rating-based modeling, the most obvious way to calculate a predicted rating is to extend the applied prediction function to multiple dimensions (Adomavicius and Tuzhilin [AT05a]). However, the $U \times I \times C$ matrix is probably too sparse to make accurate predictions. Like for prefiltering, Adomavicius et al. [AT05b] use **aggregation functions** based on context hierarchies to reduce the matrix to a classical 2D matrix. Palmisano et al. [PTG08] also exploit hierarchies but in a context hierarchy model instead of a multidimensional model. The authors found better prediction performance for customers of an online retailer if context is taken into regard. In contrast to Palmisano et al. [PTG08], Anand and Mobasher [AM07] make the assumption that the behavior of the user indeed depends on context but not all context information can be observed which is relevant. They propose a **state-based statistical process** to solve this problem.

In contextual modeling also well-known methods from other research areas are used. **Matrix factorization** is a more recent technique to predict ratings on multidimensional context models (Karatzoglou et al. [KABO10]). Oku et al. [ONMU06] propose a model with **support vector machines** (**SVM**) to calculate similarities between users. Their model incorporates additional context information and binary preferences of users for items (like or dislike).

Contextual modeling can also be applied using classical recommendation approaches. Yu [YZZ$^+$06] uses several recommenders in a **hybrid approach** (content-based, Bayesian classifier and rule-based) to incorporate context. Zimmermann [Zim03] applies context awareness to **case-based reasoning** by storing and retrieving cases along with contextual information.

2.5.4. Explanations in Recommender Systems

We already emphasized the usefulness of explanations for intelligent systems in Section 2.4.3. There, we focused on explanations for the reasoning process in Bayesian networks. Comparable to an intelligent system, a recommender system can also be regarded as a **black box** where the output is a set of items. The user may not understand why exactly these items have been chosen. Transparency could be especially useful for proactive recommenders because not a user query leads to the retrieval of items. *Amazon.com* already realized the usefulness of explanations, e.g., they show explanations like "Customers who bought this item also bought ..." along with the suggested items. Google+, the social network of *Google.com*, provides in the list of search results also information about contacts which liked the link as explanation. Besides explanations in classical recommender systems, context provides a rich knowledge source to extract helpful explanations. In contrast to intelligent systems, Lim et al. [LDA09] emphasize that context-aware systems are not necessarily built for expert users, i.e., users with rich domain knowledge like physicians. Therefore, explanations in intelligent systems focus on making complex correlations more transparent, whereas explanations in context-aware systems try to bring the system **closer to the user** and enhance **user experience**.

Argumentation Theory

The concept of explanations is a part of argumentation theory. For a detailed discussion, see common literature to argumentation theory such as Eemeren et al. [vEGH96]. An **explanation** is a set of arguments to describe a certain aspect, e.g., an item or a situation. An **argument** is a statement containing a piece of information related to the aspect that should be explained, e.g., "The gas station is inexpensive" or "Current gas level is low". In an explanation, arguments can be for (**positive**) or against (**negative**) an aspect or they can be neutral. The reason for an explanation is a **claim**, i.e., we want to claim something therefore we use arguments in an explanation. Providing the **certainty** of the system with the recommendations, e.g., as probabilities, can support claims. However, Cramer et al. [CER+08] found that this kind of information does not account for the trust in the system and are difficult to be interpreted by the user if presented as probability measure from 0% to 100%.

Carenini and Moore [CM06] describe **guidelines** for good arguments based on argumentation theory. Supporting and opposing evidence and their strength towards the claim should be evaluated. User preferences should be incorporated for this. Depending on the application scenario, the position of the main argument in an explanation is crucial (at the beginning or the end). The selection of opposing and supporting arguments depends on brevity of the explanation and the strength of the argument. Supporting arguments should be arranged at the first position of an explanation. If the explanation is long enough, some supporting arguments can be positioned at the end, too. Opposing arguments (counterarguments) can be left out, mentioned and refuted or only mentioned.

The **structure** of an explanation can either be a single argument, e.g., "the movie A is positively rated", comparative arguments, e.g., "the movie A is better rated than B" or comparative to a global value, e.g., "the movie A is better rated than the average". The **length** of an argument depends on the application domain. In product recommendations, the users prefer short arguments for low-priced products and detailed explanations for high-priced products like cars or apartments (Pu and Chen [PC06]).

Explanations in Recommender Systems

Tintarev and Masthoff [TM11] define **guidelines** to make explanations in recommender systems "good". They approach the problem from the evaluation side. First, the designers of explanations in recommender systems need to formulate the goal they want to achieve with explanations. Second, evaluating explanations is always affected by the underlying recommendation approach. Third, the presentation of explanations and recommendations also affect each other. Fourth, the style of the generated explanations should be related to the underlying recommendation approach.

Tintarev and Masthoff [TM11] describe seven generalizable **goals** for explanations in recommender systems. The goals are targeted to single item explanations. Which goals are accomplished by an explanation depends on the field of the application. **Transparency** explains how a system works, i.e., the selection of items. Sinha and Swearingen [SS02] found that the users feel much more comfortable with recommendations if they are made transparent. However, Cramer et al. [CER+08] found that it not necessarily increase trust. Related to transparency is the concept of **justifying** the behavior of the system. Herlocker et al. [HKR00] describe justification as transparency with the specific goal to explain why a recommendation or item was selected. Vig and Riedl [VSR08] emphasize that justification does not necessarily reveal the actual reasoning of the algorithm in contrast to transparency. The result of a black box system can be justified with a separate method without knowing how it works. **Effective** explanations aim to let the users evaluate items towards user preferences quickly in order to be able to accept or discard alternatives. Effectiveness is also one of the main goals of a recommender in general. Explanations that increase **efficiency** let the user make faster decisions, i.e., shorten the interaction time and reduce interaction steps. **Scrutable** explanations may contribute to this goal as well by giving the user the possibility to correct the system and to understand what is going on. In e-commerce, it is beneficial to **persuade** the user that the recommendations are useful and good. In contrast to justification, persuasion is in favor of the system, e.g., an e-retailer. Cosley et al. [CLA+03] and Cramer et al. [CER+08] show that user ratings can be manipulated towards the prediction of the system with different persuasive interfaces. Many researchers found that explanations are able to increase **satisfaction** with a recommender system (e.g., Sinha and Swearingen [SS02] or Herlocker et al. [HKR00]). Furthermore, some researchers were able to link explanations to an increase of **trust** in the system, e.g., Sinha and Swearingen [SS02]. Chen and Pu [CP07] show that the users are more willing to return to the system if they trust it.

Explanation Style

Tintarev and Masthoff [TM11] mention that the **style** of an explanation may or may not reflect the underlying recommendation algorithm. **Collaborative explanation style** either refers to the current item of the user such as "Customers who bought this item also bought . . ." or to the recommendation list by showing ratings from similar users (Herlocker et al. [HKR00]). **Content-based explanation style** takes additional properties of the item into account. Symeonidis et al. [SNM08] justify movie recommendations with the favorite actor of the user. Vig and Riedl [VSR08] measure the relevance of tags of items and preferences of users for tags to calculate recommendations and explanations. In a **case-based recommendation style**, the retrieved case is compared to the query or the current item (McSherry [McS05]). Finally, in a **knowledge- and utility-based explanation**, the knowledge base of user preferences and item attributes is used, e.g., "Camera A is more expansive than camera B" (McCarthy et al. [MRMS04]).

Generating Explanations

Depending on the recommendation approach and available knowledge, different **methods** for selecting arguments for an explanation are applied. However, we have to take restrictions of our environment into regard. Lim and Dey [LD11] emphasize that explanations should be **easy** and **quick** to understand and **compact** in mobile context-aware systems. Complex and long explanations like used for expert systems (e.g., in Amgoud and Prade [AP09]) are not applicable. Carenini and Moore [CM06] propose a strategy that allows generating simple explanations in our mobile environment. It consists of two components. **Content selection** selects and organizes the content of an explanation (deep generation). The authors use principles from argumentation theory to organize the arguments. A metric called compellingness measures the **strength of an argument**. The most compelling argument is selected first and then the system decides which arguments are further added if they are notably compelling. The output of deep generation is an abstract text plan that is converted in a human readable text by the content realization (**surface generation**). It selects terms and puts them in a correct grammatical structure.

As our recommender is utility-based and knowledge-based, we focus on generating arguments in a knowledge-based manner. Some knowledge-based approaches can be found in the area of **product recommendations**. These explanations justify items that should make the users to trust the system and persuade them to buy. Pu and Chen [PC06] group items into **categories** with knowledge-based titles as explanation. The method is applicable if recommendations contain large sets of items. Felfernig et al. [FB08] use a **utility-based approach** with multiple criteria to persuade the users. Stored text phrases represent explanations e_a with one argument. We call these explanations in our model arguments. Additionally, the contribution of an explanation to predefined item features i is stored as function $con(e_a, i)$. The utility of an argument $u_e(e_a)$ is calculated with user preferences as weights for an item feature w_i and the argument contributions: $u_e(e_a) = \sum_{i=1}^{n} con(e_a, i)w_i$. Furthermore, the position p of an argument in an explanation has a utility $u_p(p)$. The utility u_o of an ordered set of arguments is calculated as the utility over all arguments in the set for all possible permutations: $u_o(a_1, \ldots, n) = \sum_{p=1}^{n} u_e(e_p)u_p(p)$. Dependencies between arguments are taken into regard as constraints, e.g., argument e_i should always be before e_j. The task of ordering arguments is seen as **constraint satisfaction problem** (**CSP**). Zanker and Ninaus [ZN10] argue that complex recommendation approaches are difficult to track for extracting arguments. Like Felfernig et al. [FB08], they propose a justification method decoupled from the recommender. Explanations are represented as a path in a **directed acyclic graph** (**DAG**) with nodes as arguments. This leads to long and complex explanations. Arguments are constructed based on domain knowledge, user preferences and context information. Fahri [Fah08] proposes a theoretical model for explanations based on Toulmin's [Tou03] model of argumentation which also generates complex explanations in a knowledge-based manner.

For other domains such as POIs or movies, the main goal of the available approaches is providing transparency. Explanations allow the user to make more efficient and effective decisions. This is especially desirable for **mobile devices** where interaction is difficult. Baltrunas et al. [BLPR11] propose a method that is based on context-based ratings for POIs in a mobile scenario. The authors estimate the rating of an item based on the context, the rating of other users and a learned model which represents the connection of contextual information and items. An explanation consists of **one argument** that has the largest positive effect on the calculation of the rating. This is simply the *argmax* in the learned model. The authors do not distinguish between positive or negative arguments. If an item was not recommended, the argument shows the reason why. They argue that only one argument is used to impose no overload to the user and to make it easy to grasp. Instead of one simple argument, the approach of DeCarolis et al. [dCNPG09] shows comparative descriptions of POIs about a whole map area for tourists. The system selects items on the currently viewed map area by user preferences, item features and context. Based on the items shown to the users, a description is generated. It summarizes the map area by either showing individual or comparative arguments. Finally, a human readable text is generated. The explanation is **comprehensive** but too detailed to be presented automatically in an automotive scenario.

Besides explanation generation based on heuristics, **learned knowledge** can also be used. Similar to Baltrunas et al. [BLPR11], the system of Symeonidis et al. [SNM08] learns a user profile between users and features of items that were rated. Weighted features describe the users. The application domains are movies and news. The authors build clusters of users to extract justifications from groups instead of individuals. Explanations are feature-based (comparable to knowledge-based explanations without utility), e.g., "Item 1 was recommended because it has features X,Y,Z". Vig and Riedl [VSR08] use item tags instead of item features for their explanations in a similar way.

2.5.5. Evaluation Methods

Evaluation Approach

We discussed different approaches of recommender systems which are applied in different domains. To measure which approach works better in a certain domain, evaluation metrics are needed. Commonly established metrics also support the comparability of approaches. Herlocker et al. [HKTR04] emphasize that this is not trivial because the used data set, the domain and the desired output have an influence. Hence, there is not a best approach for all applications. Additionally, the goals of an evaluation can be different comparable to the application of explanations. The goals are reflected in the way the evaluation is carried out. The evaluation can be either done **offline** with existing data sets or **online** embedded in the environment of the application. Online tests are more expressive, but are also more costly and subject to environmental noise. Terveen and McDonald [TM05] recommend using scenario-based evaluation which is built on the task a recommender system.

Quantitative Metrics

The main focus in recommender systems research is to measure the quality of an approach. **Accuracy** is the metric that is used most frequently to measure quality. The ability of a recommender system to predict the rating of a user for an item accurately is called **prediction accuracy**.

$$MAE = \frac{\sum_{i=1}^{N} |p_i - r_i|}{N} \tag{2.35}$$

Mean absolute error (**MAE**) in Equation 2.35 is the most common measure for prediction accuracy. The difference between the true user rating r_i and the predicted rating p_i is summed up for all predictions N. Other variants emphasize large errors (**mean square error** (**MSE**)) or are normalized over the range of ratings (normalized MRE). MAE is meaningful if ratings are shown to the user in the recommendation. For Carenini [CM06], MAE misses a crucial point of recommender systems. Recommender systems are less a machine learning application that has to predict the correct rating, but a decision support tool to find interesting and relevant items.

Another group of quality metrics comprise **classification accuracy**. It describes how well relevant items are separated from irrelevant items. Items can be separated in four disjunctive groups (Table 2.1) based on the prediction for the items (observation) and how it actually should be (expectation).

	actual items (observation)	
predicted items (expectation)	true positives (tp)	false positives (fp)
	false negatives (fn)	true negatives (tn)

Table 2.1.: Classification results

An terms "true" or "false" follow the classification result and "positive" or "negative" refers to the prediction. The groups can be applied to the items that are shown or not shown to the users. Correctly classified items are either relevant and shown to the user (tp) or irrelevant and not shown to the user (tn). Incorrectly classified items are either relevant and not shown to the user (fn) or irrelevant and shown to the user (fp). In machine learning, the grouping corresponds to a so-called **confusion matrix**. Actual accuracy metrics are defined according to these groups. The **accuracy** measures the rate between correctly classified and all available instances n in the data set (Equation 2.36).

$$accuracy = \frac{tp + tn}{n} \tag{2.36}$$

Accuracy corresponds to the probability that a recommendation is relevant. It describes the general performance of the algorithm. In recommender systems research, metrics that focus on the accuracy of recommended items are more interesting. The **precision** is the amount of correctly classified items among all shown items (Equation 2.37).

$$precision = \frac{tp}{tp + fp} \tag{2.37}$$

The **recall** states how many correct items have been found among all available correct items (Equation 2.38).

$$recall = \frac{tp}{tp + fn} \tag{2.38}$$

For our application domain, Rhodes [Rho00] made an important observation. He found that precision is more important in proactive information retrieval than recall. Precision and recall are often considered together for evaluation. They are inversely related.

Increasing the number of items classified as relevant generally leads to higher recall but lower precision. Showing fewer items to the users decreases the probability to recommend irrelevant items but also decreases recall. Precision and recall can therefore be combined. **F-measure** is one example of combination (Equation 2.39).

$$f - measure = 2 \times \frac{precision \times recall}{precision + recall} \tag{2.39}$$

Metrics which group items depend on how the items are separated in "positive" and "negative" items, i.e., which threshold is selected for the predicted rating. This determines the notion of **relevance** in the system.

Another kind of metric is the **receiver operating characteristic** (**ROC**) curve. The ROC curve is the plot of a function of the true positive rate x against the false positive rate y. The true positive rate (**sensitivity** or hit rate) is calculated equally to the recall. The false positive rate (complement of **specificity** or miss rate) states how many incorrect instances are in the amount of all incorrect instances for a class c (Equation 2.40).

$$false\,positive\,rate = 1 - specificity = 1 - \frac{tn}{tn + fp} = \frac{fp}{fp + tn} \tag{2.40}$$

The ROC curve can be interpreted with the **area under the ROC curve** (**AUC**). The values of the AUC are in $[0, 1]$. The straight line $y = x$ separates good ROC results from poor. Therefore we interpret AUC values in $[0.9, 1.0]$ as "excellent", $(0.8, 0.9]$ as "good", $(0.7, 0.8]$ as "fair", $(0.6, 0.7]$ as "poor" and $[0.5, 0.6]$ as "failure" (according to Araujo et al. [APTE05]).

Cohen's κ is a metric that also allows an interpretation. It measures the agreement between observed rates and chance. Values close to 1 express full agreement to the model and values equal to or lower than 0 express no agreement, i.e., the model performs like predicting the output with a random number or worse. Landis and Koch [LK77] give an interpretation: < 0.0 as "poor", $(0.0, 0.2]$ as "slight", $(0.2, 0.4]$ as "fair", $(0.4, 0.6]$ as "moderate", $(0.6, 0.8]$ as "substantial" and $(0.8, 1.0]$ as "excellent".

Besides prediction and classification accuracy, **ranking accuracy** metrics describe the quality of a ranked list of items. Basically, the predicted ranking of a system is compared to the ranking of a user. Common metrics in this group are correlation-based approaches that measure the similarity of lists of items. Another approach by Breese et al. [BHK98] is the so called "half-life utility metric" which assumes that items later in a list are generally perceived as less relevant because the user does not browse the list deeply.

Qualitative Metrics

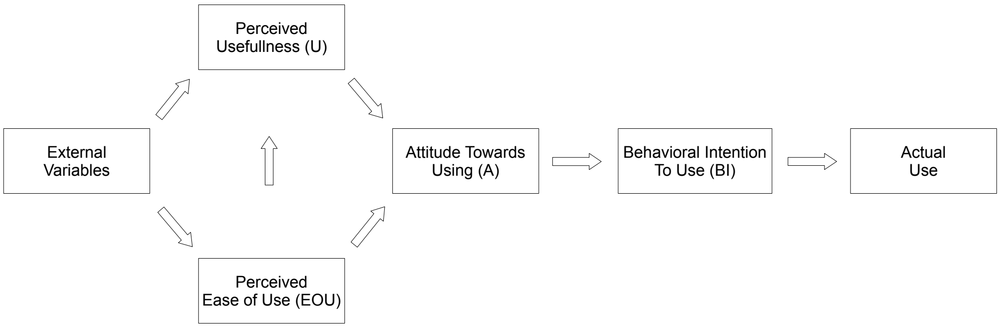


Figure 2.8.: Technology acceptance model (TAM) by Davis [Dav89]

Besides objective measurements for accuracy, we are also interested in subjective measurements for **user acceptance** of recommender systems. User acceptance is measured in several studies of information systems with a well-known model called **technology acceptance model** (**TAM**) proposed by Davis [Dav89]. It was originally proposed for desktop information systems. Figure 2.8 shows the main components of the model. The two main factors for user acceptance are:

- **Perceived Usefulness (U)**: "the degree to which a person believes that using a particular system would enhance his or her job performance" [Dav89]
- **Perceived Ease of Use (EOU)**: "the degree to which a person believes that using a particular system would be free from effort" [Dav89]

The perception of both factors is indirectly made through application-specific external variables such as the design of the system. These factors influence the **attitude** (**A**) of a user to use the system which finally determines if a system is actually used. A final statement about the actual use can be made if there is **behavioral intention** (**BI**) to use it, i.e., the user shows interest to use the system further during usage. An information system that is easy to use increases the perceived usefulness as well. Two systems with the same features may be perceived differently towards usefulness if one is more easy to use. Although they are not always easy to use, expert systems are used because the users have to. Particularly, the usefulness of a system can already be measured in an early stage of prototyping (Davis and Venkatesh [DV04]). Davis and Venkatesh argue that it has more influence on user acceptance than ease of use. However, ease of use should be considered in a study with a prototype. Otherwise it can have negative influence.

Many extensions were proposed to TAM to tailor the model to other application domains than desktop information systems. Kaasinen [Kaa05] derives for **mobile services** that trust in the system and how much effort the installation causes are additionally important in that domain. Kaasinen finds factors such as service features, user preferences, comprehensibility and seamlessness as external variables for usefulness. A clear overview of the content, no information overload, a fluent navigation, smooth interaction and low effort user preference setup to provide useful information right from the beginning are external factors for ease of use.

Regarding aspects of TAM in the context of recommender systems, classical metrics such as accuracy and relevance are not enough to measure **usefulness** (Rhodes [Rho00] and Budzik et al. [BHBK00]). This is particularly the case for proactive information systems. The most relevant restaurant can fail usefulness if the user is not hungry. User satisfaction with the system also depends on context and the situations of the user in this case (Beer et al. [BFH$^+$07]). However, because TAM is flexible enough, Jones and Pu [JP08] use the model for a survey of music recommendations to investigate which of the systems is more accepted. Usefulness corresponds to the quality of a recommended item. Quality describes whether the item fits the context, the mood and the profile of the user. Ease of use is measured by effort. Effort describes how much involvement it needs to get recommendations, how complex interaction with the systems is in general and the usability of the system. In another survey, Cramer et al. [CER$^+$08] investigate how transparency influences user acceptance in content-based recommender systems. The authors do not use an explicit model such as TAM. The results show that transparency lead to a higher acceptance of the recommendations, not necessarily to the system. In automotive research, the work of Meschtscherjakov et al. [MWST09] uses TAM to investigate user acceptance of different persuasive interfaces for supporting fuel-efficient driving. The authors perform an online questionnaire in a pre-prototype phase.

2.6. Proactive Assistance

We described Bayesian networks and fuzzy logic as methods for intelligent behavior and recommender systems for filtering useful items for the driver. The idea of proactivity uses these methods in the overall system design. Proactive computer systems are mainly investigated in the area of **artificial intelligence** (**AI**). The application domain is often office assistance (e.g., Horvitz et al. [Hor93], Myers et al. [MYs07], Hui and Boutilier [HB06], Natarajan et al. [NJTF07], Seifert et al. [SBB04] or Cohen et al. [CCM04]). Other application areas are assistance for elderly people (e.g., Haigh et al. [HKM$^+$04] and Boger et al. [BPH$^+$05]), task assistance (e.g., Suzuki et al. [SMK$^+$06]), shopping assistance (e.g., Bohnenberger et al. [BJJ05]) or route planning assistance (e.g., Mukai and Watanabe [MW05]). Billsus et al. [BBE$^+$02] emphasize that intelligent "agents that select information for the users are convenient on the desktop but essential in mobile use". In this section, we introduce the idea of proactivity and summarize the most important characteristics for the design of a proactive assistance system. Furthermore, we regard proactive information systems as special type of proactive assistance and distinguish different levels of proactive behavior.

2.6.1. Proactivity

Definition

An early use of the term ”proactive” was in the book ”Man’s Search for Meanding” by Frankl [Fra46]. People are described as proactive if they take responsibility for their lives by choosing their response based on experiences. Covey [Cov89] takes this property to make ”being proactive” as one of seven habits a highly effective person needs to have. Covey describes proactivity as the **freedom** to choose the response to a stimulus. In contrast, a **reactive** person in Covey’s definition is someone who is dependent on environmental circumstances and their success. Before Covey, Martin [Mar83] brought the term into economy with his decision making framework. The framework is based on formulating **goals** and **assessing alternatives** before taking action. Martin emphasizes that proactivity is the choice someone has.

Proactive Systems

Assisting people proactively is more difficult for computers than for people. In general, people know their tasks and how to get information to accomplish that task. Computers in contrast can only infer on users’ intention to be able to assist proactively. On the other hand, computers offer large storage and powerful processing capabilities. With the fast advance of mobile computers that are available in the daily lives of the users and the problem to interact with all of these networked computers, **proactive computing** was declared as solution to cope with this overflow by Tennehouse [Ten00]. He predicted a movement from human-centered (interactive) to **human-supervised** or unsupervised (proactive) computing. Computers sense the world and they are connected to monitor and shape their environment. They are also able to respond to stimuli faster than people. This makes people to move from in the loop to **above the loop**. Hence, the vision of **ubiquitous and mobile computing** leads to a new paradigm of human machine interaction.

A major property of a proactive or automatic system is to be able to **predict the future** to **anticipate user needs** (Want et al. [WPT02]). The delimitation to a reactive system is important at this point. Reactive decisions are well prepared actions to signals. These actions can be performed very quickly. Proactive decisions involve a process of comparing alternatives. For Duff et al. [DHT06] ”a proactive agent is required to actively work towards achieving its goals, while a reactive agent is required to adapt to environmental changes in order to maximize its ability to achieve its goals”. Salovaara and Oulasvirta [SO04] call a system proactive if it is working for or on **behalf of the user**.

2.6.2. Proactive Assistance

Origin

The idea of software providing assistance by reducing work and information overflow is influenced by work of Maes [Mae94]. The principle idea is that a piece of software often called **agent** knows habits, preferences and interests of the user who asks for **assistance**. The assistant has the capability to perform tasks that are **delegated** to it. These agents hide complexity from the user. Maes took the metaphor of autonomous agents from the field of artificial intelligence and adapts it to intelligent user interfaces. An intelligent agent is between the application and the user and assists the user by **making suggestions**, e.g., an email agent suggests an action with an email. At the same time, Horvitz worked on a project called Lumiere [Hor93]. The goal was to improve human machine interaction by means of intelligent software. However, the resulting commercial system Clippy was a failure and not accepted by the users. Based on the experience with Clippy, the research group proposed the concept of **mixed-initiative user interfaces** (Horvitz [Hor99]). It is a mixture of being able to **directly manipulating** the interface (proposed by Shneiderman [Shn93]) and **intelligent reasoning**.

Non-functional Requirements

Based on Horvitz's research on mixed-initiative user interfaces and Iba's [Iba07] criteria for what helpful assistance is, Myers et al. [MYs07] define **non-functional requirements** on proactive behavior of an agent as nine guidelines. Overall, a proactive agent has to add **value** to the task of the users or their interests. To add value, the agent should be **pertinent** and pay attention to the current situation the user is in. The **anticipation** of short-term as well as long term needs of the user is another factor. This involves continuous learning to become better. To assist a user, the agent should also have a **competence** in performing an action or gathering and processing knowledge. However, Iba [Iba07] emphasizes that competence only is not enough to classify assistance as helpful, although it is the central feature of an agent. Furthermore, if the agent executes an action or starts communication with the user, then it should be in an **unobtrusive** way, i.e., with as little as possible interference to the current task of the users and their attention. Performed actions should also be **transparent** and comprehensible to the user. The system should be **controllable** by allowing the user to terminate, invoke or ignore the assistant and refining its results. Controllability reduces the annoyance of poor system decisions (e.g., in Horvitz et al. [HJH99] or Haigh et al. [HKM+04]). This includes that the agent works in a **safe** way by minimizing negative consequences, e.g., distracting the user or costly backtracking. Finally, the agent should be **deferent**. On the one hand, it should be unimposing and perform actions at the right time. On the other hand, the users should notice that there is an assisting agent. Horvitz [HJH99] emphasizes that the underlying **uncertainty** is involved in nearly all properties.

Functional Requirements

Want et al. [WPT02] set up **functional requirements** on a proactive computer system as principles for system design. The first principle describes the **connection to the physical world**. This comprises to gather knowledge about the environment the system is in. Such systems also need to be scalable to upcoming unknown information sources. Another principle is to **generate real-time responses** to physical conditions. Otherwise, relevant situations for acting proactively are missed by the system. **Predicting the future** is a further principle. It comprises the processing of observed contextual data on a higher level, statistical reasoning about available information and prefetching and caching of data that is potentially needed in the future.

Models of Proactive Assistance

In research, several models of intelligent systems are used to realize proactive assistance. Horvitz [HJH99] apply **Bayesian networks** to cope with uncertainty and to infer user actions. Fern et al. [FNJT07] give a definition of an assistant with a **partially observable Markov decision process** (**POMDP**). It allows deciding not only on the current situation but also on future situations with statistical models of prediction. Anagnostopoulos et al. [ANH07] describe a situation awareness model as part of proactive behavior for imprecise reasoning based on **fuzzy sets**. Myers et al. [MYs07] use **belief desire intention** (**BDI**) agents to be able to model deliberate behavior of the system. The goal of all models is to infer on the actions of a user to provide the right assistance.

Interruption

A major characteristic of a proactive system is that it interrupts the user to consolidate results, to request input or to present results. Psychology research shows that interruptions generally impose **uncontrollable and unpredictable stress** (Cohen [Coh80]). It occurs without control of the user. In case of human interruption, it "requires immediate attention" and "insists on action" (Covey [Cov89]). Interruptions may lead to information overload by either imposing **time pressure** or causing **cognitive effort** at the recovery to the initial task. The more interruptions, the more a (complex) primary task performance is decreased (Speier et al. [SVV99]). McFarlane and Latorella [McF02] propose four different methods of **coordinating interruptions**. An **immediate interruption** requires handling the interruption immediately without the ability to postpone. Another way is to let the user **negotiate the interruption** in a dialog. The negotiation can also be delegated to a **mediator** system. To make the interruption less to a surprise and predictable, it can also be **scheduled** with predefined timing. Baron [Bar86] suggests making the timing of an interruption context-sensitive as a critical success factor for intelligent computer systems. McFarlane and Latorella [McF02] also describe three phases of interruption. In the **before switch phase**, the user should be best possibly introduced to the upcoming interruption. **During the switch** the performance of primary and interruption task should be maximized and **after the switch** it should be easy for the users to return to their primary task.

2.6.3. Proactive Information Delivery

General Characteristics

Proactive assistance systems involve actions that are performed by the system automatically. In case of recommender systems, actions are limited to the retrieval and presentation of items to the user. Rhodes [Rho00] proposes the delivery of information **at the right time** (**just-in-time information retrieval** (**JITIR**)). He emphasizes the advantage compared to an interactive system. The user does not have to put in a **query** into the system. Maybe the user does not even have a query in mind. Jones [Jon05] states as a benefit of proactive delivery that information that the user is **unaware** of can be found by the system.

A major characteristic of a JITIR is that it has to be **unobtrusive but accessible** (Rhodes [Rho00]). For instance, a ringing phone may be obtrusive. However, turning the ring tone off entails the risk that the user misses a call. Similarly, Billsus et al. [BHMA05] emphasize that the interface of a proactive information system should **neither be too subtle nor too intrusive**. Information should be noticeable to make a decision and should be easy to access. Another aspect of accessibility and obtrusiveness is to have the possibility to **ignore** the delivery and retrieve it later again. To enable proactive behavior, a JITIR assistant should be aware of local **context**.

Furthermore, Rhodes [Rho00] describes two major functions of JITIR. First, it should **reduce the costs** to get information by acting proactively. Second, it should be easy to find out for the user **what information is available** and to retrieve more information if necessary. Billsus et al. [BHMA05] emphasize that the users ignore recommendations because the interface does not **provide enough information** so that they can assess item relevance quickly. Information is either relevant to the current task, not relevant to the current task but leads to discovery of relevant information or is valuable of other reasons. Billsus et al. [BHMA05] state that the users are generally focused on their task unwilling to interrupt it for a recommendation. Puerta Melguizo et al. [PBD+07] add that the system should have a deep understanding of the task of the user to recommend information.

Rhodes [Rho00] also makes three assumptions about the information provided by a JITIR. First, delivered information is **not always useful** the same way because not all contextual information can be incorporated. Second, **the users determine themselves** the usefulness of information in case of sufficient information. Third, the distraction by making decisions caused by **cognitive load** should be minimized.

Characteristics on Mobile Devices

So far, we regarded the characteristics of a proactive information system either in general or for desktop systems. Jones and Brown [JB04] emphasize that in contrast to a desktop and especially an interactive system, the users in a mobile scenario are not able to fully **concentrate their attention** to the system. The users are rather engaged in many other tasks. Each delivery that catches users' attention is an intrusion. As a consequence only relevant items that cannot be filtered by the system itself should be delivered. Especially in an automotive environment, the major task of driving cannot be interrupted to focus fully on the recommendation. The driver always ends up in a multitasking situation. Jones [Jon05] proposes that a proactive mobile information system needs to (1) detect the activity of a user, (2) decide if and when to deliver information, (3) decide how much information to deliver and (4) choose the right interface for delivery. While on the go, the **physical environment** determines the need of information and needs to be captured by sensors. Furthermore, in mobile scenarios, the communication with the user is **limited** by small screens and restricted possibilities for interaction, e.g., for input of data. Therefore, van der Heijden et al. [vdHKK05] suggest making the recommender as **simple** as possible in this case. This results in a trade-off between simple accessibility and recommendation accuracy. Yeung and Yang [YY10] denote reduction of interaction with less data input and browsing as one of the main goals of proactive mobile recommender systems. As people are adaptive decision makers (Payne et al. [PBJ93]), van der Heijden et al. [vdHKK05] suggest **supporting human decision strategies** (see Section 2.3.2) by the system.

Mobile proactive recommender systems do not necessarily suggest information for the current task such as supporting visitors in a museum (Bohnert et al. [BSZ09]) or to find relevant individuals and buildings on a campus (Bouzeghoub et al. [BDW09]). They are also able to provide **task unrelated information** such as news (Young and Yang [YY10]).

2.6.4. Classification of Proactive Assistance

Separated Responsibility

In research, several kinds of proactive assistance are distinguished. One way comes from delegating tasks to a proactive system. This leads to a **separated responsibility**. Myers et al. [MYs07] distinguish between the tasks that are performed only by the user, agent tasks and shared tasks that are performed by the users and the agents together. Shared tasks are performed on behalf of an interface (mixed-initiative user interfaces). Depending on the degree of proactivity, a system decides when to perform a delegated task and when to communicate with the user, what actions to perform to fulfill the task, how to perform the actions and how to learn to perform the actions better.

Degree of Proactivity

	Degree				
Proactive Behavior	Do it yourself	Tells you to pay attention	Tells you what to pay attention to	Make suggestions	Make decisions
Example: alarm clock	Set your alarm	*„Check your alarm settings?"*	*„Looks like rain tomorrow. Check your alarm settings?"*	*„You might want to change your alarm settings because it look like rain and when it rains it takes you 20 minutes longer to get in."*	*„I've set your alarm to 6 am [because ...]."*
Feasibility		Not impossible	Harder	Much harder	Extremely hard
Cost of failure		Low	Moderate	Higher	Very high
Interface		Non-intrusive	Can be non-intrusive	More intrusive	Very intrusive
Required information		Little	More	More (personal)	A lot of, mostly personal

Figure 2.9.: Degree of proactivity (according to Isbell and Pierce [IP05])

Isbell and Pierce [IP05] describe the **degree of proactivity** with an **interface-proactivity** (**IP**) continuum (Figure 2.9). The IP continuum details a mixed-initiative by providing a continuous view from direct manipulation only by the user (left most side) to a completely autonomous agent (right most side). All degrees in between correspond to a shared task processing. On the second degree from left to right, the agent draws the attention of the user to the task, e.g., to set an alarm clock. The next degree additionally provides information necessary for the task, e.g., it is raining tomorrow. At degree 4, the agent makes a suggestion, e.g., set your alarm clock 20 minutes earlier because of rain tomorrow. All degrees from 1 to 4 reduce the cognitive effort of the users to complete the task but do not do it automatically. Only the last degree fully moves task completion from the user to the agent with feedback about what's being done. Degrees 2 to 4 correspond to the definition of Lieberman and Selker [LS03] of an **advisor** that suggests actions. The last degree is an **assistant** that performs tasks. The more proactive a system is, the harder it is to realize. The more information it needs, the higher is the cost of failure and the more intrusive it is. Proactive recommender systems are advisors but cannot be matched exactly to one of the degrees in the continuum. Although they make suggestions, they do not suggest actions to perform but items. Therefore, they are between degree 3 and 4.

Isbell and Pierce [IP05] also describe design guidelines to decide for the degree of proactivity. The first is the tradeoff between the **benefit** that the system offers to the user when it works correctly and the **costs** if it fails. Already a small benefit and a high cost of failure can make an agent helpful, if the likelihood of success is high. If the user does not expect the system to fail, it will use it regardless of failure costs. Second, the **frequency of usage** is crucial. Failure will be more accepted if the system is used more seldom. The **attention** or the cognitive effort of the user also varies on the continuum. In general, it is low if the agent is more proactive. In case of failures, the attention may dramatically increase depending on whether the user realizes mistakes quickly and how much effort is needed to fix it. Therefore, the quality and the amount of **feedback** needs to be better, the more proactivity is applied. The more proactivity, the more amounts of **high quality data** are needed. This refers both to the access to data, e.g., user's time schedule, and to the feedback from the user. Feedback can be acquired in an implicit or an explicit way. Implicit acquisition reduces the amount of attention but is more error-prone. The need to gather information also increases with the degree of proactivity. In this context, the costs for resources increase with more proactivity, e.g., computational or storage costs.

Focus of Proactive Assistance

Yorke-Smith et al. [YSSMM09] describe the **focus** of an assistance system to classify proactivity in application-, task-, and utility-focused proactivity. **Application-focused proactivity** provides assistance for a specific application, e.g., Clippy in Microsoft Office. **Task-focused proactivity** is based on the current or a future task that a user has to perform. **Utility-focused proactivity** generally helps with tasks. For instance, if the agent derives high cognitive load of the users, it offers the users to take a task.

Outcome

Horvitz et al. [HBH$^+$98] propose different **control schema** for proactive assistance. A **pulsed** strategy starts the assistance based on a time interval. A **deferred** strategy also uses a time interval but only in idle times. A **triggered** strategy starts the system after an event occurred and a **mixed** strategy operates in a time interval but only when an event occurred in that interval.

In case of proactive information systems, we distinguish two kinds of **output**. **Notification-based** approaches have the goal to deliver their results as a notification in a way that the user immediately notices them. This can be by email, a pop-up or supported by audio or haptic feedback. The system forces its delivery on the user as it is noticed immediately. In a **retrieval-based** approach, the results are only retrieved and shown in a dedicated area, e.g., a widget. The users have to look up the results by themselves. They decide when a recommendation is relevant. In case the users do not think about to look up, they probably miss the recommendation. Note, the meaning of retrieval-based is different to pull-based where the users are required to put in a search query. Which approach to take for output, depends on the urgency of information and the interface design. Gallego-Vico et al. [GVWB11] call these two approaches notification- and widget-based for smart phones.

2.7. Summary

Driver assistance systems are distinguished by their contribution to tasks inside a car. Some are for security reasons. Others are for convenience or energy efficiency. The systems differ in the level of automation, i.e., how much of the task they adapt from the driver. Preview and prediction are important skills for a driver to anticipate future situations and react appropriately to them. Driver information systems are either information systems for assistance systems (e.g., to deliver warnings) or for infotainment, e.g., a POI search. The navigation system is the most popular information system. There are several visual output channels for information, e.g., the CID or the instrument cluster. Additional channels are audio or haptic. Input is limited to controllers like the iDrive controller or speech. Information overload is a serious problem because it may cause driver distraction. Driver distraction occurs if the same human resource is occupied by other tasks than driving and there are not enough resources left to drive properly.

The knowledge of context and situations of a user contributes to make assistance for the drivers more intelligent. Many different definitions for context exist. For automotive scenarios, user, car and environment context are the most important. Context modeling distinguishes between low-level, e.g., raw sensor data, and high-level context, e.g., derived context. Situations are context information with special characteristics. They have a temporal range and may be in a relation, e.g., situation A occurs before situation B. Situation awareness comprises situation recognition, comprehension and projection into the future. Human decisions are based on situation awareness.

In general, a recommender system involves a final choice by a user. Providing choice is important if the users have to select between alternatives but to many choices may decrease decision quality. Furthermore, people want to justify their choice. People apply decision making strategies according the accuracy-effort principle. They balance the effort for decision making with the accuracy of the result. This often leads to non-comprehensible methods with heuristics. There are many comprehensive as well as non-comprehensive strategies. Besides human strategies, decision making with multiple criteria is mathematically defined in multi-criteria decision making (MCDM). MCDM can be distinguished in MAUT, Outranking or dominance-based methods. MAUT methods are the most popular and range from simple linear models to complex pairwise comparison based methods like AHP.

To enable intelligent proactive behavior of a system, methods are needed to represent operational knowledge. There are plenty of methods for intelligent systems. Based on literature review, we picked Bayesian networks and fuzzy logic for our investigation. As intelligent systems are black boxes, comprehensibility of the system by the user is an important aspect. Explanations are one possible solution to make the system comprehensible. For a proactive recommender system, explanation methods for abductive reasoning and the reasoning path are most valuable. Feature selection methods allow assessing the impact of a situation or context to the final decision for or against recommending.

Recommender systems are a special kind of intelligent systems in order to filter items. They bring the users U, items of interest I and user preferences, e.g., as ratings R, together to predict unknown user preferences for new items. Depending on the application domain, they follow different purposes, e.g., to find some good items or to annotate the context with ratings. In literature, they are often distinguished by their recommendation technique. Content-based filtering matches features of items to estimate unknown user preferences. Collaborative filtering tries to find similar items or users based on their preferences. As both methods suffer from cold start problems until enough user preferences are known, they are often combined with knowledge-based filtering. Knowledge-based filtering uses explicit user models, e.g., utility functions, to predict unknown user preferences for an item. In recent years, context C is added as additional parameter for user preference prediction. Context integration can be done with prefiltering items by context, postfiltering by context or by establishing explicit models with context. To make recommendations comprehensible, explicit explanations, e.g., "movie A is more romantic than movie B", are used. Their application depends on the intention of the explanation, e.g., to justify items or to persuade the user to buy. The style of an explanation depends on the underlying recommendation method. To evaluate recommendation approaches, the most common metrics are accuracy, precision and recall. However, they do not tell anything about user acceptance of a system. To evaluate user acceptance, the technology acceptance model (TAM) can be used.

Proactivity describes the freedom of a system to decide for an action based on an input. In contrast to that, reactive behavior defines predefined actions for a specific input. A proactive system works out tasks on its own and may contact the user for an input in a mixed initiative user interface. Helpful proactive assistants share common properties such as unobtrusiveness, competence, transparency or controllability. As they make decisions on their own, assistants have to interrupt the users in their current task from time to time. To lower the negative effects of interruptions, a proactive information system should be unobtrusive as well as accessible for the user. The degree of proactivity ranges from not providing assistance (degree 1) to fully performing a task without user input (degree 5). Proactive recommender systems are between degree 3 and 4 because they make suggestions of items instead of actions. We distinguish notification-based proactivity with an explicit pop-up and retrieval-based proactivity with the display of information in a dedicated area.

3. Related Work

Based on findings in automotive information systems, recommender systems and proactive systems research from the previous chapter, we discuss related approaches of proactive recommender systems in this chapter. The discussion is not limited to proactive systems with classical recommender systems because only a few approaches are available in this area. We regard approaches that decide for taking action automatically and deliver information to the user. This includes information retrieval. The discussion is limited to those methods that deliver information instead of working out other kinds of tasks proactively. All reviewed approaches have some kind of inference system to decide when a user needs information, a filtering component that selects items to deliver and a user interface in order to make the delivery. The chapter is structured according to the application scenario of the systems. They are either designed for desktop computers, mobile devices or in-car usage. The sections are further divided by the triggers that lead to the delivery of information. We distinguish triggers with environmental context information, user preferences and the task of the user. To decide to which section an approach belongs, one of these information sources should be mainly responsible for the delivery and the others may also be used for item assessment.

3.1. Proactive Desktop Recommender Systems

Desktop systems are laptops or desktop computers. The user works with these systems to fulfill a specific task. Therefore, most of the systems are deployed in an office context, e.g., to-do lists, email and documents. The main difference to a mobile setup is that environmental context is less important to the behavior of the system, especially the location. Besides support for office tasks, activities outside the context of working are supported as well. This comprises searching for information, media (e.g., books and news), e-commerce and web browsing.

3.1.1. Task-based

Document Retrieval

The goal of proactive office assistance is to enhance efficiency in task completion by fast access to relevant information. XLibris by Price et al. [PGS98] aims to deliver additional documents while the user works on a **document**. Rhodes' [Rho00] approach is similar but takes so-called just-in-time information retrieval (JITIR) systems in general into regard. The remembrance agent is integrated in an application for word processing to **retrieve useful information** such as email, net news or locally stored documents. Unobtrusiveness is addressed by interface design. A so-called **ramping interface** delivers only reduced information in the beginning and makes more detailed information easily accessible on user request. In both approaches, the **task** is represented by the content of a document that is currently edited by the user. The users highlight words or make annotations. This user input is transformed into queries to retrieve suitable documents. The generating of **implicit queries** without them being entered by the user manually is one of the main characteristics of proactive information retrieval. Holz et al. [HMBR05] use implicit queries for heterogeneous and distributed documents. The queries of Dumais et al. [DCSH04] are created from the content of emails. Ye and Fisher [YF02] propose a system for **reuse** of source code fragments by monitoring programming activities and generating queries from current source code. The approaches have in common that information is shown continuously in a widget next to the document.

Puerta Melguizo et al. [PBD+07] propose a deeper representation of the task in their recommender for writing a document. The task itself is distinguished in **phases** (e.g., planning, writing) and information need is supposed to be different in each stage. A **long-term user profile** helps to assess the relevance of the document. The retrieval of documents is triggered by assuming that the users are interested in the topic they write about. In case of **reading** a document, the challenge is to infer about what is currently in focus. Hardoon et al. [HSTA+07] apply eye movement to reason about the terms the user is currently looking at. Training a support vector machine leads to weights that represent the contribution of a term to an automatic query.

Proactive assistance approaches for offices are mainly document retrieval based approaches that are tailored to text fragment analysis. Reading or writing large amounts of text is difficult because of driver distraction. Analysis of a text in an automotive environment is not feasible because tasks inside a car are not characterized by reading or writing. Text-based approaches are based on frequent interaction with the system. That is also a source of driver distraction.

Monitoring User's Behavior

Proactive information retrieval approaches share the implicit assumption that text fragments determine the task and search queries are built from this text. There are other application scenarios where the user is assisted by tasks not exclusively described by text fragments. The challenge is to build powerful inference models to reason on user's actions during the task. Gong et al. [GNL+09] use **ontologies** to describe user's context related to required resources (people and machines) to perform tasks. A context reasoner infers on implicit high-level context (goals) of the users from low-level context (the resources, the users already used). Other resources that are associated with the inferred goals and not already used are automatically recommended.

In the Lumiere project, Horvitz et al. [HBH+98] monitor the user while working with the Excel spreadsheet application of Microsoft. Users' actions, the current document, their profiles (competence), their goals, task history and explicit queries are aggregated in a **Bayesian network** to reason on **user needs for assistance**. Benefits and costs of recommending help topics (out of 40) are balanced to optimize **expected utility** for the user. Goals are regarded as synonym for tasks and the user need is information that helps to fulfill the task. Reasoning not only depends on static models but also takes into regard **temporal** aspects of user goals, i.e., present or past goals cause future goals. The authors use a **"horizon of analysis"** in which observations are made for a specific goal. Not only current context information is used as evidence in the Bayesian network but a sequence. The reasoning results in a probability of user's need for assistance. If the probability of user needs for assistance exceeds a **threshold**, a notification is delivered (based on notification preferences). The interface is designed unobtrusively with an avatar. It delivers suggestions about what the user probably wants to do and if a help topic exists to that action. Details to the topic are retrieved on demand of the user. Without user action, the notification disappears after a timeout with an apology for the disturbance. Microsoft introduced results from the Lumiere project as a cartoon-like avatar called Clippy in Microsoft Office 97. However, many of the investigated features were not included in the final product (e.g., long term user profiles, differentiation between novice and experienced users, context-awareness and user action monitoring). Clippy used a simple rule-based system. This made it appear quite often. It was a failure in user acceptance and made Microsoft turn off the feature by default in the 2001 Microsoft Office release.

Proactive assistance with recommender systems inside a car does not support the drivers with the primary task (driving) but additional tasks (e.g., parking, refilling, hotel search) or for fun (e.g., sightseeing, music). Other driver assistance systems such as ACC are responsible to assist the driver while driving. Therefore, monitoring the user during task completion and reasoning on information need is not possible to enable proactive behavior towards recommendation delivery.

Reminder

The approaches discussed so far, focus on the benefit of information for a task. Other approaches deal with the **cost of delivery**. An example is an email application that receives messages and forwards them immediately to the user at arrival. Horvitz et al. [HJH99] propose such a system and investigate methods to be sensitive of user's **attention** in order to decide when to present the email. The trigger for the notification is the reception of an email. A **Bayesian network** reasons on expected **costs of interruption** and **costs of postponing** email notifications based on criticality and costs of delay. The system is integrated in Microsoft's email client Outlook and does not change the notification itself but the timing. Millard et al. [MdRS05] also describe a system for deciding proactively about email delivery with **ontology reasoning**. The current task of the user and content of the received email are analyzed to decide what to do with the email. If a message is assessed as relevant, the system also decides about the mode of output, i.e., on which device it should be delivered.

Besides emails, notifications can also be delivered to **remind** a user. Kamar and Horvitz [KH10] investigate reminders for Microsoft's Outlook calendar. A **Bayesian network** is used to establish a mental model of forgetting an appointment or details of an appointment (e.g., location or other attendees). Also the importance of a meeting accounts for the value of a reminder. The **timing of delivery** is chosen by the expected utility of a reminder, enough time until the meeting starts (not to forget it again) and closeness to the meeting location. The probabilities for the Bayesian network are learned from user interaction. CyberMinder by Dey and Abowd [DA00] gives the users the possibility to define **context conditions** in which they want to be reminded (situation editor). This extends common reminder approaches which mainly deliver according to the time. CyberMinder also incorporates context such as location or more complex combinations. The situation is described by **rules** which can be combined.

An approach that postpones or prevents information delivery because of costs (e.g., user disturbance) is useful inside a car but only for special uses cases such as appointments or emails. Its primary goal is to prevent delivery at the wrong time, not to enable delivery at the right time. The need for information is triggered by an external source, e.g., an email or a meeting, with the push of a message. For many use cases relevant for drivers (e.g., gas stations or parking lots), the system needs to decide on the need for information and has to retrieve relevant information if necessary.

3.1.2. Preference-based

Web Pages

Office assistance tries to help the users to be more efficient in their task. The reason for the trigger of recommendations can also be user's interest in information. For example, the basic idea in case of web browsing is to **watch user behavior** while browsing and generating implicit queries based on the content of a web page or previous web pages. Lieberman's Letizia [Lie97] is a widget-based recommender of web pages that continuously changes its content depending on the web page in the browser. WebWatcher by Joachims et al. [JFM97] asks the user in the beginning to state current goals and interests. This information influences the retrieval. Budzik and Hammond [BH99] extract meaningful keywords from Microsoft Word documents and web pages to generate queries. In contrast to the first two approaches, this approach takes **semantics of terms**, e.g., addresses, company names or images, into regard to select the right search engine. It shows the titles of the web pages in a widget. The users choose whether they want to see the whole item. A system called SUITOR by Maglio et al. [MBCS00] also regards semantics and suggests stock information in case of companies are recognized and tips in case of Microsoft Word is started. The system is an independent widget that builds short-term profiles based on the applications used by the user. Miettinen et al. [MTM05] extend these approaches. They analyze behavior patterns by monitoring navigation and scrolling to distinguish between **reading** and browsing. The eye movements of the users additionally determine the relevance of keywords. Implicit ratings are derived from these keywords. Henzinger et al. [HCMB05] apply these ideas to **TV broadcasts**. News TV broadcasts are regarded as text stream which is analyzed in real time to feed a news information source with implicit queries. Different approaches are compared by the accuracy of relevant results. Their results show that it is worth improving **postfiltering** of the information rather than extracting more accurate queries.

Billsus et al. [BHMA05] analyze a recommender called FXPAL for web pages. FXPAL is integrated in the web browser as toolbar. The authors realized that previous approaches lack a good interface. They made three suggestions to improve the interface. In their solution, a translucent recommendation window notifies the user about highly relevant recommendations. It should be **non-intrusive but also not too subtle** to be realized by the user. Highlighting of terms in the web page should provide **transparency** about why specific items were recommended. Finally, a recommendation digest function allows retrieving formerly delivered recommendations. If the users do not have enough resources to process recommendations at the time of delivery, they are able to **recall** them later.

Proactive preference-based retrieval is comparable to task-based retrieval. The interaction of the user with the system determines the need of information, e.g., documents or web sites. The described approaches are not applicable inside a car for the same reason as office assistance. News recommendations are an applicable use case for in-car recommender systems. However, they are triggered from external sources (news provider) like reminder.

Asynchronous Delivery

In e-commerce, a common technique to provide proactive recommendations is to send out **emails** (Ben Schafer et al. [SKR01]). Online retailers such as *Amazon.com* or *Ebay.com* create **long-term preference models** by analyzing user's purchasing and browsing history. A set of current offers that might be of interest for the users is sent via email. The recommendations are explained by "Because you recently searched for ..." or "Maybe your are still interested in ...". Another approach in e-commerce is to **integrate widgets** into the area where results are shown. For instance, *Amazon.com* shows other items with the explanation "Customers who bought this item also bought ...". As the users do not requested the recommendations themselves, the approaches correspond to an automatic delivery. However, the recommender is fully integrated in the search results and it is driven by the search query of the user. Therefore, we do not regard this approach to be proactive as the recommendations are part of the search engine. The job recommender of Lee and Brusilovsky [LB07] shows the distinction more clearly. The authors refer to automatic (recommending jobs) and manual (searching for jobs) recommendations. As the user does not ask for recommendations, the delivery is automatic. Though, recommendations are generated automatically, the delivery itself is not triggered automatically. Therefore, these approaches are not proactive because the system does not decide when to recommend.

Asynchronous delivery via emails is not applicable for proactive recommendations inside a car. The drivers consume proactively delivered information while driving. If recommendations would be delivered via email, driving will be frequently interfered or the drivers miss important information. Providing explicit explanations with recommendations has proven to be useful in commercial system, especially in e-commerce. Transferring explanation approaches to automotive scenarios is not straightforward because of information overload.

3.2. Proactive Mobile Recommender Systems

The main difference between context in mobile and desktop systems is that environmental context has more relevance for mobile information access. However, environmental context has to be captured by sensors first. This requires additional hardware in mobile devices which only offer limited space for extensions. Besides task-based and preference-based triggers, also environment-based triggers are applied in mobile scenarios.

3.2.1. Environment-based

Location-based Delivery

Location is one of the most used context information for delivering recommendations. The common idea is to deliver relevant information if the user is near a location. Rhodes [Rho03] uses the location in a system called Jimminy which is a mobile version of a JITIR (see Rhodes [Rho00]). Jimminy is a wearable computer with a head-up display which automatically displays **old notes** that are relevant at the current location and with the current accompaniment of the user. Notes are saved previously with their context. For instance, by entering a room, all notes taken in that room are shown. Similar systems are proposed by Beigl [Bei00] with MemoClip, PersonalServer by Want et al. [WPT02] for notes in a research lab or by Ryan et al. [RPM98] for archeological purposes. All scenarios share, that taking notes is inconvenient for the user. Therefore, information is **automatically saved** and retrieved if the current context matches the saved context. Marmasse and Schmandt [MS00] use in ComMotion the same assumption. The difference is that **future information** is shown. Instead of old notes, **reminders** are delivered. As with CybreMinder (Dey and Abowd [DA00]), the users are able to set up a reminder in connection to a location. To increase convenience for the users, the system learns salient locations and gives the users the possibility to tag these locations, e.g., "home" or "work". As the system is targeted to mobile users who are involved in other tasks during reminding, a **speech interface** is added. Context such as the location can be used for the trigger of recommendations but also for filtering items like in a context-aware recommender system. Vijayalakshmi and Kannan [VK09] **separate context** in context that triggers the delivery and context that is used for assessment. Basically, all mobile guides for tourists follow this idea.

Regardless of whether the location is used to store data, to deliver location-based information or to remind, researchers approach the context information location only as the position of a user. Their approaches use different granularity, e.g., a room or a GPS coordinate. However, the location bears more contextual information that determines user's need of information. In automotive scenarios, especially the route of a driver is a rich source of information, e.g., how long it takes to drive to a location or the purpose of the trip.

Mobile Guides

Many **mobile guides** were proposed over the years (see Krueger [KBH+07] for a review) but most of them rely on explicit search for information. Research in this area focuses on how mobile users can interact with the mobile system to retrieve relevant information of interest. Also technical questions in mobility are addressed, e.g., positioning sensors. The idea of proactive delivery of tourist attractions comprises a user who walks around and a mobile system that adapts its information delivery or notifies the user about interesting spots around. Following this idea, Oppermann et al. [OSJ99] implemented Hippie, a mobile guide for museums, and Schmidt-Belz et al. [SBNPZ02] propose CRUMPET, a proactive tourist guide. CRUMPET proactively triggers sightseeing spots at specific locations. An unobtrusive delivery with item icons and an audio notification is used where the user has to retrieve the items manually. Schmidt-Belz et al. note that proactively delivered items have to be "better" than pulled items. Therefore, the threshold for relevance is higher in their proactive scenario. This should help to make the reason for the recommendation more obvious for the user.

Proactive mobile guides are especially useful for **discovering new information** (Brown et al. [BCB+05]). The approach of Brown et al. recommends notes, pictures etc. which are taken at a location when another user reaches that location. Aras et al. [ALWM10] describe the same usage scenario for video- and text-based recordings. If a user enters a region, a **prefiltering** of notes based on time, location or other context is applied. A collaborative filter assesses the remaining notes. An icon indicates the availability of recommendations proactively. The user is able to retrieve desired notes. In the evaluation, proactive delivery gets higher scores in efficiency and convenience than searching manually. However, the icon was not recognized often which suggests using additional notifiers such as sound or vibration. Modsching et al. [MKHG07] investigate how effective mobile guides with proactive functionality are in discovering new information. In a user field study, the system guides the tourists along a tour and triggers sight information automatically when it is in vicinity. Results show that the users with proactive mobile guides **explored much more** sights than such without.

Additionally, Beer et al. [BRHF07] carried out a user study to find out for which kind of tasks mobile users in tourism want to receive push information. For four out of six services (navigation, risk warning, news and sightseeing) push functionality plays a significant role for the intention to use the system.

The focus of mobile guide research is on system design. Recommender systems and proactive assistance may be part of the system but only marginal. Only a few approaches combine recommender systems and proactive assistance with mobile guides. The evaluation is focused on the overall user impression of the system.

Events-based Delivery

Besides the location, also events could be the trigger for recommendations. Hinze and Voisard [HV03] came up with the idea to exploit **event-based** modeling to deliver relevant information automatically. Location events trigger information for a sight at the current position. Scheduled events trigger recommendations based on semantically related items. For instance, if a church is in focus, then another church around the corner may be recommended. External events trigger notifications if dynamic information changes, e.g., a concert is canceled. The event notification system consists of events and profiles. Events are state transitions at a certain time. They trigger actions and can be primitive or composite based on an event algebra. Events contain operators like "atLocation" or "weatherChangesTo". Profiles contain conditions that are matched to an event to decide whether to perform an action. Further characteristics and types of events are analyzed in Hinze et al. [HSB09].

In a series of publications, Beer et al. [BFH$^+$07], [BRHF07] and [BHZR06] extend the idea of event-based delivery. The authors let the users **configure events**. Beer et al. [BFH$^+$07] describe the architecture and carry out a study about which kind of information to deliver proactively to a tourist. The main requirement on their system is **ease of use**. Especially the configuration of triggers should be easy for the user. Furthermore, the user should have the possibility to provide user preferences and various sources of data should be integrated. The rule engine (Beer et al. [BRHF07]) is similar to the approach of Hinze et al. [HSB09]. It is based on so-called E-C-A rules (Event-Condition-Action). An event contains one or more operators and their parameters. Conditions check user situations and tasks, e.g., being outdoor, to decide if an event should trigger an action. The action is the delivery of recommendations. In contrast to Hinze et al., the authors explicitly use recommender systems for item assessment. Beer et al. [BHZR06] present a **hybrid recommender** containing collaborative, content-based and knowledge-based filtering to select items.

Delivering recommendations when events occur is a reasonable approach for proactive recommender systems. Authors such as Hinze et al. [HSB09] and Beer et al. [BRHF07] show the feasibility of event-based delivery. However, such approaches require the modeling of events. In automotive scenarios there are use cases, e.g., parking or hotel search, where information need is not always connected to the occurrence of events. Furthermore, the approaches monitor user activities and analyze events during these activities comparable to office assistance.

Calculation of Scores

Besides domain-specific approaches like mobile guides, also general approaches for proactivity were proposed in mobile scenarios. The approach of Wörndl et al. [WHBGV11] is one example of a generic model for proactivity in recommendation systems. The model separates proactivity and context-aware assessment. A **score** S_1 is calculated and triggers the calculation of a second score S_2 for each item. S_1 represents the relevance of a recommendation in user's situation and S_2 the quality of an item. Situational parameters are a **weighted combination** of a user, temporal, geographic and social context. The items are assessed based on one or more recommendation approaches including context-aware approaches. If S_1 exceeds a threshold, the calculation of S_2 is initiated. Every item whose score S_2 exceeds another threshold will be recommended. The score S_1 has an influence on the second threshold to make recommendations more probable if necessary.

Like Wörndl et al. [WHBGV11], Cena et al. [CCG$^+$06] also use scores. Cena et al. focus on different types of adaptation including users, context, contents, devices, interfaces and the modality of interaction (pull- and push-based). Their system UbiquiTO is a mobile guide that recommends POIs on a mobile device. To avoid **cold start** problems, recommendations are calculated by rules based on user profile information, e.g., age. For dynamic preferences, user actions are monitored and weights in the user profile are updated. In the **fuzzy rule-based** system, the antecedent comprises properties of the user, e.g., "$age \geq 45$", or features of an item, e.g., "hotel category = 4 stars". The consequence contains user profile attributes, e.g., "intention to spend a night = 0.7". The result of the user preference inference are defuzzified membership values μ for item features, e.g., "intention to spend a night = $\{\mu(\text{High},0.7) = 0.8$; $\mu(\text{Low},0.7) = 0.1\}$". This corresponds to situational user preferences. The recommender calculates a **score** at run-time for each item based on its features. Additionally, the authors use a second **rule-based** system. Rules with the membership to item features in the antecedent and weights for item features in the consequence are established. A second score describing the current context of the user, mainly the location, is calculated and summed up with the first score in a **linear model**. Another question that is addressed is the **amount of information** to provide. This depends on the device, modality of interaction, item features and the user profile. For instance, if a user is very interested in art, more detailed descriptions are delivered for museums. Rules determine the amount of information. The difference to Wörndl et al. [WHBGV11] is that this work does not detail the trigger for a proactive delivery but the processing of item features and the establishing of user profiles with rules.

Score-based approaches share the advantage that the assessment of the system is done on an abstract level, i.e., it does not depend on the application domain anymore. A reactive trigger with a threshold has the disadvantage that many recommendations are delivered in case the threshold is often exceeded. This kind of trigger is often used with scores. While driving, this would lead to many interruptions. Score-based approaches also have in common that the aggregation of different information sources is conducted with a linear model. However, multi-criteria decision making offers far more methods to derive a single assessment, e.g., AHP or Outranking.

Delivery based on Predicted Information

The approaches discussed so far mainly address the problem of proactive information delivery for present situations. Especially in the tourist domain, it is valuable to **predict user's future** location to prefetch information or deliver recommendations timely (e.g., see Hinze et at. [HSB09]). Many methods to predict destinations or context in general are proposed (e.g., Ashbrook and Starner [AS02]). In this work, we are not interested in the prediction itself but how to handle predicted context. Lee and Cho [Lee10] describe a classical scenario in this case. They use **Bayesian networks** to predict future information need. Nodes of the network represent activities, user preferences and profiles, time, weather and user's current position. The system infers on the future position of the user. Locations are described by higher semantic terms like "law building". If the probability of a prediction is high enough, the recommender retrieves relevant items associated with the location, e.g., the menu of a restaurant. The advantage of this approach is that information can be delivered in advance of the event. The user has **enough time** to consume information, e.g., to look through the menu.

In contrast to task monitoring, prediction of future tasks is better applicable for use cases of recommender systems in automotive scenarios. Prediction research focuses more on prediction accuracy than on the delivery of recommendations. Recommendations are simply assumed to be relevant if a task or event is predicted. They are directly connected to the event or the task. However, we are not directly interested in the task but which kind of information is needed for task completion. Hence, predicted information needs to be interpreted.

3.2.2. Preference-based

Publish-Subscribe Architecture

Environment-based proactive recommendations are triggered by external context, e.g., location or weather. The delivery is refined by user preferences. Triggers may also be primarily user preference-based. If a relevant item is available, it will be pushed and refined by context information if needed. Podnar et al. [PHJ02] describe a technical architecture to push information such as weather, traffic or news to a user. A common scenario involves a user who might be interested in information and a **content provider** who delivers information from time to time. The authors choose a **publish-subscribe** mechanism to distribute information. The users subscribe to a service or channel of interest with their preferences and the content provider publishes information if available. Information is filtered by a middleware for content dispatching which decides which subscribed users should receive the information. The dispatcher analyzes the content of a delivery and matches user preferences to the content.

Based on this mechanism, Choeh and Lee [CL08] describe a proactive information system. The authors focus their work on **user preferences**. They emphasize that push messages cause interruptions from the current task in general. Therefore, irrelevant items may lead to dissatisfaction with the service or the content provider. They choose a recommender system approach to filter items. Their **hybrid recommender** aims to avoid cold start problems. Two **scores** are calculated and aggregated with a **linear weighted model**. The first score is content-based and is calculated for each item individually with user preferences and item attributes. The second score is calculated collaboratively by clustering user profiles and take into regard what items other users have consumed in the same cluster. Items with high scores are moved to a push queue where the user client system is able to retrieve the items. Context-awareness is addressed by allowing the user to define **rules for the delivery**, e.g., restricted time periods or message intervals. In a user acceptance study, the authors found that push messages may lead to use the service more often. On the other hand, experienced users tend to be more sensitive to inaccuracies in item selection than beginners.

Lee [LC10] takes a closer look at the **content** of event notifications. He looks for an **event description** that is fast in matching and expressive at the same time. Therefore, he decides for a graph-based structure as a compromise to describe complex events. A context-aware collaborative filtering mechanism is proposed for content matching. First, users who rated the topics of an item in context are searched. Those users who have similar interests are filtered. The decision whether to deliver a recommendation is made based on the ratings of similar users.

Like reminders and news, push-based information delivery is triggered by external sources. Their goal is not to assist the user but to deliver information. Proactive assistance in this case is focused on postponing or filtering information. Rules for the delivery are a reasonable method to prevent pushed information from being obtrusive. Furthermore, the user defines the behavior of the proactive assistant. The more complex the behavior is, the more effort the user has. The complex rule definition might not always be easy to understand the way it was intended by the engineer of the system. In automotive scenarios, rule definition is limited due to limitations of interaction inside a car.

Mobile Advertising

Mobile advertising is one of the main applications for preference-based delivery. The content provider is an advertiser in this case. Ranganathan and Campbell [RC02] emphasize that for publish-subscribe push services, the user determines the conditions for delivery and in mobile advertisement it is in the **interest of the advertiser** to deliver information. Hence, the advertiser should be allowed to specify conditions by itself. The user may specify preferences and configure the delivery trigger. Bulander et al. [BDSK05] describe a platform called Mobile Marketing (MoMa) to enable mobile advertisement. The authors distinguish different kinds of delivery. A **broadcast** sends out the information to every mobile user whose system may refine the delivery by context or user preferences. A **point-to-point** centralized delivery sends ads to the users as soon as they are close to a retail store that offers promotions (e.g., Munson and Gupta [MG02] or Aalto et al. [AGKO04]). The third scenario is an **ad-hoc** decentralized delivery where two users share recommendations when they meet. Straub and Heinemann [SH04] detail this scenario with **mobile ad-hoc networks** (**MANET**) as a communication platform. User A collects information and recommends it in case it is relevant. If user B meets user A and both have similar preferences, recommendations are delivered automatically over a spontaneously established communication. Bulterman et al. [BJC^{+}07] propose a system for a personalized remote control with the same idea of sharing. User A and user B watch TV (not necessarily co-located) and both users collect information about the TV show. If they believe that it can be relevant for the other, they are able to push a recommendation to the other user's remote control.

Regardless of whether ads are triggered by an advertiser or by friends, external sources decide what is relevant for a user. In automotive scenarios, it could cause information overflow, if too much information is pushed to the driver. Therefore, pushed information should be filtered before delivering to the user.

News

Besides advertisement, recommending **news** is another popular use case for preference-based triggering. In this scenario, a news content provider publishes large amounts of news. To avoid information overload, recommender systems are used to decide which news should be forwarded to the user. Yeung and Yang [YY10] describe a proactive mechanism for news based on the MCDM method **AHP**. The authors notice that most researchers aggregate several criteria by a simple weighted sum model. Their AHP is able to evaluate items on several hierarchies and with several recommender approaches. The user subscribes to a news service with the news category. A content dispatcher collects current news from the content provider and matches user preferences, current needs and available content to decide about delivering a push recommendation. The first hierarchy level of the AHP contains a profile (including a Bayesian network for user interests), situations (e.g., location, weather), rates (i.e., selection history) and the age of an article. Weights for the criteria are either assigned manually or based on selection history. A combined **score** is calculated with the AHP method to rank the items. If the score is high enough, the news item is pushed to the user. To overcome cold start problems, a static structured Bayesian network predicts the interests of a user in a specific news category (class variable) by means of demographic data, e.g., gender, age or work status.

News are only one application for proactive recommendations inside a car. There are other information sources, e.g., parking or refilling, which have different properties. AHP is an interesting approach for aggregation. It goes beyond simple linear models. However, like an environment-based approaches with scores, a reactive trigger with a threshold has the disadvantages of frequent delivery.

Delivery based on Predicted Events

Thawani et al. [TGSR07] describe a different approach for advertising based on prediction of events by means of state machines. The task of the user is to watch a sports game, e.g., a soccer game. An ad is pushed by the content provider during a game when it is most relevant. It is most relevant if the content of the ad matches the current scene in the game, e.g., a player that is featured in an ad scores a goal. The users subscribe for the service with basic preferences such as their favorite player. A domain-specific **state chart** represents the game. Events like "Player X scores a goal" cause state changes. Analyzing the former occurrence of events derives the probability of a state change. If the probability of an event becomes high enough, the event is **predicted** to occur in the near future. This causes the system to retrieve relevant ads for this event from an ad pool. Ads are provided timely before the event occurs. The approach is only applicable if the domain can be described as state chart and with events.

Kwon et al. [KKY04] also rely on **context prediction**. Their system fetches relevant data and delivers on time. Their approach uses **rule mining** to build behavior models with context and recommendation rules as a user profile. If a behavior pattern can be predicted during run-time for a specific **window size**, i.e., the number of predictable patterns, then information for that pattern is prefetched. Items are selected by a **recommendation rule** and they are delivered if the user behaves according to a known pattern, e.g., the next department in a bookstore.

Bohnert et al. [BSZ09] apply a similar idea for museum visitors. They develop a **spatial process** model that predicts where a museum visitor is probably going next after viewing a painting. User's viewing time is regarded as implicit rating for the item. Recommendations for further exhibits are made based on the predicted path, distances between the exhibits and inferred ratings (based on the content similarity of items).

Predicting events is a promising approach for proactive recommendations inside a car because it allows a timely delivery of recommendations. Hence, the driver has enough time to consume the recommendations. The focus of current research is on the prediction itself, less on the quality of items that are delivered.

3.2.3. Task-based

Task-based delivery is the third category of mobile recommender systems. The task of a user triggers the recommendation. In general, tasks follow an **intention** or a **goal** (we do not distinguish the terms in this thesis). Actions that the users perform lead to achieve the goal. During the execution of actions, the users have a **need of information**. A need is the gap between current information and a goal.

Early Approaches

A general approach for task-based information delivery is monitoring the task and inferring on information need. Lamming and Newman [LN92] explain early ideas of **storing notes** along with information about the task of a user, e.g., travel or meeting, as a mobile automatic diary. Hart and Graham [HG97] propose with FIXIT an automatic **manual recommender** for copy machines that is able to retrieve information proactively. The system is embedded in the copy machine and tracks the state of user's troubleshooting. It automatically recommends pages of the user manual with the potential fault of the machine. The fault of the system is detected by means of a **Bayesian network**. Every fault is connected to a topic in the manual. The system displays a small icon indicating new available recommendations. The user has to retrieve these recommendations manually.

The approach of Hart and Graham is based on ideas of information retrieval. It is embedded in a special device and hardly applicable in other domains.

Exercising

The Xpod is a special device developed by Dornbush et al. [DEO$^+$05] which tracks human activity (e.g., while exercising) to select appropriate **music** automatically. The scenario is similar to driving because the users are not willing to interrupt their primary task of exercising to interact with the system. Hence, the interaction should be as easy and short as possible. The device is equipped with sensors to monitor context information that describe the activity, the motion and the physical state of the user. A **discrete set of states** of the user is predefined and the system infers on the current state. The user is able to give ratings for recommended songs. These ratings are stored in relation to the current state. Ratings for songs are predicted based on previous ratings for similar songs (content-based) and the current state. As songs are consumed more often than one time, several ratings are stored. Hence, a rating can be different for different states. The predicted ratings are used to weigh the probability that the song is played. Several **machine-learning** techniques are examined to calculate the rating prediction.

Although the model is tightly tailored to the task (like the approach of Hart and Graham [HG97]), it shows that usage of common machine learning methods are applicable for many different domains to model the trigger for a recommendation. The domain of exercising shares some characteristics with driving.

Emergency Workers

Similar to exercising, individuals or teams working in an **emergency situation** also need to focus on the primary task in which few or no interruptions are acceptable due to time pressure. Fan et al. [FYW+04] point out research that shows that **teams** using proactive information delivery outperform teams without proactive delivery. However, irrelevant information that is delivered proactively can have negative effects on the performance. The authors propose an approach to help teams to receive only relevant information based on complex domain-specific mental models. **Agents** monitor team members in their execution of subtasks. Relevant information is filtered with **rules**. For instance, information that has a high probability or is observed by the team itself is not delivered . Information need is modeled as a graph where states of teams are nodes. Hu [Hu09] proposes an **ontology**-based approach for the same scenario. A domain specific ontology supplements a generic task ontology containing the task itself, time, location, the user and occurring events. Context changes are monitored. If changes occur, a set of predefined **rules** that are based on the ontology determine which information (events in this case) to deliver to the user.

Ontologies are a powerful tool to model complex relations, e.g., in a task model, in a formal way. In contrast to machine learning methods, uncertainty is not an integral part of an ontology.

Shopping

A further scenario of proactive recommendations is **shopping**. The basic idea is to deliver products inside a store based on the situation of shopping, e.g., which products are still needed or where the user is. Asthana et al. [ACK94] describe such a mobile shopping assistant but their research focuses on **infrastructure** questions. Sae-Ueng et al. [SUPOK08] published a more recent approach. A camera is used to infer the state of the user and RFID is used for positioning. Logs of the sensors are used to set up a model with **multiple linear regression** to be able to infer on the state of the shopper, e.g., standing, viewing, touching, carrying or fitting (cloth) (similar to Dornbush et al. [DEO+05]). A **hybrid recommendation** approach is proposed for item selection. The content-based filter uses previous purchases. The collaborative filter extracts implicit ratings from the current state where interaction with the product such as touching leads to higher ratings. The recommender searches similar customers with similar ratings to make a prediction for the current product. Product items are prefiltered. For instance, the items can be prefiltered based on the department a user is in. Automatic recommendations are presented in a display next to the product or on a mobile robot.

The use case of shopping is different to automotive use cases for proactive recommendations. Shoppers can more easily be monitored during shopping to infer on need of information. If information is presented, the shoppers are able to interrupt their activity and focus on the recommendation.

Situation-aware Delivery

The approaches so far aim to assist special groups of users such as shoppers, athletes or emergency workers. Therefore, the approaches are mostly limited to special characteristics of the domain. Other approaches focus on an **everyday mobile user** equipped with a mobile device. Kwon et al. [KYS05] propose a **multi-agent based system** which delivers useful information during the primary task of browsing with a mobile browser. The assumption is that browsing for information while on the go aims mostly to cover information need which is based on the situation of the user, e.g., at lunch time. **Ontologies** are established to describe users and context. Context is associated with information sources. The strength of the association depends on user preferences, context (e.g., location or schedules) and users' to-do lists (e.g., what the users need to purchase). Based on the strength of an association, a strong push, i.e., a notification, or a weak push, i.e., a link to a recommendation, is applied.

Bouzeghoub et al. [BDW09] propose a system that recommends **resources** such as individuals or buildings on a university campus. Their focus is on **situation awareness** for proactive recommendations. Situation awareness contains location, user profile and user's tasks from the daily agenda. Like Kwon et al. [KYS05], a set of **ontologies** is used to describe context. A situation is a set of context information in a time interval. It consists of relations like "in(Tom,Office)", constraints to be satisfied, an interval stabilization and a time range. The **interval stabilization** keeps the situation in a stable state, i.e., only significant context changes change the situation. Furthermore, a situation may include special operators for **temporal aspects** to be able to incorporate context in the future or the past, e.g., "soon". Events are defined as transitions of situations such as moving from one building to another building. The events contain constraints that need to be fulfilled to trigger an action. Constraints are logical expressions similar to rules. Recommendations are triggered in case a situation changes, i.e., if an event occurs.

The approach of Bouzeghoub et al. [BDW09] uses a **knowledge-based approach** that processes rules for recommendations. Rules contain events in the antecedent and a recommendation action with the type of recommendation and the user in consequence.

Additionally, constraints based on situations are analyzed. A situation monitor detects context changes and delivers recommendations to the interface. If too many items are available, information filtering is applied. The application scenario includes only small sets of items in general. The system intends to make complex situation inference. For instance, if the users do not checked their emails for a period of time, the system may recommend (highlight) buildings with computer terminals. It is questionable whether the users understand such recommendations. Transparency is handled by using icons to show the user which type of recommendation is fired. The system is **unobtrusive** by only highlighting items such as buildings and it allows the users to retrieve further information if they are interested. It is a retrieval-based approach where the user needs to interact with the system but do not have to formulate a query.

Bouzeghoub et al. [BDW09] show how situation awareness can be incorporated into proactive delivery of recommendations. The model includes important aspects of situation awareness such as the relationship between users and their situation and the stabilization of system behavior because of uncertainty. Again, recommendations are delivered based on the occurrence of events.

Retrieval-based Proactivity

Hanamura et al. [HKT03] also apply retrieval-based proactivity. The users receive recommendations as soon as they turn on their devices. A model for user intentions with **fuzzy sets** based on **radial basis networks** (**RBF**) determines the recommendations. The network connects main intentions, subintentions and situations represented by nodes. Main intentions are basically tasks or goals of a user and subintentions are subtasks or actions that need to be fulfilled to achieve the main intention. Tasks are activated by situations, i.e., a numeric value of task involvement is calculated. Situations are recognized by rules that are fired based on context information. Some situations are modeled fuzzy such as the time to an event (early, on time or late). The network proactively delivers subintentions ranked by their involvement when the device is turned on. Recommendations are connected to the subintentions. For instance, walking to a station leads to the recommendation of the route to the station.

Fukazawa et al. [FLW$^+$06] propose a similar modeling approach based on **ontologies**. A task ontology contains distal goals (long-term goals) which correspond to main intentions. Distal goals consist of proximity goals (short-term goals) which are subintentions. Services which should be recommended are connected to proximity goals, e.g., automatic route planning. A **situation ontology** enables proactive retrieval of services for subintentions by sensing context. Context implies situations and the system infers on proximity goals from situations. High-level situations are represented by keywords like "Sunday morning". Crisp context information is mapped to such terms.

In automotive scenarios, retrieval-based approaches have the disadvantage that the drivers usually do not look up information often (in contrast to mobile users). Mobile device users are able to turn their attention from time to time to their devices. This is true only in a few cases for driving (e.g., at a traffic light). The drivers focus on their primary task of driving during the whole trip. Therefore, the risk of missing relevant information is high. The approaches show the usefulness of fuzzy modeling for situations. Fuzzy logic is able to handle uncertain knowledge about situation involvement. The usage of human understandable terms like "Sunday morning" adds transparency to an intelligent black box system.

Activity Prediction

The focus of the approaches for mobile devices so far is to deliver information in the current situation. The next approach aims to infer on upcoming tasks and to deliver information for these tasks. In a series of publications ([BPR+08], [PP09] and [DPH+09]) a system called Magitti was proposed to recommend **leisure time activities**. Information sources are POI services such as Yelp, Zagat or Google local search. The assumption is that information is only relevant if it follows activity patterns. For instance, restaurant recommendations are relevant if someone wants to have lunch.

Bellotti et al. [BPR+08] describe the system architecture. The approach follows a **separation by design**. One component is for recognizing activities (Activity module) and another one for selecting suitable recommendations (Recommendation module). Both modules exploit context information to make decisions and they learn to improve their behavior. The user interface covers the whole screen of a mobile device. It presents the **state of the activity recognition** as icon to make clear why something has been recommended. The users have the possibility to change that prediction. This way, they are able to adapt provided items. The approach is a retrieval-based approach without explicit queries. A user study shows that the users discover new places, e.g., to eat, contrary to their regular pattern of taking a few known places (53% of the places are new). Inaccuracy or omission of expected items have negative effects on the confidence in the system (also shown by other researchers such as Cheverst et al. [CDM+00], Herlocker et al. [HKTR04] or van Setten [vSPK04]). The study shows that users use the system like a mobile guide. This leads to users' expectation of close places to be recommended. The first item in the list influences the view on others significantly. People spent much time to understand why the system has recommended those specific items (lack of transparency).

In another work, Partridge et al. [PP09] detail the task inference approach. The engine predicts tasks and user preferences in these tasks. A **horizon** for prediction is applied (2 hours) to lower complexity. The goal is to provide recommendations for upcoming tasks. A task is defined by single categorical terms such as "Eat" or "Shop". The prediction results in a probability distribution over the terms. Environmental context as well as user context is involved. **Structured models** are applied to make inference (e.g., based on probabilities) instead of machine learning to be able to use existing data sets and avoid cold start problems. Models are distinguished between static models that are once configured for all users and dynamic models which are updated based on observations such as interaction or places to go. The predictions of all models are combined to a probability distribution. Finally, relevant information is selected based on the final probability distribution.

Ducheneaut et al. [DPH$^+$09] describe the **hybrid recommendation system** in this approach. Again a **set of models** is used, e.g., collaborative filtering, explicit preferences, implicit learning from interaction or a distance model based on the location of the POI and the user. The prediction of each model for the preference towards an item is combined with a **linear weighted model** where the weights are adapted over time. Evaluation suggests that collaborative filtering seems to be useful for the users who are unfamiliar with the area. A mixture of models provides more divers recommendations.

The separation of concerns in a task and a recommendation engine is a promising idea for proactive in-car recommendations. If the system knows the task which the user has to fulfill in near future, it is able to derive relevant recommendations for the task and deliver it. The underlying assumption is that if a type of task occurs, information is needed. However, this is not necessarily true. If the users always attend the same restaurant for eating, they are probably not interested in restaurant recommendations for the task "eating". Hence, information need during a task is more significant than the occurrence of a task for recommendations. Concerning transparency, Partridge et al. [PP09] go further than other approaches. They show their inference result to the users and let them to correct the system. This contributes to a better comprehensibility of the system.

Situation recognition

An important component of proactivity is **situation recognition**. There are several approaches available in research for recognition in general. For instance, Goix et al. [GVCF07] explain how to infer on situations of mobile users with **rules** based on high level interpretations of situation states like "location = home" or "activity = working". Ciaramella et al. [CCLM09] focus on situation recognition for delivering proactive service recommendations for tasks. They emphasize the usefulness of **fuzzy logic** in this context. Real world events happen approximately, e.g., an approximate time or location. Hence, recognition of situations should contain vagueness. Furthermore, sensors are subject to uncertainty and imprecision. Another reason for fuzzy logic is that systems with a user interface should deploy human understandable concepts to be intuitive like natural language terms. Therefore, fuzzy logic is promising as a description language. The authors propose a structured model with **ontologies** for context (time, location and calendar) to infer on situations and for tasks to connect situations to tasks. Situations are seen as a set of context information that is invariant within a period of time. They distinguish fuzzy inference and crisp inference. If the resulting certainty is larger than zero with fuzzy inference, the situation is considered to occur (with uncertainty). Ciaramella et al. [CCMS10] describe the integration of fuzzy logic into the ontology description **semantic web rule language** (**SWRL**) in detail. If a situation is inferred, the task ontology links tasks to services that are needed in that task. The inferred need for services is delivered to the mobile device. The device selects appropriate resources such as files based on keywords and context. As soon as the system infers that the user is involved in a task, it triggers recommendations for required services in that task.

Ciaramella et al. show that fuzzy sets are useful for situation representation. The focus of their research is situation recognition. Recommendations and proactive assistance are of minor interest.

Unstructured Models

Many approaches use predefined structured domain models that are built by experts. However, it is also possible to learn models with data mining tools. Cheng et al. [CSC+08] use **unsupervised learning** to determine which information is relevant in which situation and how user preferences are. To overcome cold start problems, a statistical model for artificial usage data is generated to build the model. A snapshot of relevant context information, e.g., time or location, determines situations. Tasks are determined by user interaction and involved applications or web sites. Situations and tasks that are performed in these situations define a task pattern by means of **co-clustering**. The centroid of a cluster determines the recognized situation. At run-time, the system calculates the similarity between the current situation and the model and ranks tasks performed in this situation. The ranking is shown to the users in a widget on their mobile device (retrieval-based). Recognized situations are icons which makes the system more understandable.

Lei et al. [LZS07] use **reinforcement learning** to decide what to recommend. Their situation model is more complex compared to Cheng et al. [CSC+08]. The four-level model is based on Endsley's general model for **situation awareness**. The first level consists of atomic data elements directly from the source, e.g., row data from sensors or feeds from a web service. The second level aggregates and interprets the data atoms as data elements with an **interpretation** function, e.g., temperature as cold, pleasant or humid. Data transitions are defined by actions that lead to a state change of data elements. They consist of the action, a beginning and an ending state. The fourth level defines situations as an ordered combination of actions. This involves a data transition and a temporal relationship between the transitions according to Allen and Ferguson [AF94]. Information delivery is based on an **interval-based approach** rather than a point-based approach, i.e., a situation is an interval rather than a snapshot in time. For instance, the situation "entering a building" is a state change from outside the building to inside. The authors use reinforcement learning to analyze relationships between a recommendation and a situation with unstructured data. The user gives feedback to delivered messages (cancel, more information or just more) and the system applies a reward to each type of message in a situation (relative to the level of detail of the message). **Delivery policies** determine when a message is triggered. The trigger is described by an interval based on the reward of a message and its level of detail. A dispatcher sends out the messages as SMS, e.g., stock data.

In contrast to a structured model such as fuzzy logic or ontologies, these approaches learn the relation between context, situations and need for information. The representation of situations is only an abstract construct of contextual information. Hence, the user is not able to reconstruct the reason for information delivery out of the black box. Another disadvantage is that such models need to be trained before application. Training based on interaction with the user and selection of items is difficult while driving due to limited interaction possibilities.

3.3. Proactive In-Vehicle Recommender Systems

Improving In-Vehicle Interfaces

In-vehicle information systems (IVIS) aim to make information easily accessible. They use controller such as BMW's iDrive controller, interaction techniques such as speech interfaces or intelligent visualization such as proposed for POIs by Ecker et al. [EBJ09]. The research in this area mainly focuses on the interface. For instance, Krueger et al. [KBS$^+$04] propose a "connected interface" between desktop computers at home, the car while driving and a mobile device outside the car. The so-called BMW Navigator assists the user in information management such as route planning or parking spot finding. The separation of information access and connection with a **seamless interaction** on homogenous interfaces makes it easy for the user to access each kind of information through the appropriate interface.

Reduction of Information by Postponing Information Delivery

With recommender systems, we follow a different approach of information access inside a car. Our main goal is to **reduce the amount of information** as much as possible and deliver information at the right time. This goal can be achieved in many ways inside a car. Alt et al. [AKS$^+$10] propose a system that uses **standing times** at a traffic light to deliver information proactively. As information consumption while driving is difficult, the authors use short standing times where the drivers can focus their full attention to the system. The authors emphasize that people spend more time at traffic lights in recent years. This is due to higher volume of traffic in cities. The work describes how standing times can be learned from GPS traces (position, speed) based on a state chart model. The length of delivered information is adapted to the standing time in order to bring the attention of the driver back to the driving task as soon as the traffic light turns green again.

Although standing time delivery is far more effortless than while driving, it is restricted to short periods. These periods can be very short which gives the driver not enough time to make a decision if necessary. Additionally, automatic inference of waiting times is difficult because most digital road maps do not have information about traffic lights nowadays. Finally, traffic lights need to be equipped with additional infrastructure to allow reliable estimation or a central server needs to know all current waiting times. The traffic light method is cost-oriented, i.e., it tries to reduce the cost of delivery.

Reduction of Information by Adapting In-Vehicle Interfaces

Another way to reduce information is to adapt information intelligently before presenting it to the driver. In the work of Bachfischer et al. [BBH+07], an **intelligent agent** for in-car POI selection is proposed. Context information is used to filter POIs before displaying on the digital map of the central information system (CID). This improves usability as well as utility of the information system. Two **Bayesian networks** assess driver load and relevance of POI categories. The driver load indicator is based on speed, rain, visibility and other context information. It is used to adjust the number of displayed POIs. The higher the driver load, the less POIs are shown. The second Bayesian network is used to derive a general interest for POI categories. For instance, if the current gas level is low, then gas stations are displayed on the map instead of tourist attractions. Specific POIs are selected by a **rule-based system** according to POI attributes and a user profile. If the drivers are interested in one of the displayed POIs, they are able to retrieve further information by touching the display at the location of the POI. However, it might not be easy for the drivers to understand why the system shows that category of POI. As long as the selection is obvious, this should be no problem but unexpected displaying of POIs could be a problem.

The approach of Bachfischer et al. [BBH+07] resembles research in the area of **adaptive user interfaces** for in-car use. Rodriguez Garzon and Poguntke [GP11] explain such an adaptive user interface for POI recommendations. The user searches and browses through a specific category of POIs, e.g., super markets, according to a pattern, e.g., at a specific location along the route. After a learning phase, the interface jumps directly to the POI when the user starts interacting with the system. The goal is to reduce interaction steps.

Research in the area of adapting interfaces intelligently focuses on the display and interaction with the user interface. These approaches mainly adapt the interface based on contextual information. They make the access of information easier. However, the systems do not make any recommendations proactively. Therefore, the risk to miss relevant information is high.

Reduction of Information by Filtering Information Intelligently

The third way of reducing information is intelligent filtering like described by Ablassmeier et al. [APRR07]. In contrast to Bachfischer et al. [BBH$^+$07], the authors follow a **mixed-initiative** approach where the agent can be invoked manually or takes initiative by itself. They motivate for reactive as well as deliberate software agents based on environment, user and system context. Two **agents** are described based on **Bayesian networks**. The probabilities in the networks are set up with data from a user study. The contact agent monitors appointments with contacts from the address book. If an appointment could not be met, several possible actions such as calling the person are inferred and pushed to the user. The gas station agent monitors the current gas level, the consumption and the distance to the destination. If the destination is not reachable, the driver is asked whether gas stations should be displayed. Gas stations are filtered and ranked by means of a further Bayesian network and attributes such as brand, price or distance to the gas station. Both agents learn from user selection. An offline evaluation shows that the agents select better gas stations after a learning phase. The number of selection steps for a gas station could also be reduced. The reason why recommendations were made is transparent because the trigger (reachability of the destination based on the current gas level) is hard coded and it is the only situation for which the agent provides information. The focus of the approach is on the assessment of gas stations.

Horvitz et al. [HKS07] discuss time pressure due to an appointment and gas station selection. In contrast to Ablassmeier et al. [APRR07], these two aspects are combined to assess items. First, a **Bayesian network** to estimate the cost of time is constructed by asking users which costs they would apply to delay. The detour each gas station causes is also considered by calculating alternative routes to the destination via the location of the station. The model for cost of time together with the current schedule of the user is then combined to the **cost of divergence** of a gas station in dollars. The more important an appointment is, the higher the costs for a detour. The calculated costs are added to the total costs of refilling based on the remaining gas level at the gas station and the price of gas. With the total costs, the gas stations are filtered by means of a ranking. The top 5 items are displayed to the users on their mobile devices. The users are able to configure the time of delivery based on the current gas level of the car. The delivery of information itself plays no important role in the work. Therefore, the system is not integrated into an in-car information system.

Cho et al. [COYO06] also make use of **route planning** to improve item selection. They proactively create routes with POIs from the current position of the car to the destination. Their idea is to predict context information along the routes to be able to take upcoming items into regard. These items might be better than surrounding items. Other location-based systems push items as soon as specific context conditions are fulfilled without taking into regard upcoming context and items. The authors show the advantage of using planning mechanisms for item filtering. For instance, the frequency of recommendations can be reduced over the whole route.

Only a few approaches explicitly use recommender systems inside a car. The work of Wörndl et al. [WBE08] uses **recommender systems** techniques to assess items for in-car recommendations. Items are **prefiltered** based on context and user preferences. For instance, context such as the current gas level filters all gas stations not reachable with that level. User preferences such as the gas brand exclude all items not matching the preferences. Then, other cars within vicinity are searched by means of an ad-hoc network. Thereby, other user ratings and gas prices can be exchanged. A **multidimensional collaborative filter** uses this information to assess gas stations. The approach focuses on rating-based assessment of items. However, in-car use cases such as refilling or parking may lack enough ratings.

Another use case for the in-vehicle application of recommender systems is **music recommendations**. Turlier et al. [THG10] present an interface that allows a driver to set up a play list easily with **multi-criteria fuzzy terms** for genre, popularity or mood. The recommender selects appropriate songs and creates a play list proactively. The drivers only need to interact with the system if they want to skip a song, to adapt the play list or to change play list settings. Applying automatic detection of the mood as Park et al. [PYC06] proposes may further reduce the interaction.

Assessment of items with recommender systems or intelligent filters for vehicle use cases focus on the filtering mechanism itself. Proactivity is mostly out of scope or a simple trigger such as in the work of Horvitz et al. [HKS07] is used.

User Acceptance of Proactively Delivered Information Inside a Car

Providing information while driving has an influence on **user acceptance** of that system. Tang et al. [TGC$^+$09] made an interview-based study of **mobile ads** in a portable navigation device (PND). Context-aware ads for nearby retail stores were shown while driving, at the destination and at traffic light stops. Large ads were more disliked and stated as intrusive compared to small hints. Ads while driving are also disliked by half of the participants, especially when they were interrupted in tasks such as navigating. However, this depends on the experience with the device. The results are only partly transferable to other application scenarios of proactive recommendations inside a car because advertisement does not support any task of the user. However, user acceptance is an important issue concerning proactive system behavior (see the Lumiere project of Horvitz et al. [HBH$^+$98]). As proactive recommendations always interfere with the current task of driving, we cannot be sure that the drivers accept such systems. Although the users tend to accept proactive delivery on mobile devices (e.g., push notifications of social networks or news), the automotive environment is different.

3.4. Summary

The focus of automatic in-car information systems is either on adaptive user interfaces or on the assessment of relevant items based on contextual information. A popular use case is a gas station information system. To our best knowledge, there is no overall investigation of proactive recommender systems inside a car. A common definition should be given along with general requirements. Bayesian networks are a promising tool for intelligent behavior in automotive scenarios. They are applied in many research approaches. Also other domains of proactive information delivery on desktop computers and mobile devices rely on Bayesian networks. However, Bayesian network models depend on specific data of a domain. They need to be investigated with available data sets.

In many approaches, the awareness of user's situation and context is part of the proactive behavior. Often, only simple models are used to trigger recommendations. For instance, mobile guides use the location of a user to deliver information. More complex models rely on machine learning methods such as Bayesian networks. Situations are often represented by structured models, e.g., ontologies or fuzzy sets. Fuzzy sets have been proven to be a powerful method for situation representation because they support comprehensibility and cope with uncertainty. Another way of situation representation is an unstructured model. However, it is less applicable in automotive scenarios because complex machine learning is necessary. Situation representation is only one component of situation awareness. Prediction of future situations and decision making are further components. Only a few approaches take these aspects into regard. Instead, proactivity is often modeled in an event-based fashion. If events occur, then information is delivered. However, many applications of recommendations in automotive scenarios such as parking and refilling do not explicitly depend on events. Additionally, event-based modeling requires user task monitoring. This is difficult inside a car because recommendations are not given for the primary task of driving.

Another important element of a proactive recommender is the recommendation approach itself. The incorporation of context and the usage of several methods in a hybrid approach are common for proactive recommenders in mobile scenarios. Most of the approaches use structured models or user data from other domains to avoid cold start and ramp up problems. The results from these sources are aggregated. For instance, some approaches use an abstract score or linear models for the aggregation. However, decision theory offers far more than linear approaches, e.g., AHP, Outranking or dominance-based filtering.

Research in intelligent systems indicates that transparency in case of intelligent proactive behavior improves user acceptance. Nevertheless, only a few approaches explicitly investigate additional methods to make proactive recommendations understandable for the user. Mostly, transparency is handled by design, e.g., with icons or by highlighting objects. In recommender systems research and in commercial systems, e.g., *Amazon.com* or *Ebay.com*, explicit explanations gained popularity to make recommendations comprehensible. To our knowledge, no approach of proactive information systems inside a car used explicit explanations so far.

Nearly all approaches of proactive delivery emphasize the need of an unobtrusive behavior in system design. However, following design guidelines does not necessarily make the users accept a proactive system. Evaluation results of user acceptance are in general difficult to transfer from other domains to automotive environments. Desktop approaches are different because they require a lot of interaction and do not use physical context. Mobile assistants for domains such as shopping or exercising allow the users to interrupt their primary task and focus their full attention to the system. Researchers with similar approaches in the automotive environment do not present results for user acceptance.

4. Proactive In-Vehicle Recommendation Systems (P-IVRS)

Proactive recommendations inside a car are not investigated intensively in research yet. We define our understanding of proactive in-vehicle recommender systems (P-IVRS) in the context of recommender systems. Several use cases in the area of POI and route recommendations are potential applications of proactive recommendations. Based on our review of literature, we set up requirements on a P-IVRS. The discussion of the automotive environment in Section 2.1 shows that the design of in-vehicle information systems (IVIS) is limited in information output and input. Driver distraction and information overload have to be regarded when designing IVIS. Functional requirements comprise an intelligent behavior of the system, incorporation of situational information need, selection of useful items and a comprehensive system behavior. We describe which components of a P-IVRS are most important and how they cover functional requirements. Besides the processing of knowledge, we also describe the representation of knowledge inside the system. The central controller of our P-IVRS is the decision making process. It decides whether to look for suitable items, which items are relevant and when to recommend them.

4.1. In-Vehicle Application of Recommender Systems

We define proactive recommendations inside a car in relation to similar approaches like context-aware recommender systems (CARS). There are several use cases of proactive recommendations within a car. We group them according to the main automotive context that triggers the recommendation: car, environment or driver. Our investigation follows mostly the use case of a gas station recommender. To narrow the scope of this thesis, we suppose that relevant information and sensor data is accessible. Information sources for gas station assessment already exist, e.g., gas prices, and a simple sensor like a gas station indicator can easily be integrated.

4.1.1. Classification of In-Vehicle Recommendations

Figure 4.1 shows our classification of modes that are relevant in our environment. The area outside the circles covers other **recommender systems** (**RS**) that we know from

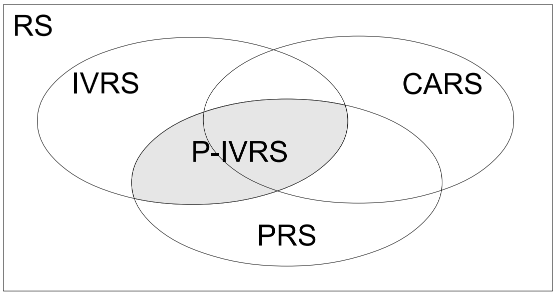


Figure 4.1.: Application modes of recommender systems inside a car

many research approaches or services such as *Amazon.com* or *Netflix.com*. A context-aware recommender system (CARS) incorporates contextual information in the recommender process to improve recommendation quality, e.g., a context-aware movie recommender (e.g., Adomavicius et al. [ASST05]). An **in-vehicle recommender system** (**IVRS**) comprises the application of recommender systems that are integrated in a car. Note, invoking *Amazon.com* in the Internet browser of the car or on a mobile device does not belong to this group. Examples are music recommendations such as proposed by Turlier et al. [THG10]. **Proactive recommender systems** (**PRS**) are recommender systems that provide recommendations in a proactive manner. The intersecting areas comprise systems that have several properties. A context-aware proactive recommender system uses context to deliver recommendations proactively, e.g., on a mobile device (e.g., Bouzeghoub et al. [BDW09]) or on a desktop (e.g., Puerta Melguizo et al. [PBD$^+$07]). Context-aware recommender systems that are used inside a vehicle exploit context to assess items that are of interest for the driver. For instance, they may provide recommendations along with a POI search. Recommender systems can also be applied without using context like an integrated product recommender for *Amazon.com*. This thesis focuses on proactive in-vehicle recommender systems (P-IVRS). These systems automatically provide recommendations to the user and are accessible from inside the car. They either use context to decide about when and what to recommend or information like user preferences. This also applies to PRS. However, PRS do not have to use context and do not have to be inside a vehicle (e.g., Lee and Brusilovsky [LB07]).

4.1.2. Use Cases

This section contains some possible use cases of P-IVRS. First, we introduce our continuous use case of a proactive gas station recommender. The rest is grouped by the trigger (context information) that is responsible for the delivery. For each use case, it is assumed that there is a user need or user preference that trigger recommendations. Item types and context vary among the use cases. There is also general context information that influences the proactive delivery in all cases, e.g., time pressure. We do not mention this information explicitly. Furthermore, user preferences based on ratings are not explicitly listed but can be utilized in all use cases.

Gas Station Recommendations

The task of refilling the gas tank concerns every driver of a car with a combustion engine. The drivers have to refill their tank before it is empty. The situations of the drivers influence the intention to approach a gas station. The current gas level, available gas stations or the circumstances of a trip determine the situations of the drivers. Furthermore, the drivers have to select a gas station among alternatives. They may either take the next available station along their route or one that they know from experience. Content provider like *Gasbuddy.com* can provide additional information about available alternatives. Drivers' choice of a gas station depends on several criteria. The location of the station determines how much detour they have to take into regard to reach the gas station. The distance to a gas station is also relevant in the decision. The further away a gas station is, the less gas remains in the tank at arrival. A low gas tank allows for large amounts to refill but is also a higher risk of becoming empty before having the possibility to refill. If gas prices are available, the drivers are able to calculate how much they save comparing alternatives. A bonus card from a specific chain or gas station may also have an influence on savings. There might also be other reasons leading the drivers to prefer specific brands. Finally, additional facilities like ATMs or the possibility to pay with credit card may also have an influence.

Car Situation Triggers

Triggers utilizing the situation of a car depend on sensors that capture the status of its components. The user need that is covered is mostly the intention to ensure that the car is working properly. This involves information about the current gas level, tire pressure, oil level and control unit failures. Today's cars deliver a warning embedded in the instrument cluster of the car with semantic information about what kind of problem occurred, e.g., in form of an icon or a text (e.g., see Hoffmann [HG09]). Recommender systems allow providing also a solution to the problem in form of a facility that is able to fix the problem such as a gas station or a service garage. This can either be done when the problem occurs or in an anticipatory way. For instance, the car status can be analyzed to detect a potential break down due to faulty components in near future or there are not enough gas stations along the route to refill. Items that help to prevent this potential problem are delivered as a recommendation. User preferences are taken into regard for features of the items and dynamic information like opening times or gas prices (e.g., available from sources like *Gasbuddy.com*, *Clever-tanken.de* or others).

Environment Situation Triggers

An example of an environment trigger is the incorporation of traffic situations into the navigational system. This information triggers a recalculation of the route of the driver because of events or congestion, called dynamic routing. The new route is either automatically applied in the system or the driver is asked to confirm it. If we regard routes as dynamic items, the use case can be extended to deliver several alternative bypasses. Another example of alternative route delivery is to compare the current route with a static route (e.g., a scenic route) and suggest the static route as item if it is suitable in the situation. In general, it is reasonable to incorporate user preferences for route attributes. These preferences can be learned from formerly selected routes (e.g., like Park et al. [PBKB07].

Every car has to be parked when the destination of a trip is reached. If no private parking opportunities are available, searching for parking places can be annoying and time consuming. Services such as *Parkinfo.com* offer information about free parking lots for some cities. With new ad-hoc communication technologies like C2X, information can also be exchanged decentralized among the drivers in an area of interest (e.g., with methods like described by Prinz et al. [PBW10]). The trigger of information delivery is the availability of free parking spots matched to the need of parking, e.g., if the driver is close to the destination. User preferences cover the type of parking place, prices and the distance to the actual destination. Furthermore, context information such as other passengers in the car and luggage also influences the proactive decision, e.g., to choose a more expensive but closer spot.

Parking is not the only way to end a trip. After a car is parked, it is sometimes necessary or convenient to switch the modality of travel to continue. For instance, in large and congested cities, park and ride offers are usually available. Route planning algorithms that take different modalities into regard already exist (e.g., Delling et al. [DPW09]). Information about timetables or pricing is usually offered by local transportation facilities, e.g., MVG in Munich. More recently, some providers offer car-sharing concepts in large cities, e.g., DriveNow, Zipcar, Flinkster or Car2go. User profiles can be built on travel history to infer preferred modalities and sequences of travel (e.g., like Zheng et al. [ZZXM09]). The users may also enter information about themselves, e.g., available long-term travel discounts. Transportation assessment also incorporates the number of travelers, traffic events or the luggage of travelers. The items are dynamic with a complex combination of modalities. A proactive recommender for travel modalities calculates several opportunities to switch the travel modality. The results are presented to the user along with the next changing point. In contrast to regular items, a travel modality plan is complex.

Recently, some providers such as *Avego.com* or *Pickuppal.com* offer services for people who do not have a car but want to use the convenience of individual mobility. The approaches can save money and energy. Their real-time ride sharing service requires a positioning-enabled mobile device, drivers who offer seats in their cars and a platform to connect the users. Geisberger et al. [GLN+10]) show how the routes of a user and a driver can be matched efficiently based on the similarity of the start or the end position or similar route sections. Supported by a P-IVRS, the drivers can share their route and offer seats to a community. People looking for a ride indicate their interest. The system recommends the drivers a set of interested people to take with them. User preferences of the drivers are the detour they are willing to take, the gender of the traveler, the reliability of the traveler in former rides and the price the traveler is willing to pay. The scenario can also be extended to offer the traveler several drivers.

Driver Situation Triggers

According to Hell [Hel04], 24% of the accidents in Germany are due to low vigilance caused by drowsiness. Car manufacturers are working on systems to recognize the fatigue state of the driver by fusing information that is captured by sensors (e.g., like Yang et al. [YLB05]). Additional sensors inside a car (e.g., camera, lane keeping systems or steering wheel) capture the state of the driver to measure eyelid closure or reaction time. A driver model infers the fatigue state from these input data. Today, this information is used to give the driver a warning. For instance, Ford [For11] suggests coffee breaks with a coffee cup that appears in the instrument cluster. Recommender systems can be used to go one step further by providing suitable solutions for the problem. The simplest way is to retrieve POIs that help the drivers to solve their problems. For instance, resting places along the route give the drivers the possibility to take a break. Recommendations can also be combined with other needs. If the drivers also need to refill their tanks, gas stations can be recommended to cover resting and refilling. Depending on the time of day, restaurants for eating or hotels for sleeping are also possible. It is important not to patronize the drivers towards what they should do. Fatigue is a sensible issue on which attention should be drawn with an unobtrusive suggestion.

A recommender system does not need to wait until the problem occurs to deliver solutions. Monotonic driving conditions contribute to make the drivers feel fatigue and loose their vigilance because they are underutilized (Chan and Atchley [CA11]). Chan and Atchley work with concurrent verbal tasks to increase driving performance. A monotonic driving situation can be classified by the steering wheel angle, the accelerator pedal usage, information about cars in front, the speed or the acceleration. Classification is not restricted to the current situation. Analyzing the route ahead (e.g., based on street type, curviness, expected stops, speed limit change) gives us the possibility to estimate whether driving will become more or less monotonic. Additionally, our circadian rhythm or body clock (Halberg and Stephens [HS59]) tells us in a 24 hour cycle when our willingness to perform is low and high. Monotonic driving situations and the circadian rhythm can be used to deliver recommendations before the driver gets fatigue. The system may recommend alternative routes with diversified driving situations to keep up vigilance, especially if the circadian rhythm indicates low activity. However, alternative routes should also incorporate driver preferences and general route planning criteria such as driving time to avoid irrelevant routes.

4.2. Requirements on a P-IVRS

Chapter 2 already discussed requirements on information systems from the area of automotive information systems, proactive assistance and recommender systems. The last Chapter 3 shows which characteristics of proactive information systems researchers give high importance. We select the requirements on a P-IVRS from this review of literature. The discussion comprises non-functional requirements that describe restrictions to an in-car information system and characteristics of proactive information systems. Functional requirements involve aspects of a system design covering intelligent behavior, situation awareness, recommender systems and comprehensibility. Our list of requirements does not claim to be complete. It covers necessary requirements but leaves out aspects like privacy or trust.

4.2.1. Non-functional Requirements

Non-functional requirements are the conditions that determine system design in general. A P-IVRS should avoid driver distraction and information overload. To make a user accept the system, it should be easy to use and useful.

Information Overload and Driver Distraction

Requirement 1 ***Information overload:*** *Proactive recommendations should be processable with as little as possible cognitive load.*

Nowadays, drivers face large amounts of information that might be of interest for them (e.g., restaurants, gas stations, resting places, opening times, gas prices, etc.). Broadband mobile Internet makes this information available during driving. Processing of this kind of information is a knowledge-based behavior (according to Rasmussen's model [Ras83]). It causes more cognitive load than skill-based or rule-based behavior. Therefore it should be as condensed as possible.

Requirement 2 ***Visual driver distraction:*** *Displaying of proactive recommendations should as little as possible turn the glance of the driver away from the street.*

Furthermore, processing of such kind of information involves visual interaction of the driver with an information system. Driving also occupies the visual sense of humans. According to Wickens [Wic84], the risk of interfering the driving task is higher if the same human resource is addressed. Hence, such kind of information may lower driving task performance. The driving task is the primary task, whereas consumption of interesting information is only a tertiary task (Bubb [Bub03]). It is of minor importance. Such kind of information is usually integrated into the CID. Other output channels like the instrument cluster and the head-up display are generally not targeted to display information that involves intensive knowledge-based processing (comparable to planning in navigation systems). Other output channels like acoustic or haptic output provide only low or moderate amount of information (Hoffmann and Gayko [HG09]). Speech dialogs might be a solution but are out of scope of this work. As the CID is in the center console of the car, the drivers need to turn their glance from the street to the CID to consume such kind of information.

Requirement 3 ***Physical driver distraction***: *Interaction with a P-IVRS should be reduced as far as possible.*

Consuming information mostly involves interaction of the driver with the system, e.g., to browse through presented results or to retrieve additional information. Interaction causes visual as well as physical distraction.

Ease of Use and Usefulness

Requirement 4 ***Ease of Use***: *A P-IVRS should be easy to use for the driver.*

Avoiding information overload and driver distraction are conditions for a successful system design. However, it does not address if the drivers accept such a system. The technology acceptance model (TAM) (Davis [Dav89]) provides a common method to evaluate acceptance of information systems. According to the model, the main indicator of user acceptance is perceived ease of use and usefulness of a system. Ease of use of a P-IVRS corresponds to how much effort the driver needs to get the desired items.

Requirement 5 ***Unobtrusiveness***: *The behavior of a P-IVRS should be unobtrusive.*

Variables for ease of use for proactive recommender systems can be derived from research about the system design of a proactive information system (e.g., Billsus et al. [BHMA05] and Rhodes [Rho00]). A central requirement is unobtrusive behavior. Unobtrusive behavior comprises several aspects inside a car. Because human resources are differently occupied during driving (dense traffic vs. standing still at a traffic light), the recommender should not require immediate attention to process the delivery. The drivers should have the possibility to postpone their reaction on a recommendation and to choose the right time to process information. When items are delivered, the drivers should have left enough time to process the alternatives and to make a decision. Time pressure causes cognitive load and may lead to driver distraction. This also applies to frequent interruptions by the recommender. These aspects contribute to an unobtrusive behavior of the system.

Requirement 6 ***Accessibility***: *Relevant information should be found easily by the driver and should be comprehensible.*

For their final choice, the users need information to make a decision. In the best case, required information is delivered right with the recommendation. However, too much information causes information overload. If for some reasons information is missing, the driver should know how to retrieve it with little interaction. Furthermore, the information that is provided should be comprehensible for the driver. This applies to text as well as to icons.

Requirement 7 ***Transparency****: The behavior of a P-IVRS should be transparent to the user in the context of the automotive environment.*

In contrast to classical recommender systems, a P-IVRS chooses itself what to deliver to the driver. The system makes decisions based on information and assumptions that the user does not know. In case the decision is not obvious, i.e., the user does not know why the decision has been made, a good recommendation may not be recognized as such. As the users do not request information themselves, they have to understand the intention of the system to assess whether the recommended items are useful for them.

Requirement 8 ***Usefulness****: A P-IVRS should be useful for the driver.*

The second criterion for user acceptance is the usefulness of the system. The drivers should perceive the recommender as useful for any reason. The system may assist them by their tasks or is fun. In case of recommender systems, usefulness comprises to deliver relevant items in drivers' situation. It defines the quality of the recommended items. Rhodes [Rho00] emphasizes the importance of usefulness in case of proactive information systems.

4.2.2. Functional Requirements

Functional requirements are derived from literature review of related approaches on the desktop and for mobile devices. They describe what features a mobile proactive recommender should have to be successful.

Intelligent Proactive Behavior

Requirement 9 ***Technical competence****: A P-IVRS should be able to work out tasks on its own.*

Information overload causing driver distraction is the overall problem we address in this thesis. We focus on proactive recommender systems as a solution for information overload. This involves intelligent filtering of information (Second level of the in-car information flow according to Ablassmeier [Abl09]). The idea of proactivity is to delegate tasks like filtering to an autonomous system. The driver either supervises the task or processes the results (Tennehouse [Ten00]).

Requirement 10 ***Discretionary competence****: A P-IVRS should be able to make decisions proactively towards an action to perform.*

To work out tasks, the system has competences. The competences of a P-IVRS might be the assessment of items, the calculation of trade-offs between alternative items, the collection of information or the initiation of a dialog with the driver. Knowledge that results from the competence of the system corresponds to a better understanding of the situation of the user. The system processes this knowledge to make a decision about what action to perform.

Requirement 11 ***Execution competence****: A P-IVRS should be able to execute the actions it decides for.*

In case of proactive information systems, decisions involve which items to select or when to deliver information. The competence of the system to make decisions relies on context information, information about the driver and on data provided by content provider, e.g., gas stations, hotels or traffic information. A decision leads to actions performed by the system. Actions for an information system are to deliver information and to communicate with a user.

Requirement 12 ***Decision making****: The decision making of a P-IVRS should take into account human decision making methods.*

Decisions that a system makes are subject to uncertainty because underlying information sources can be uncertain and incomplete. The decision making of the system is based on the knowledge that the system has of the situation of the user. To derive this knowledge, several heterogeneous information sources need to be aggregated. A proactive recommender either makes decisions on its own or they are made collaboratively in a mixed initiative between the driver and the system. People use non-comprehensive strategies (heuristics) to cope with complex decisions balancing accuracy and effort (Payne et al. [PBJ93]). To assist a driver properly, a proactive system should also take human decision making into regard.

Requirement 13 ***Choice****: A P-IVRS should provide not too much and not too little choice.*

Although a proactive recommender makes decisions on its own, it does not work completely autonomous. The final decision for an item is still made by the user. To reduce decision complexity and information overload, only a few items should be provided to the driver. Too much choice may decrease decision quality and causes information overload (Iyengar and Lepper [IL00]). However, it is better to provide choice than providing no choice at all.

Situation-Aware Information Need

Requirement 14 ***Situation awareness****: A P-IVRS should be situation-aware.*

To make intelligent decisions for a user, a system should be aware of the situations of a user. According to Endsley [End00], situation awareness comprises the perception of elements in the current situation, comprehension of the situation and a projection into the future. This matches to the behavior of drivers. Drivers preview potential traffic situations and predict how they have to behave in that situation.

Requirement 15 ***Uncertainty***: *The situation awareness of a P-IVRS should be able to handle uncertainty.*

To enable situation awareness in a system, it needs sensors to perceive context information, models to comprehend the situation and models to predict future situations. Sensors and models are generally subject to uncertainty and have to cope with incomplete information.

Requirement 16 ***Situation-aware benefit***: *A P-IVRS should pay attention to the situation of the driver to assess the benefit of a recommendation.*

The relevance of information differs, depending on the situation of the user. The user of a desktop information system needs other information than the driver of a car. Situations determine which kind of information is beneficial for the drivers and when they need information.

Requirement 17 ***Situation-aware user preferences***: *A P-IVRS should pay attention to drivers' situations to be able to assess what they prefer in these situations.*

User preferences are a central part of recommender systems. They change slowly over time in classical recommender systems. However, drivers are embedded in a dynamic physical world. Not only the information need but also user preferences are influenced by the situations of the drivers.

Requirement 18 ***Task awareness***: *A P-IVRS should have a deep understanding of the task it provides information for.*

Presented information fills the gap between available and required information to perform a task. The interaction with the recommender itself is a tertiary task but items support tasks (e.g., parking or refilling) that require physical involvement, e.g., driving to a POI. Only items that are filtered according to the task of the user are useful for task completion.

Selection of Useful Items

Requirement 19 ***Context awareness***: *A P-IVRS should take the context of the user into regard when selecting items.*

Knowing that information delivery is beneficial for the users does not involve that the system knows which items are useful. Among all available items, those should be presented to the users that are most useful in their situations. Items of interest for the drivers share common features, e.g., location, opening times, price, etc. Situation-aware user preferences determine what is important for the drivers in their situations. The context of the user determines which items are not just relevant but also useful for the users (Rhodes [Rho00]).

Requirement 20 ***Cold start****: A P-IVRS should avoid cold start problems for new users.*

One challenge of recommender systems is to gather enough information from a user to make recommendations (new user problem). A P-IVRS appears without explicit user request. This is always an intrusion to the current task of the user. Learning of user preferences requires presenting potentially irrelevant items in the beginning of using the system. However, useless information may confuse the driver. Furthermore, learning of user preferences by analyzing interaction is difficult in automotive scenarios because more interaction with the system causes driver distraction. Finally, relevant information for the drivers requires physical involvement, e.g., driving to a POI. The users are less tolerant towards irrelevant items in this case.

Requirement 21 ***Some good items****: A P-IVRS should be able to find some good items.*

The recommender is only able to provide a small set of items in an automotive scenario. Too many items cause information overload. Furthermore, browsing through many items also causes driver distraction through interaction. Therefore, the purpose of the recommender should be to find some "good" items. Good items cover useful as well as relevant items. A proactive recommender should always deliver useful items that could be relevant. Relevant items that are useless should never be delivered proactively. For instance, restaurants that are located in New York and match user preferences (relevant) should not be delivered to a driver in Munich (useless). In contrast, a close restaurant that does not meet user preferences (irrelevant) can be delivered if the user is hungry and it is the only restaurant available (useful).

Comprehensible System Behavior

Requirement 22 ***Justification****: A P-IVRS should shortly justify its decisions to support the user in decision making.*

Comprehensibility of system behavior is an important characteristic of intelligent proactive systems. We investigate explicit explanations to make system behavior comprehensible. However, the automotive environment is restricted in interaction and information consumption. Providing too much additional information causes driver distraction. The driver finally chooses an item out of some alternatives. A choice is based on information to distinguish alternatives. Furthermore, people want to have reasons to make a decision, even if it is a gut feeling (Shafir [SST93]). These aspects require short explanations to justify the behavior of the system.

4.3. System Design of a P-IVRS

The system design should reflect functional requirements on a P-IVRS and respect non-functional requirements. We described four groups of functional requirements (intelligent proactive behavior, situation-aware information need, selection of useful items and comprehensible system behavior). These requirements are mapped to components of a system. We distinguish two aspects in system design. First, knowledge that is processed by a P-IVRS needs to be represented in the system. It comprises context, situations, user preferences, tasks, items and a prediction horizon. Knowledge should be processable by the components of the system. Second, system components are processing units for knowledge and they communicate with the user. System components comprise decision making, evaluating items with a recommender, generating explanations and a human machine interface (HMI). Decision making is a central part of the system.

4.3.1. Knowledge Representation

The capabilities of the system depend on the knowledge it has about the users and their situation. We describe what kind of knowledge our proposed P-IVRS uses and how it is represented inside the system.

Context

Context is required to cover situation-aware information need (Requirements 14, 16 and 17) and context awareness of the recommender system (Requirement 19). For proactive recommendation systems, we distinguish between context C_P influencing the decision about when and whether to recommend and C_I for item assessment. These sets of context do not have to be disjoint: $C_P \cap C_I \geq \emptyset$. There is context information that is taken into account in both system components. For instance, the time of day may determine when to recommend a restaurant but it also plays a role in item assessment as the restaurant may be closed at this time. Context information is determined by a set of parameters $c(timestamp, type, value, unit) \in C$. The *timestamp* indicates when context information is received by the system. This can be the time when a sensor delivers information, the driver enters data or data is received via digital communication. The *type* is an identifier for the affiliation of information to an information group. Context information with the same *type* describes the same fact independent from the unit of its value. The *value* is actual context information. Its type ranges from simple numbers or strings to complex objects. The *unit* indicates what kind of type the value has, e.g., kilometer, cent or liter for numbers or complex for an object structure. The *type* and the *unit* determine what kind of *value* is stored in the context information to allow applications to process information properly.

Situations

A situation and its state is a central part of situation-aware benefit (Requirement 16) and user preferences (Requirement 17) to allow a situation-aware behavior (Requirement 14). We use fuzzy logic as modeling technique for situations to incorporate uncertainty and incompleteness (Requirement 15). Smooth transitions between values in a fuzzy set allow modeling a system without abrupt behavior changes. Abrupt behavior changes should be avoided in proactive recommendation systems. Fuzzy logic also allows for comprehensible behavior of a system by using linguistic terms (Requirement 22). Comprehensibility may increase the acceptance of a proactive recommendation system. Furthermore, people usually perceive situations fuzzy and describe them in terms with fuzzy transitions.

According to our understanding of situation in Section 2.2, a situation is the interpretation of context information by the users, e.g., people or systems. A relevant set of context information implies a situation: $C^* \subseteq C \Rightarrow S$. The mapping is associated with perception uncertainty, i.e., not every user has exactly the same interpretation in mind. For example, the temperature of 15 °C is perceived by some people as cold and by others as mild. We use fuzzy sets to model perception uncertainty by means of linguistic terms. A situation is a fuzzy variable $S = s_1, \ldots, s_n$ with n terms called situation states s. An involvement to a situation results from fuzzification of a crisp value, e.g., a context information, or inference on the situation. The same context information can be interpreted differently in different situations. Furthermore, a degree of involvement determines our belief in which of the possible situation states the user is. The involvement is represented by means of the membership function $\mu_S(s)$. To keep the situation description generic for statistical and fuzzy logic inference models, we impose constraints on the membership function:

- $\sum_{i=1}^{n} \mu_S(s_i) = 1$: All observed or inferred memberships to situation states have to sum up to 1. This implies that situations have to contain a state for no involvement if necessary.

- $\forall s \in S\, \exists \mu_S(s) = 1$: Every state has to allow a maximum membership of 1 (normal fuzzy set) to express that we are absolutely sure about a state.

- $\forall s_i, s_j \in S \wedge \mu_S(s_i) \in [0,1]\, \exists\, \mu_S(s_j) = 1 - \mu_S(s_i)$ if $s_i \cap s_j \neq \emptyset$: If two states s_i and s_j are neighbors in S and share involvement, then for every involvement of s_i there need to be a complement in s_j. This only holds for membership functions that are not singletons.

A situation has a current state s_c that represents the situation of a user with exactly one term. A change of state from s_c to another state s_n is defined in Equation 4.1.

$$s_c = \begin{cases} s_n & \exists s_n : \mu_S(s_c) < \alpha \times \ \mu_S(s_n) \\ s_c & \text{otherwise} \end{cases} \tag{4.1}$$

The current state s_c changes to another state s_n if its membership is higher in respect to a situation stability $\alpha \in [0, 1]$. The stability ensures safe transitions between situation states without jumps at the border where $\mu_S(s_c) \approx \mu_S(s_n)$. Stable state changes address fluctuation problems of fuzzy systems, which can be a problem in proactive systems. It prevents the system from changing its behavior frequently if a sensor measurement varies around a value.

Horizon

Models of human situation awareness (Requirement 14) involve anticipation and prediction of future situations. Theoretically, this involves an infinite time range. To make calculation feasible, we limit the time range to a so-called horizon. A perfect decision could be made if all future situation states are predicable but in practice they are not. Hence, we define a boundary for proactive decisions. The boundary comprises all items and context information that are taken into consideration. The boundaries of the horizon $H(TA)$ are determined by time $T = [t_{-m}, ..., t_0, ..., t_{+n}]$ and space $SP = [sp_{-m}, ..., sp_0, ..., sp_{+n}]$ with discrete steps. The index of 0 represents the current context s of the user in the horizon. The definition does not restrict the dimension of the space. It can be 1-, 2- or 3- dimensional. The range from 0 to n determines the output of a recommendation, i.e., recommendations are delivered ahead of the current context. The horizon depends on the task TA. For gas station or parking recommendations, the horizon can be the route of the driver. A whole day can be suitable for a restaurant recommendation and a document or a section for article recommendations. The selection of a horizon depends on the scenario as well as on the capabilities of a system to predict. The users of the system are assumed to be in each context p of H. They are either in a context currently, have been in a context recently or will in a context soon.

Tasks and Items

Our requirements on a P-IVRS involve a deep understanding of the task (Requirement 18). The items that the recommender provides should be useful for a specific task. Tasks TA are user activities we want to deliver recommendations for. A task is determined by the parameters $ta(type, benefit(c), items) \in TA$. The *type* is a unique indicator, e.g., eating, parking, sleeping, etc. A specific task is only represented once in the system. Parallel tasks have to have different types. The $benefit$ of a task is a real value in $[0, 1]$ which indicates whether a recommendation would be beneficial for that task. Similar to situations, a user can be involved in a task for a period of time. Therefore, we represent benefits as a function over the horizon with context p. For instance, if it is 12 PM, recommendations for restaurants may be more beneficial than at 3 AM. To each task, categories of *items* are associated. They determine which items are assessed by the recommender if a recommendation should be delivered. Note, we do not regard the involvement of a user in a task directly but how beneficial a recommendation is for a task, i.e., how much it covers information need. Hence, we do not need to infer on the involvement in a task. Our recommendations do not support the primary task of a driver (driving).

User Preferences

To be able to make useful recommendations (Requirement 8), a system has to know what the drivers prefer and how this changes among different situations (Requirement 17). User preferences are a major component of recommender systems. In our proactive recommender, there are two user models. They personalize the decisions of the system about whether and when to recommend and what to recommend. The user model of the first decision determines the benefit of a recommendation for a task TA. The structure of the model depends on the underlying inference model to make that decision. For instance, in a Bayesian network, the user model is represented by probabilities and in a neural network as weights. For the second decision, the user model is represented as the knowledge the system has about the interests of a user for item features. To address the requirement of avoiding cold start problems (Requirement 20), we apply knowledge-based representation of user preferences. Knowledge-based user preferences are represented by weights $U =< w_1, w_2, ..., w_n >$ for specific features of an item. We do not determine how the weights are established for each user, e.g., from past user actions or explicitly given by the user. They reflect long-term preferences but can be adapted at run-time (short-term) to the current situation (Requirement 17). Our representation is not limited to knowledge-based recommendation approaches. Collaborative or content-based user preferences can be represented as feature of the item.

4.3.2. System Components

A functional requirement on a P-IVRS is intelligent behavior to make decisions. First, the system should have technical competence (Requirement 9) to work out tasks on its own. Second, discretionary competence (Requirement 10) is required to be able to make a decision towards an action. Third, execution competence (Requirement 11) allows the system to perform the action. Discretionary competence is the central part of the system. It brings technical and execution competence together.

Our proposed system distributes competences over several system components that process knowledge (Figure 4.2). The explainer, the recommender system, the situation manager and the user profile manager have technical competence. The system assigns tasks to them. They work on the task until it is finished and return the result. Technical components are stateless. They either offer interfaces with their competence, e.g., the recommender system is able to assess items, or they run periodically, e.g., the situation manager. The decision process has discretionary competence to decide and execution competence to delegate tasks. It has an internal state and communicates with the situation and context manager asynchronously, i.e., the components push information to

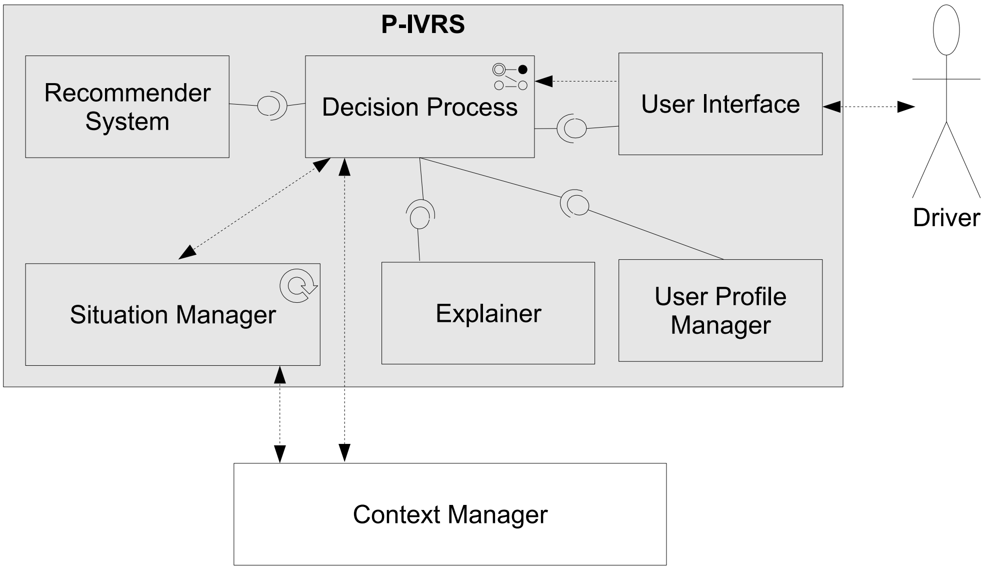


Figure 4.2.: System Components of a P-IVRS

the decision process from time to time. The context manager delivers information, but it is outside the P-IVRS design. It does not cover any specific competence to contribute to the recommender. It is rather a generic information distribution competent.

Context Manager

The design of our context management follows the proposal of Dey et al. [DAS01] for a framework for context-aware applications. It suggests a separation of concerns between the application, i.e., the user of context, and the context management. It allows an extendable and flexible management of context by reusing the same semantic notation of context in several applications or components of the applications. For our application, we can be sure that each decision is made on the same knowledge base. The context manager is on the middleware layer (for layer structure see Hong et al. [HSK09]) between application layer and network infrastructure where the communication to other systems and sensory systems is located. It is a centralized component inside the car which appears as singleton. It is common for context management middleware components to operate according to the publish-subscribe pattern (e.g., Chen and Kotz [CK02]). Consumers of context subscribe with the type of context and other parameters (e.g., frequency of delivery) to the middleware. Information components publish their context information and the middleware notifies all consumers. The middleware operates as broker between context and consumer (like described by Chen et al. [CFJ03]). The advantage is that applications do not have to make resource intensive polling but get context information pushed asynchronously.

Situation Manager

The technical competence of the situation manager is to supervise situations. The decision process controls the situation manager in case a task (Requirement 18) is active. The situation manager also includes a situation awareness subcomponent. A timer triggers the processing of data because it might be computational intensive. Comparable to human situation awareness, we perceive, interpret, predict and resolve information towards a task (Requirement 14). This information is transferred to the decision process. Hereby, uncertainty of information is considered (Requirement 15). The result is either the benefit of a recommendation (Requirement 16) or user preferences (Requirement 17). Situation awareness is only executed if context information changes. The second subcomponent monitors current situations. It compares the beliefs of the system with the actual situation state and notifies the decision process if beliefs and the observations do not match. This way, a wrong decision may be corrected. For instance, if the decision process believes that there are enough reachable gas stations and the monitor recognizes that only a few are available, it informs the decision process. The decision process is also notified in case of situation state changes.

Decision Process

The decision process is the central component that controls the behavior of the system. It has discretionary competence to make a decision and has execution competence to delegate tasks to other components. The decision capability is aligned to human decision making (Requirement 12). It makes decisions about whether a recommendation should be calculated, what items should be delivered and when to deliver those items. Requirements on the selected items are that they should provide choice (Requirement 13) but only a few should be presented (Requirement 21). This requires to delegate subtasks to components over interfaces, e.g., to assess items. The decision making processes data when the situation monitor notifies about the situation of a user. Notifications may result in changes of internal states. State transitions are either based on system or user decisions (represented by user input through the user interface).

Figure 4.3 shows the internal states of the process and how they change. The description follows the $Event[Condition]/Action$ notation. If an event occurs asynchronously, outgoing transitions are examined. If the conditions of a transition are fulfilled, an action is executed and the system changes its state. As soon as the system is activated for a task, it goes to the state IDLE. After a period of time, the system checks whether context changed and if so it starts to "think" (DECISION MAKING) about what to do in the situation. If the system decides to deliver a recommendation, then it delegates subtasks, e.g., assessing items or explaining, to subcomponents and waits for the result (WAITING). Based on the results, the system decides what items to recommend and when to deliver the recommendation. After the items and the situations of delivery are decided, the system waits for the situations to occur (PREPARING). If the prediction

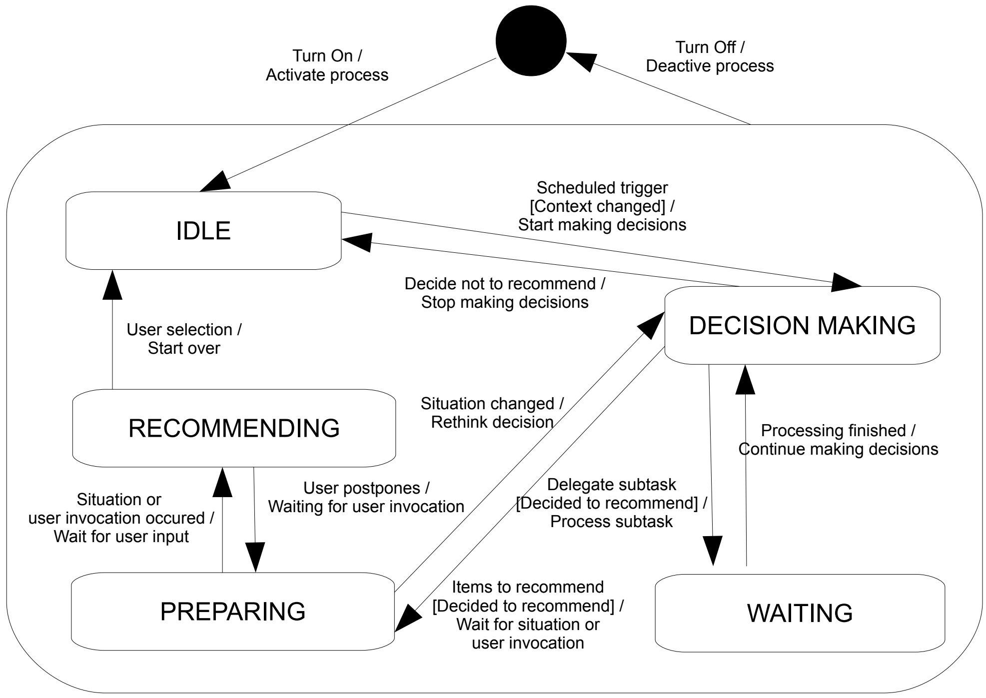


Figure 4.3.: Internal states of the decision process

of a situation changes while preparing for a delivery, the system "rethinks" its decision (DECISION MAKING). Rethinking corresponds to deciding about the delivery and the items again because predicted assumptions changed. Finally, if the situation that causes a delivery occurs or the driver invokes the recommender manually, the items are presented to the driver (RECOMMENDING). During interaction, the driver either selects an item or postpones the decision. If selection occurs, the system enters the state IDLE again. Otherwise, it waits until the user invokes the recommender again (PREPARING). The system can be turned off at any time.

Recommender System

The functional competence of the recommender system is assessing items based on knowledge about the items, context and situational user preferences (Requirement 17). Context (Requirement 19) and user preferences are integrated in the assessment. The recommender works without cold start problems (Requirement 20) by exploiting different features of an item. Items are represented by features. User preferences are represented by weights for these features. The recommender returns a set of relevant items with an assessment for each item.

Explainer

Another component with functional competence is the explainer. Its task is to build explicit explanations for system decisions. This should make the behavior of the system more transparent. If the decision concerns recommending, the explainer assesses the situations according to their contribution to the decision. If the decision process selects a set of items, the explainer determines arguments for the items. Resulting explanations justify why the system made this decisions (Requirement 22). The component returns a set of arguments for recommendation decisions as well as items.

User Profile Manager

The functional competence of the user profile manager is to organize user preferences in the system. The component is required because the system needs to know general preferences to adapt them to the situation. User preferences are recorded over the user interface and provided to the situation manager. User preferences are seen as context of the user. The situation manager uses them for situation adaptation (Requirement 17). The recommender system needs user preferences to assess items. In depth investigation of user profile management is out of scope of this thesis.

User Interface

The user interface is the channel to the driver. It has execution competence towards actions that are performed with the driver in mixed initiative dialogs. This comprises displaying of information and reacting on input. It plays an important role for user acceptance by being easy to use (Requirement 4) with an unobtrusive (Requirement 5) and accessible (Requirement 6) behavior.

4.3.3. Decision Making

Decision making is the central component that controls the behavior of the system. The first decision it makes is whether recommendations should be searched proactively. If this is the case, then we need to check which items are available and when they should be delivered. We use the satisficing (SAT) strategy to make both decisions. This is a common strategy in human decision making (Requirement 12) for such decisions. The application of satisficing for a proactive recommendation decision can be motivated by an example of Goodrich et al. [GSB99]. The authors describe a driver who is behind another car and waits for a moment that is good enough to pass the slower car. This behavior represents a trade-off between a solution that is good enough and a possible solution that follows in the future and is better. In the case of overtaking, additional lanes may be added to the street later. The additional lanes allow for a more secure overtaking. Applied to our problem, a satisficing strategy delivers items if an upcoming situation is good enough, i.e., the benefit for a recommendation is high enough.

Whether to Recommend

The decision whether recommendations are relevant at all is a binary SAT decision. The system decides whether a situation is good enough to search for recommendations. The decision should be stable, i.e., it should not be subject to minor fluctuations in the model.

$$D(t) = \begin{cases} \text{recommend} & B(t) \geq (1.0 + \alpha) \times B_{threshold} \\ \text{not recommend} & B(t) < (1.0 - \alpha) \times B_{threshold} \\ D(t_-) & \text{otherwise} \end{cases} \tag{4.2}$$

In our safe Equation 4.2, we make a decision D at time t based on the last stable decision in the past $D(t_-)$. A security value α ensures that the system is less susceptible to minor fluctuations around the threshold $B_{threshold}$. We take the last decision $D(t_-)$ in case only minor changes occur. Otherwise, we decide to recommend in case of benefits above $B_{threshold}$ and not to recommend below.

What and When to Recommend

The second SAT decision involves the selection of items inside the horizon and to decide when to perform an action with them. In case of recommending, actions comprise to deliver items to the users or to notify them. The first part of the decision is when to deliver a recommendation. The decision problem involves the analysis of the situations of a user and to select exactly one context $c \in H$ out of the decision horizon to deliver the recommendation. Its goal is to make the recommendation useful. The second part of the decision problem is about what to recommend. The system selects a set of items $se^* \in SE$ out of all assessed items I^* and all of their combinations in a set SE. The goal is to select a set se^* with useful items. The challenge is to select useful items for a delivery at context c. Our SAT strategy combines both parts of the decision. We take the next upcoming context c that has enough benefit B to deliver a recommendation (good enough). The recommender system optimizes the items with a comprehensive assessment (usefulness). We try to recommend the most useful (optimized) items in a situation that is good enough for recommending. We call this an **optimizing-satisficing** (**OS**) strategy.

$$\begin{aligned} \underset{a \in A^*}{\text{maximise}} \quad & D_{A^*}(a) \\ \text{subject to} \quad & f(i_l, i_j) = 1 \; \forall \, f \in CON, \; l = 1, \ldots, n \wedge j = 1, \ldots, m \wedge i, j \in I. \end{aligned} \tag{4.3}$$

More formally, we select the satisficing solution a in all satisficing solutions A^* which is a subset of all alternatives $A^* \subseteq A$. The result is a optimization problem (Equation 4.3). $D_{A^*}(a)$ is the decision function that specifies the membership of a to the set of satisficing solutions. A solution a contains exactly one set of items se^*. Each solution has an expected utility eu for the user (Equation 4.4).

$$D_{A^*} = \begin{cases} eu & a \in A^* \\ 0 & a \notin A^* \end{cases} \tag{4.4}$$

If an item i in a satisficing solution a cannot be delivered in a satisficing situation, then the expected utility eu of a is 0. Otherwise, we optimize a by means of the assessments of the recommender. Furthermore, the items i_l and i_j in a set of items se are subject to constraint functions CON. Constraints regulate whether an item is included in an item set se based on the assessments of the recommender. For instance, two items that are both useful but very similar should not be in the same set. We set $f = 1$ if the constraint is satisfied, otherwise 0. Constraints can be a Pareto dominance condition, the similarity/diversity of items or the number of items in a set. The expected utility is calculated with a function g over the items in the set ($eu = g(se^*)$). Appendix A describes implementation details of our optimizing-satisficing (OS) strategy for item and context selection.

4.4. Summary

In this chapter, we gave a definition of proactive in-vehicle recommender systems (P-IVRS) to distinguish the term from similar terms for recommender systems. As P-IVRS are a new field of research, we described potential use cases. One of the use cases is a proactive gas station recommender. We derived non-functional and functional requirements from research and related work in recommender systems, automotive, decision theory and proactive assistance. Non-functional requirements comprise the avoidance of driver distraction and information overload and to make the system easy to use and useful. Ease of use is enabled by an unobtrusive and accessible behavior of the system. Functional requirements are grouped by intelligent proactive behavior, situation-aware information need, selection of useful items and comprehensibility. The requirements are mapped to aspects of our proposed P-IVRS system. The system design of this P-IVRS comprises the representation of knowledge and system components. The decision making component is the most important one. The system decides whether to recommend based on the benefit of a recommendation. Then, it takes the results from an item assessment (recommender system) and the benefit of a recommendation (situation awareness) to decide which items to deliver and in which situations to deliver them. We use a satisficing strategy to make decisions. It is combined with the optimization of items by the recommender system. We call the strategy optimizing-satisficing (OS). Our OS strategy assumes that the quality of items is more important than the situation in which the items are delivered to the driver. Furthermore, the final decision of the driver is about the item not about the context. In the following chapters, we take a deeper look into the functional components and how they fulfill the requirements.

5. Situation Awareness in P-IVRS

A fundamental part of a P-IVRS is the situation manager. Its task is to interpret contextual information towards the recommendation task. Generally, context information is independent from a specific kind of recommendation. A recommendation task contains decision making by the system and the user. Situation awareness bridges the gap between contextual information and decision making. It fulfills the requirement to pay attention to the situation of the user in order to enable intelligent proactive behavior. Human situation awareness models already exist. We describe a computational situation awareness framework based on the model of Endsley [End00]. Our framework is tailored to proactive recommendations. The input for the framework comprises user situations. The output comprises preferences that the user has in these situations and the benefit of a recommendation for a specific user task, e.g., refilling or eating. We implement the framework with fuzzy logic and Bayesian networks to investigate its feasibility for the use case of a gas station recommender.

5.1. A Generic Framework for Situation Awareness

We discussed basic concepts of situation awareness in Section 2.2.2. A lot of work is already done in human situation awareness and high-level information fusion for computational situation awareness. Our goal is neither to extend this work nor to propose a new general model for situation awareness. We take previous findings and apply them to a model tailored to proactively delivered recommendations. Situation awareness is crucial in the assessment of benefit, costs and dynamic user preferences of a recommendation in the situation of a user. Benefit and costs are used by our optimizing-satisficing (OS) strategy to find good enough situations to recommend. The recommender system needs the situational preferences of the user to select useful items for the user. We propose a situation awareness model that comprises three basic levels according to Endsley's [End00] model and a forth level as an extension by McGuinness [MF00]. We already published the general idea of mapping Endsley's model to fuzzy variables and introducing temporal prediction in [BWP10]. Our abstract model is depicted in Figure 5.1.

Each level in the model is applied to a specific task. Components of the levels, e.g., a prediction method, can be reused for other tasks as well. However, the levels itself are task specific, e.g., situations are perceived and comprehended towards a task. The model does not comprise the relationship to other tasks explicitly. On the lowest level, context

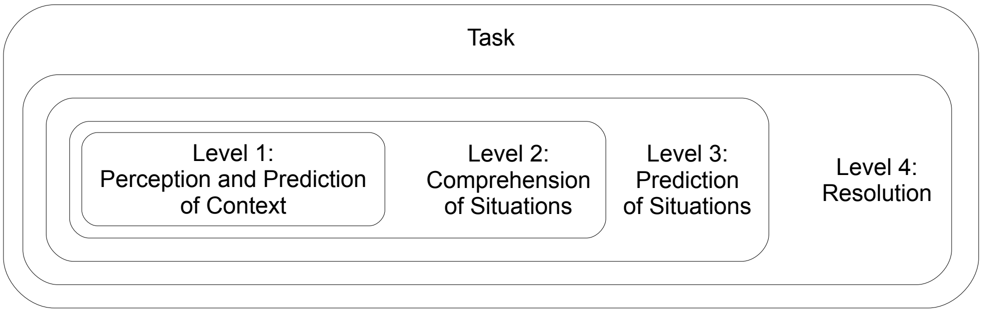


Figure 5.1.: A four level situation awareness model for proactive recommendations

parameters are captured by sensors or are derived based on captured data. This refers to lower level information fusion. The context parameters are interpreted as fuzzy situation variables (see Section 4.3). The next level takes the situations and interprets them towards the task, i.e., we infer on high-level knowledge about recommendations. Both, the comprehension of individual situations as well as their relationship is addressed. On level 3, situations are predicted inside the horizon H. Two ways of prediction are possible: Either the prediction of context on level 1 or the prediction of situations on level 2. The resulting predicted situations are again interpreted. Then, we now in which situations the user is going to be and comprehend what these situations mean for providing recommendations. On level 4, interpreted and predicted situations are resolved towards task specific information. In our case, this is the benefit of a proactive recommendation and situational user preferences.

5.1.1. Perception Level

The first level of situation awareness transfers crisp low-level data to fuzzy variables. Situations are represented by fuzzy variables (Section 4.3). Hence, we first need to establish fuzzy variables for all situations that are relevant for a task. This requires a set of discrete states for each situation. However, some context parameters that we want to interpret as situation are rather continuous (e.g., the gas level). Entropy-based discretization (Fayyad et al. [FI93]) is a suitable method for discretization in our case. It reflects the interpretation of a situation towards high-level knowledge concerning a task. Hence, it is a perception towards the task. A detailed description of entropy-based discretization can be found in Appendix B.5.

To illustrate situation perception, we take the current gas level of a car as an example. In one application, the current gas level could be interpreted with equally distributed states such as "empty", "low", "medium", "high" and "full". For an application like a gas station recommender, the interpretation could be different because drivers usually do not refill if the current gas level is higher than the half. Therefore, we can merge the states "medium", "high" and "full" and call it for example "enough gas".

The perception of situations is not finished with discretization. Fuzzy variables consist of membership functions for each discrete state. Our method to calculate memberships is described in Appendix B.1 in detail. It incorporates the restrictions on the membership function that we defined in Section 4.3.

5.1.2. Comprehension Level

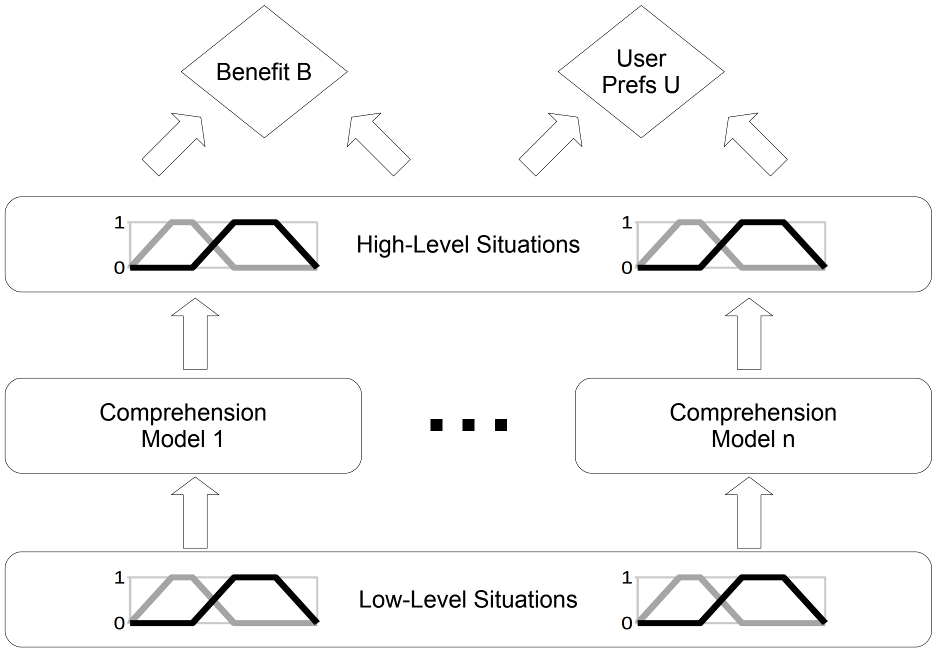
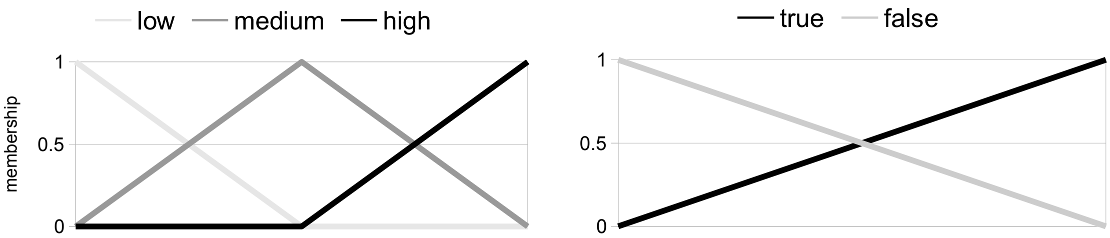


Figure 5.2.: Comprehending situations low-level as well as high-level

On level 2 of the framework, we deal with the comprehension of situations towards a task. Comprehension means to understand which impact perceived situations have on proactive recommendations for a task. Figure 5.2 shows our two-stage comprehension approach. The lower stage represents single situations after the situation perception process on level 1. A comprehension model is responsible for interpreting these situations according to their influence on the decisions of a user towards recommendations. This comprises the usefulness of a recommendation (Benefit B) and user preferences U. Costs can also be modeled the same way but are not scope of this work. To receive B and U, low-level situations are transferred to high-level situations with comprehension models. Relationships between situations are included in the comprehension model. We introduce specific comprehension models for user preferences and benefit in Section 5.2.

High-level situations are resolved to B and U. To describe the benefit B of a recommendation, we define a high-level situation variable "Proactive Recommendation" PR with the states "Yes", "No", "Later" and "Too Late". PR describes if a recommendation should be given in a situation ("Yes") or not ("No"). We add a time dimension because situation awareness also contains predicted situations. A recommendation may also be given later ("Later") or it is already too late ("Too Late"). Recommendations are relevant later, if the users already know that they need information but not in their current situation. For instance, we already know in the morning that we want to have lunch later. If a recommendation is relevant but the system delivers items after the user wants to consume them, we call the recommendation too late. For instance, the system delivers restaurants after lunch. The terms are valid inside a horizon H. Hence, a recommendation cannot be too late in the beginning of a horizon and not be given later at the end of the horizon. Comprehension models infer on the membership to the states of the high-level situation "Proactive Recommendation" PR and other high-level situations describing user preferences.



(a) Three-state fuzzy variable (b) Two-state fuzzy variable

Figure 5.3.: Fuzzy variables to describe benefit B and user preferences U

The benefit B itself is either described by a crisp value in $[0, 1]$ or as fuzzy variable like in Figure 5.3. High-level user preferences are a vector $U = w_1, \ldots, w_n$ with n weights for features of items. Each weight is represented by exactly one high-level situation, e.g., "gas station price" or "detour for a gas station". The weights may also be represented as fuzzy variables. The representation of benefit and user preferences as fuzzy or crisp variables depends on the application.

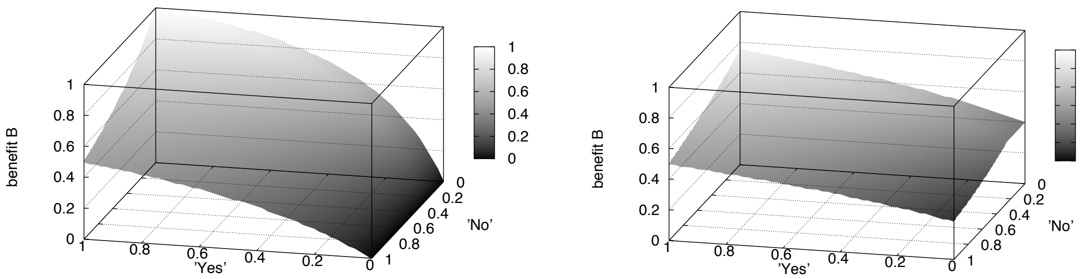
5.1.3. Resolution Level

The goal of a resolution is to transfer high-level situations in information that is distributed to other components of a P-IVRS. In our case, this is the benefit for recommendation B which is used by the decision process and user preferences U which are used by the recommender system.

User preferences are simply resolved by taking a subset $U^{TA} \subseteq U$ of relevant preferences for the task TA. The user preferences which are relevant for TA are determined by the item categories connected with the task. For instance, the item category "gas stations" is associated with the task $TA = refilling$. Only preferences which are relevant for TA are included in U^{TA}. For instance, the quality of service in a restaurant is not relevant for the task of refilling.

$$
\begin{array}{llllllllll}
\text{R1:} & \textbf{IF} & \mathit{TooLate} & \textbf{IS} & \mathit{true} & \textbf{THEN} & B & \textbf{IS} & \mathit{med} & \textbf{WITH}\ 1.0 \\
\text{R2:} & \textbf{IF} & \mathit{Later} & \textbf{IS} & \mathit{true} & \textbf{THEN} & B & \textbf{IS} & \mathit{med} & \textbf{WITH}\ 1.0 \\
\text{R3:} & \textbf{IF} & Yes & \textbf{IS} & \mathit{true} & \textbf{THEN} & B & \textbf{IS} & \mathit{high} & \textbf{WITH}\ 1.0 \\
\text{R4:} & \textbf{IF} & Yes & \textbf{IS} & false & \textbf{THEN} & B & \textbf{IS} & \mathit{low} & \textbf{WITH}\ 0.25 \\
\text{R5:} & \textbf{IF} & No & \textbf{IS} & \mathit{true} & \textbf{THEN} & B & \textbf{IS} & \mathit{low} & \textbf{WITH}\ 1.0
\end{array} \tag{5.1}
$$

The benefit B^{TA} for task TA is resolved from the high-level situation "Proactive Recommendation" PR. We reduce the four states of PR to a single variable (numeric or fuzzy) by using fuzzy rules (Listing 5.1). The benefit B is described by a three state fuzzy variable with the states "low", "medium" and "high" (like in Figure 5.3a). The states of PR are described by two-state fuzzy variables with "true" and "false" (like in Figure 5.3b) with a range of crisp values in $[0, 1]$. A crisp value of 0 results in the membership $\mu('true') = 0$ and $\mu('false') = 1$ and a crisp value of 1 in $\mu('true') = 1$ and $\mu('false') = 0$.



(a) $\mu_{Later}('true') = \mu_{Too\ Late}('true') = 0$ (b) $\mu_{Later}('true') = \mu_{Too\ Late}('true') = 1$

Figure 5.4.: Resolution plane with arbitrary membership to the situation PR

The states "Yes" and "No" of PR indicate the relevance of a recommendation. Therefore, they account for "high" B or "low" B respectively. The resolution for the benefit B relative to these two states can be seen in Figure 5.4. In Figure 5.4a, we assume no membership to "Too Late" and "Later" and in Figure 5.4b a membership of 1. The Figure 5.4a shows a maximum benefit B for $\mu_{Yes}('true') = 1$ and $\mu_{No}('true') = 0$ and lowest for $\mu_{Yes}('true') = 0$ and $\mu_{Yes}('true') = 1$. If we have no knowledge about the primary situations ($\mu_{Yes}('true') = \mu_{No}('true') = 0$), then the benefit is 0 (Figure 5.4a), unless we know that a recommendation should be given later or it is too late. In case we have full membership for both "Later" and "Too Late", the benefit B is 0.4 (Figure 5.4b).

The rule $R4$ seems useless at the first glance because the membership of "Yes" in state "false" seems already be determined by the membership in state "true" with

$\mu_{Yes}('false') = 1 - \mu_{No}('true')$. Hence, we already take this information into account with rule $R3$. However, in our case, $R4$ implies explicit knowledge about the absence of knowledge of "Yes". If we know that a recommendation should be given with a low degree of belief, then we also assume that it should not be given with high belief. The second piece of knowledge is much weaker because it is not based on observations. Therefore, the rule $R4$ has a lower weight of 0.25. Without $R4$, the benefit B would be 1 in the absence of knowledge for the primary situations ($\mu_{Yes}('true') = \mu_{No}('true') = 0$). Figure 5.4a shows that with rule $R4$, B is 0. Our intention is to assume no benefit in case of any knowledge to lower the risk of false recommendations which might be obtrusive. Note that the situations do not reflect a probability distribution. Hence, we can be absolutely sure about giving a recommendation and not given one at the same time. The absence of knowledge about not giving a recommendation does not necessarily means that a recommendation should be given. Membership $\mu_{Yes}('true') = 1$ does not necessarily implies $\mu_{No}('true') = 1$.

5.1.4. Prediction Level

The third level of situation awareness is the prediction of future situations. Prediction allows reacting on future situations. This may result in fewer recommendations along a horizon and hence less interruptions of the users in their current task. For prediction, we distinguish 3 kinds of situations:

- **Static** situations do not change along the horizon.
- **Dynamic** situations change along the horizon but the comprehension only depends on their state at a specific point in time.
- **Temporal** situations change along the horizon (like dynamic) but their comprehension depends on one or more states in the future or the past respectively (in the boundaries of the horizon).

A static situation is one that does not change over time in the horizon, e.g., friends, the user is with a whole day. In contrast, dynamic situations change over time. A simple example is the location where the user is over a day. The most complex situations are temporal ones. Like in a Markov process, the comprehension of temporal situations not only depends on the current state but on states before and after. Depending on how many states are taken into consideration, we can call them $1 - stage$, $2 - stage$ and so on situations. To make clear what a temporal situation is, we give an example in Table 5.1. In [BWP10], we used a theoretical example by means of the gas level as situation to demonstrate how temporal reasoning works. However, gas level corresponds to a dynamic situation. Therefore, we use a traffic jam as real temporal situation here.

The traffic jam example in Table 5.1 shows temporal connections of situations and whether they lead to a recommendation. The idea is to deliver recommendations such as restaurants or gas stations in advance of a traffic jam. If we are in a traffic jam

Situation before	**Situation current**	**Situation soon**	**Deliver Recommendations now?**
Traffic jam	Traffic jam	No traffic jam	-
Traffic jam	Traffic jam	Traffic jam	-
Traffic jam	No traffic jam	No traffic jam	-
Traffic jam	No traffic jam	Traffic jam	Yes
No traffic jam	Traffic jam	No traffic jam	Too late
No traffic jam	Traffic jam	Traffic jam	Too late
No traffic jam	No traffic jam	No traffic jam	-
No traffic jam	No traffic jam	Traffic jam	Yes

Table 5.1.: Traffic jam example for temporal situations

currently and this situation stays the same in the next moment or changes to no traffic jam, then it has no impact on the delivery. The same applies, if there is no traffic jam at all. If we are going to enter a traffic jam soon, a recommendation is delivered. This changes if we also look in the past. If we were in a traffic jam before, nothing changes. However, if we were not in a traffic jam and we are in one now, then recommendations are too late.

The three situations are predicted and resolved in a high-level prediction. Note that temporal situations do not refer to a prediction process itself, i.e., the connection between situation states in time does not describe a model to predict state s_{t+1} from s_t. This is part of the underlying low-level context or situation prediction. Low-level prediction comprises prediction of context information and its interpretation. Context prediction mechanisms itself are not scope of this thesis. We give details to the implementation of high-level and low-level prediction in Appendix B.2.

5.1.5. Discussion

Our approach of situation awareness for proactive recommendations aims to infer on the benefit of a recommendation and situation-aware user preferences. Structured models (called comprehension models) are used to derive benefit and user preferences from low-level situations modeled as fuzzy sets. We use structured models where the parameters are estimated to avoid cold start problems. Unstructured models as proposed by other researchers need more data to learn in the beginning of usage (e.g., Lei et al. [LZS07]).

Many approaches in literature infer on situations (e.g., Ciaramella et al. [CCLM09]) or tasks (e.g., Partridge et al. [PP09]) and deliver recommendations associated with the specific task or situation. Our model infers on information need (benefit) instead. It resembles the research of Horvitz et al. [HBH+98] who infer on the need for assistance. This corresponds to the benefit of a recommendation. The main difference to Horvitz et al. [HBH+98] is that we also take future states of the situation into regard. The prediction of information need is different to the prediction of upcoming situations or tasks (e.g., Lee and Cho [Lee10]). Our approach relies on situation and context prediction methods to make information need prediction on a higher level.

Our model is different to an event-based approach like proposed by Beer et al. [BFH+07] or Hinze and Voisard [HV03]. Event-based approaches react on events that lead to a state change of the context of the user. Events bear semantic information, e.g., "entering a room". In our model, situations have semantic information, e.g., "inside a room". In both cases, the system is triggered to make decisions. Event modeling is useful in domains in which actions of the user or environmental events can be monitored. Our application domain does not involve monitoring of the users during their task. Our approach resembles point-based approaches such as Cheng et al. [CSC+08]. Their system reacts on a snapshot of context information with a specific value. However, their approach is an interactional view on context without structured models.

We follow the general model of situation awareness proposed by Endsley (perception, comprehension, prediction) and apply a resolution on top. Only few approaches for proactive information delivery take all these levels into regard. Lei et al. [LZS07] propose a model with data atoms (context) and data elements (situations). In their case, the system reacts on transitions between data elements (events). Therefore it is different to our approach.

Some approaches also use fuzzy logic to represent situations. Park et al. [PYC06] infer on context information that is interpreted by fuzzy variables. Their goal is to derive the current mood of the user with a fuzzy logic system instead of deriving information need. Ciaramella et al. [CCLM09] use fuzzy logic in a similar manner. Context information is fuzzified and used to infer on the situation. Inferred situations correspond to our low-level situations. Hanamura et al. [HKT03] use fuzzy sets based on radial basis networks (RBF) to connect tasks, situations and actions that are performed in tasks to reach a goal. The RBF can be seen as comprehension model in our case. Cena et al. [CCG+06] propose a fuzzy logic approach that is similar to our user preference model. Context is fuzzyfied and used in the antecedent for adapting user preferences in that context. However, context in their approach only comprises characteristics of the users, e.g., age, and fuzzy reasoning is only used to overcome cold start in user profiling.

5.2. Evaluation

In the previous section, we presented our generic framework for situation awareness in proactive recommender systems. To evaluate our framework, we investigate its implementation for a proactive gas station recommender system. First, we collect data for gas station recommendations by presenting the users situations and asking them whether a recommendation is useful for them. The data is used to derive situations towards gas station recommendations. With these situations, we discuss a fuzzy logic comprehension model for user preferences U and two models for benefit B based on Bayesian networks and fuzzy logic. Our main reason of using Bayesian network and fuzzy logic models is in order to be able to backtrack the inference on high-level situations to low-level situations. This allows us to explain why a recommendation was given. Fuzzy and Bayesian representation is symbolic, i.e., its components describe real-world connections. Non-symbolic systems such as neural networks or support vector machines (SVM) do not have an explicit declarative knowledge representation. This makes it much more difficult to extract explanations for the users (Dietrich et al. [Die92]) and to interpret the model (Nugent et al.[NC05]). Their knowledge is represented by weights which does not correspond to human causal comprehension.

5.2.1. Study Design

For our evaluation, we need real user data about when a gas station recommendation is relevant in specific situations. We carry out an offline survey to collect this data. Our idea is to present the subjects of the study a series of situations related to the task of refilling. To get a large amount of data, the subjects answer the simply question "Do you want to get gas station recommendations in this situation?" several times. The answers "Yes", "No", "Later" and "Too late" correspond to the states of our high-level situation "Proactive Recommendation" PR. The horizon H is the current route of the user. The simulation of relevant situations is described in Appendix B.3. The situations are based on context information concerning car, user and environment.

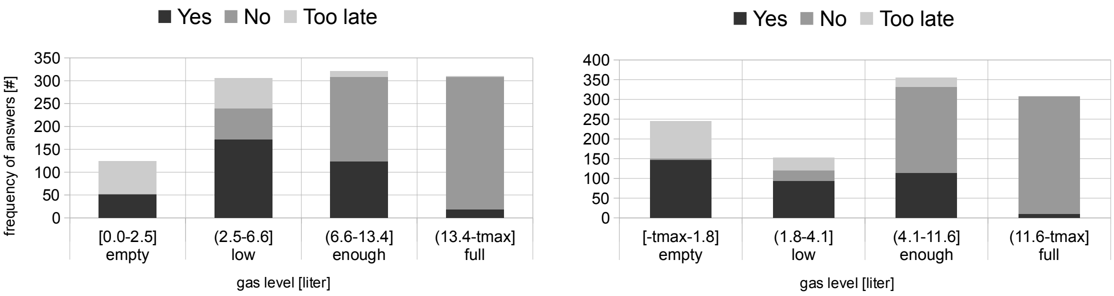
We had a total of 24 subjects with 3 women and 21 men. Each subject finished 50 runs which yields 1200 data instances. In a final questionnaire we asked the subjects about their user preferences U towards price of gas, detour for the station and what they perceive as "well located" and "inexpensive" gas stations. The preferences were given on a scale from 1 ("not important") to 5 ("important").

5.2.2. Perception of Situations

Based on the collected data, we define the perception of situations for gas station recommendations, i.e., crisp context information is interpreted as fuzzy situation variables. The interpretation comprises a discretization and the calculation of fuzzy memberships. All resulting situations can be found in Appendix B.4.

Discrete Context

Discrete low-level situations are the type of route, e.g., business, home or vacation, and the modality of refilling, i.e., if refilling is necessary. The subjects behave differently depending on their type of route. A proactive recommendation is less requested for routes with appointments that are not private (trips to business appointments or an event). It is also less requested for short trips like working or shopping. In case of longer



(a) Current gas level

(b) Gas level at the destination

Figure 5.5.: Discretization and distribution of the low-level situations "current gas level" and "gas level at the destination" with the capacity of the gas tank *tmax*

trips in the holidays, recommendation delivery is more requested because the destination is not reachable with the current gas level. Recommendations are in general requested if the destination is not reachable, i.e., refilling is necessary.

Continuous Context

Most context information is not discrete but continuous. We use the method of entropy-based discretization (Fayyad et al. [FI93]) for discretization (Appendix B.5). Figure 5.5 shows the distribution for discretized states of two low-level situations. They describe the car context information "current gas level" and "remaining gas level at the destination". The values of the interval borders in liter result from the discretization. We add an interpretation for each interval range by hand, e.g., "low" for the interval $(2.5 - 6.6]$ of the current gas level. The interpretation is used later as term in the fuzzy variable. As we would expect, recommendations are not requested, if enough gas level is currently in the tank or is still in the tank when the destination is reached. The threshold of 13.4 liter or 11.6 liter respectively represents around 25% of the tank capacity (50 liter). Recommendations usually become relevant below a threshold of around 25%. They are most relevant in the second interval of Figure 5.5a and also in the third. The first interval covers very low current gas level with a risk of breaking down. In this interval, the subjects state recommendations mostly as too late. In Figure 5.5b, we have similar results for the gas level at the destination. In case of the first interval with very low gas level at the destination, recommendations are most relevant and can be too late depending on the current gas level. In the second interval, recommendations are also relevant and they may be in the third as well but are mostly not.

We also use user and environmental situations in our simulation. User situations describe aspects concerning users' route (e.g., the length of user's route), the current position on the route and whether the users have some kind of appointment at the arrival. Other situations describe environmental aspects towards surrounding gas stations. They comprise the reachability of gas stations in general, how many gas stations are well located along the route and how many of the well located gas stations are also inexpensive. The length of the route only plays a role if it is long. In this case, the probability of reaching the destination is lower which is a reason of requested recommendations. The subjects also prefer to get recommendations in the first part of their route. For routes with appointments, fewer recommendations are requested. In case of reachability, recommendations become too late if only a few stations are reachable, more relevant with some more reachable and less with lot of reachable stations. This is intuitively clear as the reachability of gas stations correlates with the current gas level in general. This is also valid for the reachability of well located gas stations. Surprisingly, if no gas stations are well located, the subjects state recommendations as less relevant. If well located gas stations are inexpensive at the same time, relevance of recommendations is higher.

5.2.3. User Preference Comprehension Model

Our comprehension model for user preferences is simple. Long-term preferences consist of weights $U =< w_1, w_2, ..., w_n >$ for n item features (Section 4.3). They are handled like the context of the user and are interpreted as low-level situations. Low-level situations of preferences have binary membership functions (Figure 5.3b). This gives the application the possibility to capture preferences in the range $[0, 1]$. Fuzzy rules transfer low-level to high-level preferences U. Depending on the application, they can be defuzzified to weights $u = w_1, \ldots, w_n$ in $[0, 1]$ with continuous or discrete values, e.g., 0.2, 0.4, 0.6 and 0.8. An example for gas station preferences is in Listing 5.2.

$$\begin{array}{llllllll}
\text{R1:} & \textbf{IF} & PrefsDetour & \textbf{IS} & true & \textbf{THEN} & WeightDetour & \textbf{IS} & high \\
\text{R2:} & \textbf{IF} & PrefsDetour & \textbf{IS} & false & \textbf{THEN} & WeightDetour & \textbf{IS} & low \\
\text{R3:} & \textbf{IF} & PrefsBrand & \textbf{IS} & true & \textbf{THEN} & WeightBrand & \textbf{IS} & high \\
\text{R4:} & \textbf{IF} & PrefsBrand & \textbf{IS} & false & \textbf{THEN} & WeightBrand & \textbf{IS} & low \\
\text{R5:} & \textbf{IF} & PrefsGaslevel & \textbf{IS} & true & \textbf{THEN} & WeightGaslevel & \textbf{IS} & high \\
\text{R6:} & \textbf{IF} & PrefsGaslevel & \textbf{IS} & false & \textbf{THEN} & WeightGaslevel & \textbf{IS} & low \\
\text{R7:} & \textbf{IF} & PrefsPrice & \textbf{IS} & true & \textbf{THEN} & WeightPrice & \textbf{IS} & high \\
\text{R8:} & \textbf{IF} & PrefsPrice & \textbf{IS} & false & \textbf{THEN} & WeightPrice & \textbf{IS} & low
\end{array} \tag{5.2}$$

The antecedent represents long-term preferences with two-state fuzzy variables as low-level situations. The resolution of the rules leads to memberships to three-state fuzzy variables in the consequence. The consequence represents high-level preferences. To incorporate situational changes of preferences, we add further rules containing other situations. For instance, the user prefers inexpensive restaurants but for a business dinner higher class restaurants should be recommended. An example for situational gas stations preferences is in Listing 5.3.

$$\begin{array}{llllllll} \text{R9:} & \textbf{IF} & routeType & \textbf{IS} & business & \textbf{THEN} & WeightDetour & \textbf{IS} & high \\ \text{R10:} & \textbf{IF} & routeType & \textbf{IS} & business & \textbf{THEN} & WeightBrand & \textbf{IS} & low \\ \text{R11:} & \textbf{IF} & routeType & \textbf{IS} & business & \textbf{THEN} & WeightGaslevel & \textbf{IS} & low \\ \text{R12:} & \textbf{IF} & routeType & \textbf{IS} & business & \textbf{THEN} & WeightPrice & \textbf{IS} & low \end{array} \tag{5.3}$$

We know that the intention of the driver is a business appointment (antecedent). Hence, we assume that the driver has time pressure. This circumstance is reflected by the rules. They increase the long-term preferences for detour and lower the other preferences (consequence).

5.2.4. Fuzzy Logic Comprehension Model

The first comprehension model for benefit is based on fuzzy logic. The states of the high-level situation "Proactive Recommendation" PR are variables in the consequence of the fuzzy inference rules. The results after defuzzification are memberships in the range $[0, 1]$. For technical reasons, states of PR are represented with singleton membership functions. To set up the rules, we first analyze which impact low-level situations have on the high-level situation PR with feature selection methods.

Impact of Low-Level Situations

We investigate the influence of low-level situations to PR for gas station recommendations with feature selection methods described Section 2.4.4. Feature selection is a common technique in data mining to find out the influence of observed attributes on a class variable. In our case the class variable is the high-level situation "proactive recommendation" PR. First, we derive low-level situations as primary decision dimensions for gas station recommendations. These are dimensions that mainly determine the decision independent from other dimensions. Therefore, each situation is tested independently relative to the class variable PR. The results for each situation and method are summarized in Table 5.2.

We calculate a normalized average (AVG) for all low-level situations SL over all methods M to build a ranking (Equation 5.4). The value $R_i(SL, M)$ for each method M is normalized by the maximum value of all situations $R_{max}(M)$ for that method. The result is summed up for all m methods.

$$AVG(SL) = \sum_{i=1}^{m} \frac{R_i(SL, M)}{R_{max}(M)} \tag{5.4}$$

All feature selection methods select the gas level at the destination as major decision dimension and the current gas level as second. Not surprisingly, the gas level is the

Situation	MI	GR	RFF	weighted RFF	SU	CHI	AVG
Gas level at the destination	0.506	0.261	0.370	0.301	0.301	609.675	1.000
Current gas level	0.425	0.222	0.316	0.261	0.255	578.872	0.868
Modality	0.129	0.238	0.072	0.066	0.132	151.933	0.378
Reachability of gas stations	0.127	0.109	0.043	0.016	0.098	211.540	0.252
Well located gas stations	0.095	0.063	0.069	0.075	0.065	147.467	0.220
Route length	0.057	0.083	0.032	0.036	0.054	81.111	0.159
Route type	0.048	0.017	0.053	0.079	0.022	67.662	0.125
Position on route	0.035	0.036	0.065	0.055	0.029	50.023	0.124
Well located and inexpensive gas stations	0.018	0.029	0.053	0.078	0.018	28.194	0.109
Appointment	0.013	0.015	0.016	0.017	0.012	19.442	0.043

Table 5.2.: Feature selection metrics for low-level situations

strongest indicator for recommendations. The modality correlates with the current gas level at the destination but is far less important. This is due to the nature of the feature selection methods. As the driver only has to refill in a few situations, the entropy of the variable "modality" is low. Normalizing with the entropy, the variable becomes much more important (see GR of modality). The same applies to the reachability of well located and inexpensive gas stations. The result emphasizes the importance of the route as context information. Without user's route, the gas level at the destination can only be estimated. Other attributes than current gas level and gas level at the destination may play a role as secondary decision dimensions.

To investigate secondary decision dimensions, we look at the influence of situation X together with the gas level at the destination ($GLAD$) and the current gas level (GL) on the class variable PR. The resulting rules follow the implication $X = a \wedge GL = b \Rightarrow PR = c$. We use support and confidence of the rules to assess their quality. Support of a rule is defined as the occurrence of the antecedent in the set of data instances divided by the number of all instances I where $Y = GL|GLAD$ (Equation 5.5).

$$S(X \cup Y) = \frac{\#(X \wedge Y)}{\#(I)} \tag{5.5}$$

Only variable combinations with sufficient support are regarded. We choose $S > 0.02$, which corresponds to more than 20 instances in our data set. The confidence of a rule represents the strength of implication. It is defined as the occurrence of the antecedent in the data set divided by the occurrence of the consequence C (Equation 5.6).

$$C(X \Rightarrow Y) = \frac{\#(X \wedge Y)}{\#(C)} \tag{5.6}$$

The closer the confidence is to 1.0, the more subjects make similar decisions in the situation. Additionally, we use a method similar to χ^2. We compare the joint distribution $P(X, GL, PR)$ and $P(X, GLAD, PR)$ after the observation of X with the joint distribution $P(GL, PR)$ and $P(GLAD, PR)$ respectively. The increase or decrease of the probabilities represents the influence of attribute X along with GL or $GLAD$.

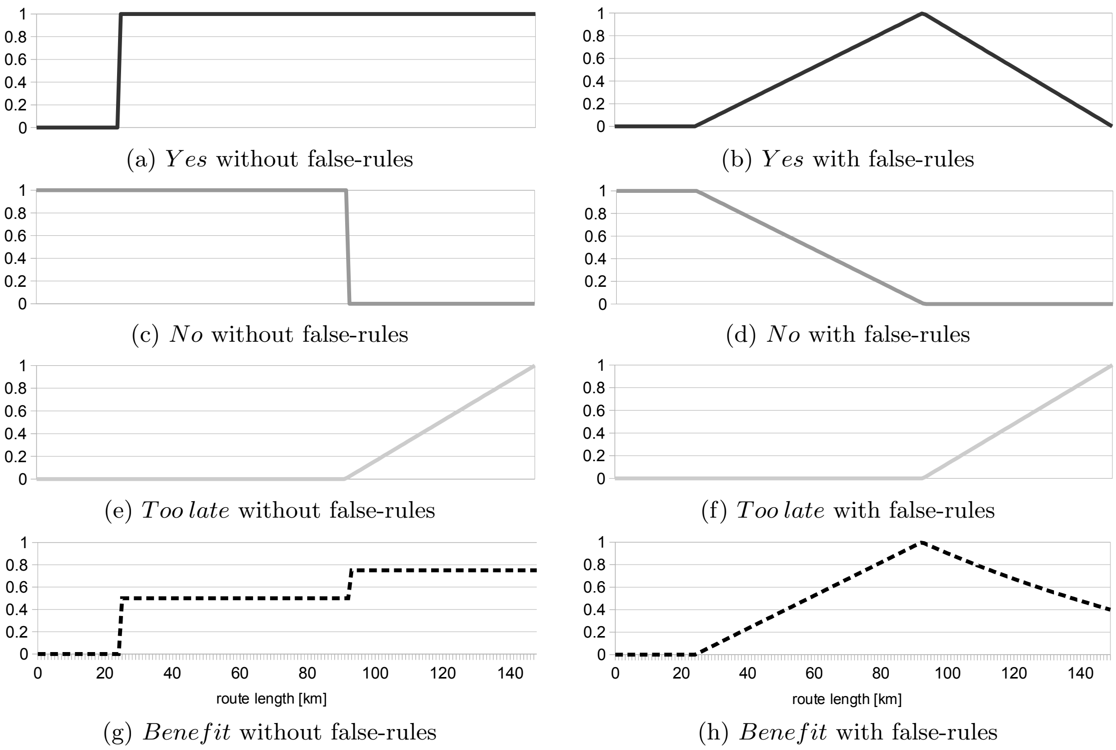
A detailed list of the results is shown in Appendix B.6. They show that route length, position on route, route type, appointment and the reachability have influence on the decision for a proactive recommendation. This applies mainly to states of these situations with a low prior probability of occurrence. Because the states are unlikely, feature selection methods cannot reveal their influence. Gas station recommendations become more relevant with a current gas level in $(6.59, 13.5]$ and a long route, a position of the route in the first half, an appointment and whether a well located and inexpensive gas station is reachable. An appointment and whether no well located gas stations are reachable have a negative influence on recommendations, in case the current gas level is in $(2.49, 6.59]$. With a gas level at the destination below 1.8 liters, recommendations are more often assessed as too late in case the position of the driver on the route is in the last third of the route or the reachability of gas stations is very low.

Dynamic Fuzzy Logic Model

Based on the results from the feature selection, we describe a dynamic fuzzy logic model. Our fuzzy rules contain low-level situations in the antecedent and high-level situations in the consequence. The rules with the current gas level are shown in Listing 5.7.

$$\begin{array}{llllllll}
\text{R1:} & \textbf{IF} & \textit{GasLevel} & \textbf{IS} & & \textit{low} & \textbf{THEN} & \textit{Yes} & \textbf{IS} & \textit{true} \\
\text{R2:} & \textbf{IF} & \textit{GasLevel} & \textbf{IS} & \textit{NOT} & \textit{low} & \textbf{THEN} & \textit{Yes} & \textbf{IS} & \textit{false} \\
\text{R3:} & \textbf{IF} & \textit{GasLevel} & \textbf{IS} & & \textit{empty} & \textbf{THEN} & \textit{Too late} & \textbf{IS} & \textit{true} \\
\text{R4:} & \textbf{IF} & \textit{GasLevel} & \textbf{IS} & \textit{NOT} & \textit{empty} & \textbf{THEN} & \textit{Too late} & \textbf{IS} & \textit{false} \\
\text{R5:} & \textbf{IF} & \textit{GasLevel} & \textbf{IS} & & \textit{enough} & \textbf{THEN} & \textit{No} & \textbf{IS} & \textit{true} \\
\text{R6:} & \textbf{IF} & \textit{GasLevel} & \textbf{IS} & & \textit{full} & \textbf{THEN} & \textit{No} & \textbf{IS} & \textit{true} \\
\text{R7:} & \textbf{IF} & \textit{GasLevel} & \textbf{IS} & \textit{NOT} & \textit{enough} & \textbf{AND} & & & \\
 & & \textit{GasLevel} & \textbf{IS} & \textit{NOT} & \textit{low} & \textbf{THEN} & \textit{No} & \textbf{IS} & \textit{false}
\end{array} \tag{5.7}$$

The rules $R1$, $R3$, $R5$ and $R6$ can be derived from the observation that a low current gas level leads to high benefit and vice versa. The results of applying the prediction on the third level of our framework (described in in Appendix B.2) only with the rules $R1$, $R3$, $R5$ and $R6$ are depicted in Figures 5.6 (a), (c), (e) and (g). Contrary to our expectation, defuzzified values for "Yes", "No" and "Too late" resemble first-order logic inference instead of smooth fuzzy values. This effect is caused by the normalization in the COG defuzzification method for singletons. As only one evidence situation is used ("current gas level"), it does not matter what membership we have for "Yes" in the consequence. It is always resolved to a value of 1.0 for memberships $\mu > 0$ and 0 else. Therefore, we introduce rules $R2$, $R4$ and $R7$ to make clear that the abstinence of membership to "true" means existence of membership to "false" explicitly. We call this kind of rules false-rules in contrast to true-rules without negation. The results with all rules together with false-rules are depicted in Figures 5.6 (b), (d), (f) and (h).



(a) Yes without false-rules (b) Yes with false-rules (c) No without false-rules (d) No with false-rules (e) $Too\,late$ without false-rules (f) $Too\,late$ with false-rules (g) $Benefit$ without false-rules (h) $Benefit$ with false-rules

Figure 5.6.: Simulation of the benefit with the current gas level

The results with false-rules are the expected smooth transitions from one state to another. However, we have to be careful with the application of false-rules. The more false-rules we define for each situation that implies the same high-level situation, the less impact has a true-rule on that high-level situation in the aggregation of rules. For instance, if we have 10 rules with low-level situations which imply the high-level situation "Yes" and only one low-level situation has high membership, then one true-rule is fired against nine false-rules. The knowledge of giving recommendations in a situation does not necessarily imply the knowledge not to give recommendations. Hence, the knowledge about not delivering recommendations is defined as a rule separately. Note, a false-rule does not represent the knowledge about not giving a recommendation in the example.

To make this circumstance more clear, we give another example with the situation "well located and inexpensive gas stations reachable". We know from our analysis of secondary decision dimensions that giving a recommendation in this situation is desirable under certain circumstances and we also know that if no well located and inexpensive gas stations are reachable does not mean not to give a recommendation. To incorporate this observation in our rules, we can combine the antecedent of all false-rules for a specific high-level situation with a conjunction. Thus, the rule only fires fully if there is no membership to all low-level situations. One example is rule $R7$ that also covers an exception. If two adjacent states of a low-level situation imply the same high-level situation, then they have to be handled together. This leads to "current gas level is low" in the rule, which ensures that the false-rule covers the area outside of "high" and "full".

Fuzzy Logic Rules with Secondary Decision Dimensions

So far, we handled primary decision dimensions with simple rules. To incorporate secondary decision dimensions, we define a general structure for our true-rules in Listing 5.8.

$$\begin{array}{lllllllll} R_C & \textbf{IF} & D_p & \textbf{IS} & true & \textbf{AND} & D_s & \textbf{IS} & & true \\ & & & & & \textbf{THEN} & SH_1 & \textbf{IS} & & true \\ R_{NC} & \textbf{IF} & D_p & \textbf{IS} & true & \textbf{AND} & D_s & \textbf{IS} & NOT & true \\ & & & & & \textbf{THEN} & SH_2 & \textbf{IS} & & true \end{array} \tag{5.8}$$

The rule R_C incorporates secondary decision dimensions D_s as condition together with a specific state of the primary dimension D_p to address a different high-level situation SH_1 in the consequence. In case the condition is not fulfilled, we define the rule R_{NC} that lets the rule fire for the regular high-level situation SH_2 associated with D_p without the condition. For instance, the reachability of well located and inexpensive gas stations (D_s) causes a recommendation ($SH_1 = true$) if current gas level (D_p) is enough. If there are no well located and inexpensive gas stations reachable, the behavior of the system with enough current gas level would be to deliver no recommendations ($SH_2 = true$). This would also be the usual behavior of the system with enough current gas level, if no secondary decision dimension is known. All secondary decision dimensions can be modeled this way.

Temporal Fuzzy Logic Model

Our data collection does not cover temporal situations. Therefore, we continue our example in Table 5.1 by formulating traffic jams as conditions. We assume that recommendations for gas stations become useful in the state "low" of "current gas level" if there is a traffic jam ahead. Furthermore, recommendations are too late if we are in a traffic jam, the current gas level is low and there was no traffic jam before. To incorporate temporal relationships, we measure the distance from the current situation state s_c to another situation state s_n along the horizon. The distance is mapped to a fuzzy variable for a temporal connection of situations. The temporal connections "after" and "before" are described by the fuzzy variable in Figure 5.7. We describe distance in minutes here. Kilometers would also be possible. The variable represents the distance to the state s_n after the current state s_c or before the current state respectively. This corresponds to Allen's Interval algebra (Allen and Ferguson [AF94]) used in temporal reasoning. In fact, every relation of the interval algebra can be described as a fuzzy variable and used in the comprehension model.

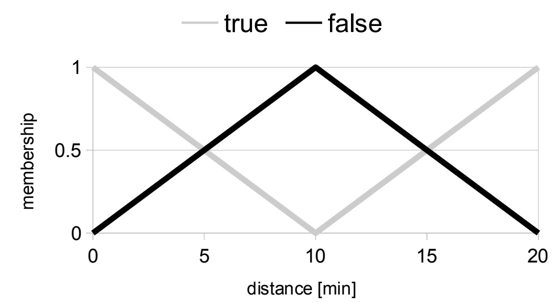


Figure 5.7.: Fuzzy variable for the distance d_{after} (d_{before} is identical) with time: maximum membership for $d_{after} = true$ at 10 minutes

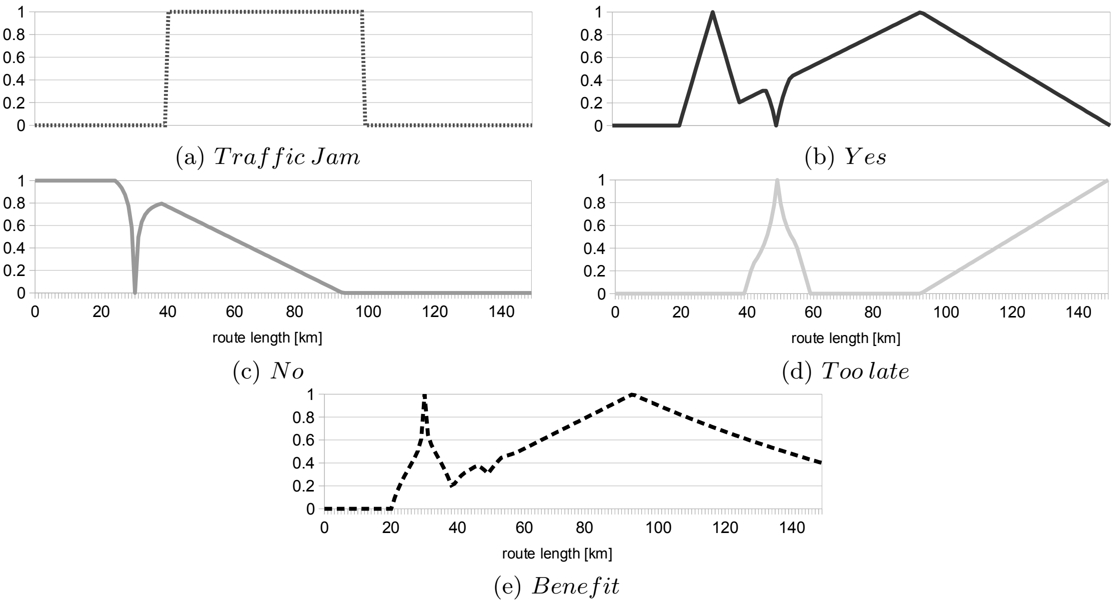
The distance has full membership to $d_{after} = true$ for 10 minutes and ranges from 0 to 20 minutes. Every state of a low-level situation outside this range is not taken into regard. Every state change is described by the current state s_c, the state to change to s_n and the distance. For temporal situations, we look backward and forward at s_c within the range of the fuzzy distance variable. The number of stages n defines how many changes of situation states we take into regard, i.e., how often a rule is fired. The result is a vector $r_p = < r_1, \ldots, r_n >$ of defuzzified values. To decide how this influences the system at s_c, we define an operator $\oplus r_p = r$ which aggregates the results. A possible operator can be the maximum of all values $\oplus \equiv \max$. We define the conditions for our traffic jam example in Table 5.1 in Listing 5.9.

$$\begin{array}{llllll} \text{C1:} & d_{after} & \textbf{IS} & true & \textbf{AND} & traffic_{now} \;\; \textbf{IS} \;\; false \\ & & & & \textbf{AND} & traffic_{after} \;\; \textbf{IS} \;\; true \\ \text{C2:} & d_{before} & \textbf{IS} & true & \textbf{AND} & traffic_{now} \;\; \textbf{IS} \;\; true \\ & & & & \textbf{AND} & traffic_{before} \;\; \textbf{IS} \;\; false \end{array} \tag{5.9}$$

Condition $C1$ covers the situation that there is no traffic jam currently but one is following. In Condition $C2$, we are in a traffic jam now but have not been in one before. We extend our rules for recommendations based on the current gas level with the conditions in Listing 5.10.

$$
\begin{array}{llllllllll}
\text{R1a:} & \textbf{IF} & GasLevel & \textbf{IS} & & low & \textbf{AND} & C2 & & \\
 & & & & & & \textbf{THEN} & Too\,late & \textbf{IS} & true \\
\text{R1b:} & \textbf{IF} & GasLevel & \textbf{IS} & & low & \textbf{AND} & NOT & C2 & \\
 & & & & & & \textbf{THEN} & Yes & \textbf{IS} & true \\
\text{R2:} & \textbf{IF} & GasLevel & \textbf{IS} & NOT & low & \textbf{THEN} & Yes & \textbf{IS} & false \\
\text{R3:} & \textbf{IF} & GasLevel & \textbf{IS} & & empty & \textbf{THEN} & Too\,late & \textbf{IS} & true \\
\text{R4:} & \textbf{IF} & GasLevel & \textbf{IS} & NOT & empty & \textbf{THEN} & Too\,late & \textbf{IS} & false \\
\text{R5a:} & \textbf{IF} & GasLevel & \textbf{IS} & & enough & \textbf{AND} & C1 & & \\
 & & & & & & \textbf{THEN} & Yes & \textbf{IS} & true \\
\text{R5b:} & \textbf{IF} & GasLevel & \textbf{IS} & & enough & \textbf{AND} & NOT & C1 & \\
 & & & & & & \textbf{THEN} & No & \textbf{IS} & true \\
\text{R6:} & \textbf{IF} & GasLevel & \textbf{IS} & & full & \textbf{THEN} & No & \textbf{IS} & true \\
\text{R7:} & \textbf{IF} & GasLevel & \textbf{IS} & NOT & enough & \textbf{AND} & & & \\
 & & GasLevel & \textbf{IS} & NOT & low & \textbf{THEN} & No & \textbf{IS} & false
\end{array}
\tag{5.10}
$$

Adding the conditions changes the rules $R1$ and $R5$. $R1a$ and $R5a$ are fired if the conditions hold and $R1b$ and $R5b$ cover regular behavior without the conditions. With these rules, we predict again at the second level of our framework with a 1-stage assumption of the development of traffic jams, i.e., we only regard the first state change. Figure 5.8 shows a simulation of a traffic jam along the route after 40 kilometers. The traffic jam lasts 60 kilometers. This means that we run low on gas inside the traffic jam.



(a) $Traffic\,Jam$ (b) Yes (c) No (d) $Too\,late$ (e) $Benefit$

Figure 5.8.: Simulation of the benefit with a traffic jam

The change of the behavior of the system with the incorporation of the traffic jam conditions occurs around 30 and 40 kilometers. The high-level situation state "Yes" reaches its maximum before the traffic jam and "Too late" reaches its maximum in the beginning of the traffic jam. This changes the benefit in these areas. Now a recommendation becomes highly relevant ahead of the traffic jam. The probability is high that the system recommends a gas station to refill before getting into the traffic jam.

5.2.5. Bayesian Networks Comprehension Model

Besides fuzzy logic, we also investigate Bayesian networks as a comprehension model for the benefit of a recommendation. We define a structured Bayesian network for gas station recommendations. The conditional probability tables (CPT) in the network are estimated by means of the data from the user survey. We show how the resulting network can be embedded in our situation awareness framework as comprehension model.

Structure of the Bayesian Network

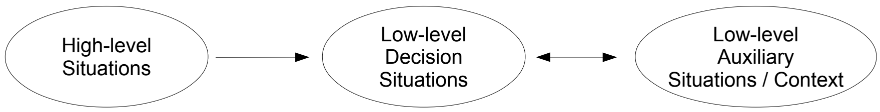


Figure 5.9.: Abstract Bayesian network comprehension model

The general structure of our network is depicted in Figure 5.9. High-level situations are not observable and therefore they are inferred in a diagnostic way (from high-level to low-level situations). This allows for a supervised learning of the CPTs from data with high-level situations as class variable. The CPTs for the low-level decision situations SL_i represent decisions towards high-level situations SH (Equation 5.11).

$$P(SH|SL_1,\ldots,SL_n) = \frac{P(SL_1,\ldots,SL_n|SH)P(SH)}{P(SL_1,\ldots,SL_n)} \tag{5.11}$$

In contrast to the fuzzy logic model, we do not distinguish primary and secondary decision situations because their influence on high-level situations is covered by the CPTs and is established during parameter estimation. We distinguish low-level decision situations that make a contribution to the decision and auxiliary situations or context. Low-level auxiliary situations contribute to the inference on unobserved decision situations, e.g., calendar entries in order to infer on an appointment. In contrast to decision situations, auxiliary situations do not have to be interpreted towards the task. We even do not have to interpret auxiliary nodes. These nodes can be raw context information as well. Furthermore, the observation of a situation node changes the probability distribution of its child node and vice versa. Thus, causal as well as diagnostic inference makes it possible to handle unobserved situations. To represent fuzzy situation variables as nodes in the Bayesian network, every fuzzy situation variable has a discrete counterpart node in the network with the same states.

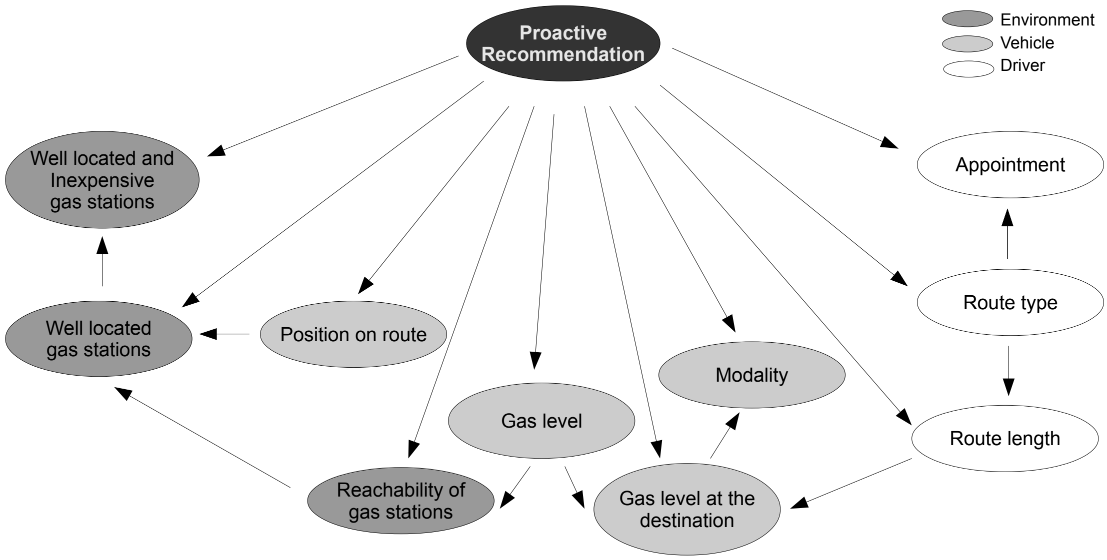


Figure 5.10.: Bayesian network comprehension model for gas station recommendations with driver, environment and vehicle context

Figure 5.10 shows an example network for the gas station recommender. The high-level situation is "Proactive Recommendation" PR. The posterior probability distribution of this situation corresponds to the membership to the states of PR. Every node in the network is a low-level decision situation grouped in driver, environment and vehicle context. Some of the variables also act as auxiliary situations, represented by inter-situation connections, e.g., the current gas level and the route length influence the gas level at the destination. The network reflects the dependencies from the data collection. It is built by hand without structure learning.

Parameter Estimation

With the structure of the Bayesian network comprehension model, we are able to set up the conditional probability tables (CPT). We use the Maximum Likelihood (ML) estimator described in Section 2.4.2 with a 10-fold cross validation. Before we run the estimation, the data set is randomized. To interpret the results, we use common metrics from machine learning that are also used for recommender systems (see Section 2.5.5). An instance is classified correctly (*true positive* tp_c) if the class value $c^* \in C$ of the data instance equals the class value with the maximum posteriori probability $c \in C$.

Our first run of the classification computes the CPTs with an accuracy of 65.667% and a "moderate" agreement with $\kappa = 0.461$. Looking into the results in Figure 5.3 unveils that the class "Later" performs poor. The metrics for the classes show that most of the values for "Later" are misclassified ($precision_{Later} = 0.308$) and nearly all values are classified as something else ($recall_{Later} = 0.086$). The AUC is also "poor" for that class value.

Class	**Precision**	**Recall**	**F-Measure**	**AUC**
Yes	0.605	0.668	0.635	0.825
No	0.718	0.852	0.779	0.874
Too late	0.609	0.447	0.515	0.887
Later	0.308	0.086	0.135	0.699
Weighted Avg.	0.622	0.657	0.628	0.840

Table 5.3.: Classification metrics for all high-level situation states

We want to know to which class value the "Later" values are classified. The confusion matrix in Figure 5.4 shows that most of the values of "Later" are classified as "No" (64%) and also some as "Yes" (27%). This is clear as "Later" means not now but later, hence a recommendation should not be given now. In the following, we exclude all 139 data instances with the class "Later" and run the classification again.

		Predicted			
		Yes	No	Too late	Later
Actual	Yes	244	70	34	17
	No	62	465	9	10
	Too late	59	24	67	0
	Later	38	89	0	12

Table 5.4.: Confusion matrix for all high-level situation states

We receive an improved accuracy of 76.2488% and a better but still "moderate" $\kappa = 0.593$. Detailed metrics for each class value are shown in Table 5.5. We have nearly an "excellent" AUC value and reasonable values for precision, recall and f-measure over all class values. Especially the classification of "No" shows good results with an "excellent" AUC. The class values "Yes" and "Too Late" have a "good" AUC. Much more values of "Too Late" have been classified as something else compared to the other class values ($recall_{Toolate} = 0.473$).

Class	**Precision**	**Recall**	**F-Measure**	**AUC**
Yes	0.704	0.685	0.694	0.848
No	0.830	0.894	0.861	0.927
Too late	0.602	0.473	0.530	0.877
Weighted Avg.	0.754	0.762	0.757	0.893

Table 5.5.: Classification metrics without high-level situation state "Later"

The confusion matrix in Table 5.6 shows that most of the instances with "Too Late" are classified as "Yes" (55). This makes sense as "Too Late" means that recommendations should be given but earlier. Critical for proactive recommendations are those values that make the system to recommend in case the user does not want a recommendation. These are 50 instances of "No" misclassified as "Yes".

		Predicted		
		Yes	No	Too Late
Actual	Yes	250	76	39
	No	50	488	8
	Too Late	55	24	71

Table 5.6.: Confusion matrix without high-level situation state "Later"

Our final analysis investigates the classification without the situations "modality" and "gas level at the destination". These situations are derived from "current gas level" and "route length". The accuracy of this run is 72.3845%. The confusion matrix in Table 5.7 shows that particularly more instances of "No" and "Too Late" are misclassified. The critical value for proactive recommendation went from 50 to 80 misclassified instances of "No" to "Yes". The result shows that the derived situations contribute to a better prediction.

		Predicted		
		Yes	No	Too Late
Actual	Yes	251	79	3
	No	80	457	9
	Too Late	65	25	60

Table 5.7.: Confusion matrix without the low-level situations "modality" and "gas level at the destination"

Fuzzy Bayesian Inference

With the Bayesian network comprehension model, we are able to predict the state of high-level situations with inference in the network. This corresponds to the third level of our framework. Our inference task is a belief update. As our network is small, we use exact inference with observed context called evidence. An evidence e is a function $f(SL) = (0, \ldots, e, \ldots, 0)$ with exact states of observed low-level situations SL. With the evidence, we calculate the update of belief in our high-level situation with Equation 5.12.

$$P(SH|SL = e) = \frac{P(SH, SL = e)}{P(SL = e)} \propto P(SH, SL = e) \tag{5.12}$$

As we are sure with our observation, the term $P(SL = e)$ is constant. This reduces the problem to a joint distribution. Unfortunately, the membership of the situation state $\mu_S(e)$ is lost. If we incorporate the state membership, we get an evidence function $f(SL) = (0, \ldots, \mu_S(e_1), \ldots, \mu_S(e_k), \ldots, 0)$ with k states with a membership higher 0. Memberships fulfill the equation $\sum_{j=1}^{k} \mu_S(e_j) = 1$ for each situation $S \in SL$. Based on Equation 5.12, we use a fuzzy Bayesian inference method similar to Park et al. [PYC06]. To calculate the posteriori probability $P(SH|SL = (e_1, \ldots, e_n))$, we use the fuzzy inference in Equation 5.13.

$$P(SH|SL = (e_1, \ldots, e_n)) = \sum_{e_i} \frac{P(SH|SL = e_i) \prod \mu(e_i)}{\sum \prod \mu(e_i)} \tag{5.13}$$

With the constraints which we defined for the membership of situations, the term $\sum \prod \mu(e_i)$ should always equal 1. We demonstrate fuzzy Bayesian inference with a short example. We use observations for the two situations "current gas level" and "reachability of gas stations":

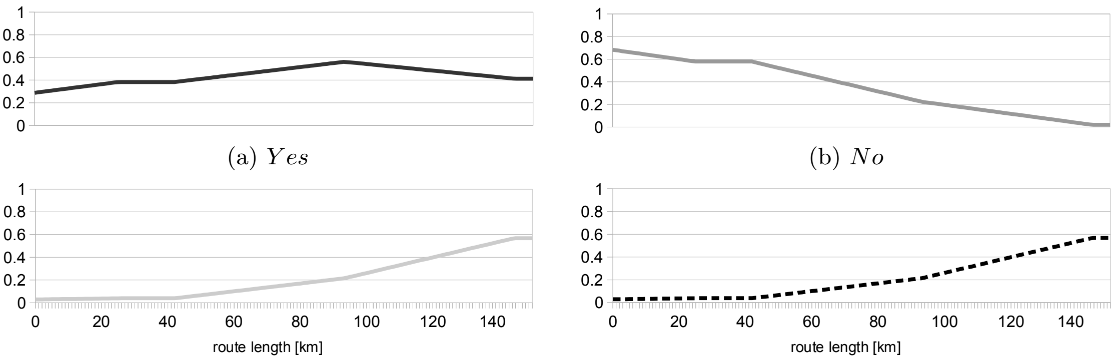
Situation	**Memberships to states**	
Current gas level	$\mu('low') = 0.7$	$\mu('medium') = 0.3$
Reachability of gas stations	$\mu('few') = 0.2$	$\mu('much') = 0.8$

Instead of one evidence e we have four evidences e_i:

- $e_1 = (\mu('low') = 0.7, \mu('few') = 0.2)$
- $e_2 = (\mu('low') = 0.7, \mu('much') = 0.8)$
- $e_3 = (\mu('medium') = 0.3, \mu('few') = 0.2)$
- $e_4 = (\mu('medium') = 0.3, \mu('much') = 0.8)$

Hence, we calculate a regular Bayesian belief update with Equation 5.12 four times and weigh the results with $(0.14, 0.56, 0.06, 0.24)$, e.g., $\mu('low') = 0.7 \times \mu('few') = 0.2 = 0.14$. The weights sum up to 1. This would not be the case, if the memberships would violate the second constraint, e.g., with $\mu('low') = 0.3$ and $\mu('medium') = 0.2$). The posterior distribution $P(SH|SL = (e_1, \ldots, e_n))$ is mainly determined by the states $'low'$ and $'much'$ because together they have a weight of 0.56.

Comprehension and Prediction

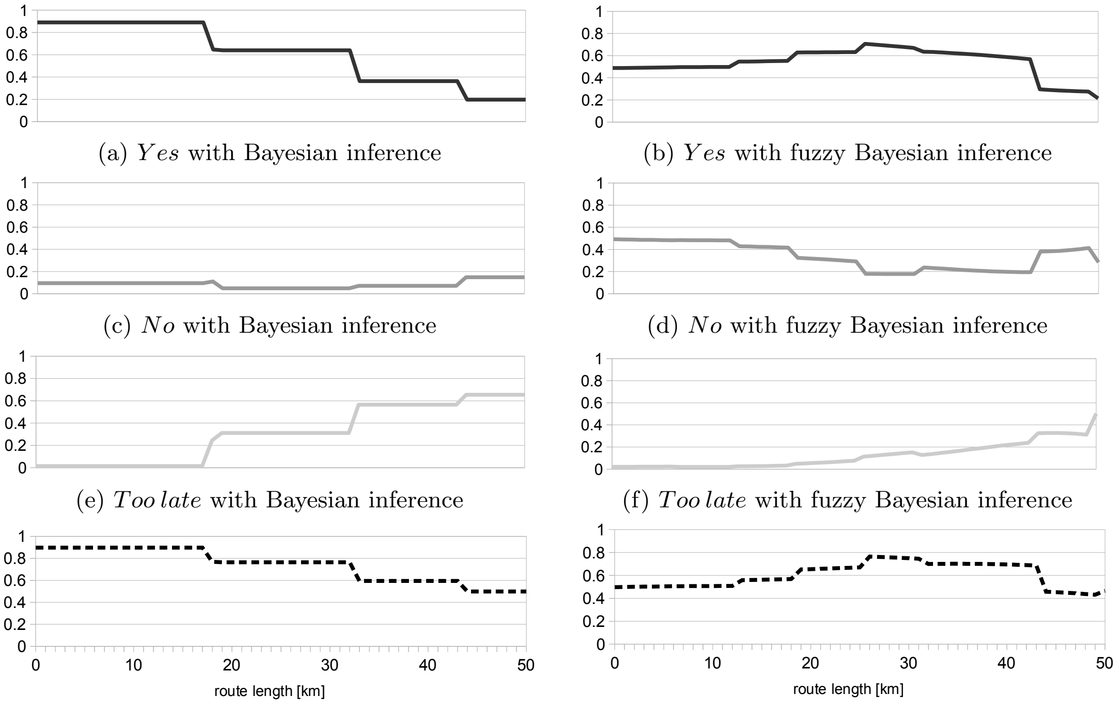


(a) Yes (b) No (c) $Too\,late$ (d) $Benefit$

Figure 5.11.: Simulation of the benefit along a route with a gas level of 12 liter in the beginning

We repeat the simulation from Figure 5.6 with the Bayesian comprehension model and an observation for current gas level. The result is depicted in Figure 5.11. The course of the curves is similar but the fuzzy Bayesian result varies less. The main reason is that the fuzzy inference mechanism does not regard unobserved situations, whereas unobserved situations play a role in the Bayesian model with their prior probabilities.

To make the difference clear, we compare fuzzy Bayesian and regular Bayesian inference. Figure 5.12 depicts the evidence of all available low-level situations in the network along a route. The simulation shows a general route of 50 km and a current gas level of 8 liter (16%) in the beginning. The driver has no appointment and is driving home. Furthermore, no inexpensive gas stations are well located. The Figure 5.12 shows the difference between Bayesian inference with Equation 5.12 and fuzzy Bayesian inference with Equation 5.13. The main difference occurs in the beginning of the route. The membership to "gas level at the destination = low" is slightly higher than the membership to "current gas level = enough". With a current gas level of 8 liter in the beginning and an assumed average consumption of $8\frac{\text{liter}}{100\text{km}}$, we get a gas level of 4 liter at the destination. Resolving this value with the "gas level at the destination" fuzzy situation in Figure 5.5b results to the highest membership in the second interval ((1.8-4.1]). The inferred membership for the high level situation "Yes" should become very high because the largest part of



(a) Yes with Bayesian inference

(b) Yes with fuzzy Bayesian inference

(c) No with Bayesian inference

(d) No with fuzzy Bayesian inference

(e) $Too\,late$ with Bayesian inference

(f) $Too\,late$ with fuzzy Bayesian inference

(g) $Benefit$ with Bayesian inference

(h) $Benefit$ with fuzzy Bayesian inference

Figure 5.12.: Simulation of the benefit along a route with Bayesian and fuzzy Bayesian inference

the second interval is "Yes". In contrast to that, Figures 5.12 (b) and (d) show that the membership for "Yes" and "No" are nearly equal in the beginning. The benefit reaches its peak later on the route. For a proactive recommender it would be likely that gas stations are recommended in the beginning of the route with Bayesian inference and later with fuzzy Bayesian inference.

5.2.6. Discussion

We investigated the application of our framework for situation awareness towards the use case of gas station recommendations. The focus was on the application of intelligent systems methods to comprehend situations towards the benefit of a recommendation and user preferences. We showed that both methods, a fuzzy logic model as well as a Bayesian network, can be integrated into the framework.

Entropy-based discretization and feature selection show that the current gas level and the gas level at the destination are the primary decision dimensions for gas station recommendations. However, further investigation unveils that other situations also play a role. This is the case if primary decision dimensions are in specific states. For instance, if there is enough current gas level in the tank, refilling is relevant in case an inexpensive gas station is available. An inexpensive gas station plays a no role, if the tank is completely full.

To resolve situational user preferences, we chose a simple but effective method. Regarding user preferences as situations has the advantage that they can be combined with other situations easily. The complexity increases if user preferences are correlated in specific situations. In this case, rules may become complex.

Using fuzzy logic to infer on the benefit of recommendations is a fast method in case of few situation variables. We showed with the differentiation of true- and false-rules how this can be done effectively for all states of a situation. The complexity increases with more situation states. With more situation variables, we have to decide whether these variables are modeled together in a set of rules or separated. The disadvantage of modeling variables together is that each state of each variable has to be regarded in a combination. This may result in many rules. Modeling situations in separate rules has the disadvantage of estimating rule weights. With many situations, this can be a challenge. An advantage of fuzzy rules is that temporal situations such as traffic jams can easily be integrated. We showed this with the introduction of a fuzzy temporal variable. Another advantage of fuzzy logic is the establishing of knowledge without data collection.

Compared to the establishing of fuzzy rules, defining the structure for a Bayesian network was easy in our case. The dependency of variables is often clear in the use cases. Otherwise it can be learned. Estimating parameters is easily possible with common methods. However, it is more complicated to add new situations to a Bayesian network model. Another disadvantage is the need of a data set for parameter estimation. Estimation by hand like for rules may become complicated. Hence, we established a user data set for our Bayesian network. Our network comprehension model shows good results for proactive recommendation prediction. The subjects could clearly assign situations to whether they want recommendations ("Yes") or do not want recommendations ("No"). The assignment of "Too Late" in case recommendations are too late in a situation also shows reasonable results. However, the subjects have problems with assessing situations where recommendations have to be given later ("Later"). This is either because the differentiation between "Yes" and "Later" is unclear or the assignment of "Later" is in general unclear or not necessary. Another result of our investigation with Bayesian networks is that derived situations like "gas level at the destination" are able to further increase accuracy in our case. Although these situations are already described by other situations in the network, estimated parameters are different.

The difference between fuzzy logic inference and Bayesian inference is that the first does not handle incomplete data in general, i.e., unobserved situations lead to no rules to fire. As a possible solution, default situation states would cause the same behavior as if the situation is observed. Bayesian networks are able to capture prior knowledge about situation states. Furthermore, conditional dependencies between situations are represented. This allows for inference on unobserved situations based on observed situations.

For instance, the route to a destination is often not entered by the drivers because they know the way from experience. The difference between both methods becomes clear in the inference. Fuzzy inference results exploit the whole membership range between 0 and 1 of the states of the variable "Proactive Recommendation" independent how many situations are observed. A Bayesian inference result only reaches a membership of 1 if the situation is clear, e.g., if the tank is full, "No" has a posteriori probability of 1.

Another difference between fuzzy and Bayesian inference is that the membership to situation states is lost in a classical Bayesian inference. Therefore we extended classical Bayesian inference to fuzzy Bayesian inference. The difference becomes clear, if the system behavior for two successive states of a situation is rather different. The result of a classical Bayesian inference shows no difference between full membership to one of the states or if the situation is only slightly more in one state than the other. Furthermore, if the membership slightly changes and the other state becomes higher, the system would abruptly change its behavior. The decision making is able to make more differentiated and smoother decisions with fuzzy Bayesian inference.

5.3. Summary

In this chapter, we presented a framework to model situation awareness for proactive recommender systems and evaluated its components with the use case of gas station recommendations. The framework consists of four levels derived from human situation awareness. The first level transfers crisp context information into fuzzy situations (perception) in regard of the task of a proactive recommendation. These situations are low-level situations. On the second level, high-level situations are interpreted by low-level situations with inference models (comprehension). The models fuse situation variables to high-level knowledge for a specific task. In our case, this is a proactive recommendation and situational user preferences. On the third level, perception and comprehension is predicted in a determined horizon, e.g., along a route (prediction). Finally, the fourth level resolves predicted higher-level situations to knowledge that can be used by the decision process (resolution). This is the benefit for a proactive recommendation and situational user preferences in our case.

The evaluation shows that the model can be implemented for recommendations of gas stations. We investigated fuzzy logic and Bayesian networks as comprehension models. Fuzzy logic shows better results if decisions are clear based on situations. They can easily be set up without much data. Also they allow to model special temporal situations easily. If situations become more and decisions based on situations become more unclear and interconnected, Bayesian networks show better results. We showed how fuzzy variables could be incorporated in Bayesian networks. To use the memberships of fuzzy variables, we extended Bayesian inference to fuzzy Bayesian inference.

6. Context-Aware Recommendations for P-IVRS

The core competence of a P-IVRS is to select relevant recommendations for the driver. The recommendation engine assesses items towards user preferences. The situation of a user determines which preferences a user has. To make the items useful for the user, context should be integrated into the assessment. We present a general approach of integrating context into recommender systems following well-known paradigms for context integration. An important requirement on a P-IVRS is that it should avoid cold start problems. Therefore, the drivers enter their long-term user preferences in advance. A recommender based on multi-criteria decision making (MCDM) is responsible for the assessment of items. It allows integrating context information as additional dimensions. To make heterogeneous dimensions comparable, item attributes are converted into dimensionless scores with utility functions. We investigate relevant criteria for the use case of a gas station recommender. Among these criteria, we believe that the route is crucial for POI selection. A user study should reveal how aspects of a route relative to a POI (e.g., detour) influence the decision of the driver regarding a POI. We use the utility functions derived from the study to implement our MCDM approach for context integration. A second user study collects ratings for gas stations. These ratings are compared to the prediction of our approach. Our focus is to investigate the performance of several different MCDM methods. We already published the ideas and results of this chapter in [BNWP11]. This chapter adds further aspects and a more detailed evaluation to our original publication.

6.1. Filtering Context-Aware Recommendations

The main idea of our approach is to regard the problem of selecting items as multidimensional decision problem and use multi-criteria decision making (MCDM) methods to calculate scores for each item. The conceptual architecture follows the paradigm of Adomavicius et al. [AMK11]. We combine the method of contextual prefiltering with postfiltering.

Figure 6.1 shows our filtering process. The process combines prefiltering and postfiltering to incorporate context into recommender systems. In between, a MCDM recommender evaluates items based on context information C_2, item features I and user preferences U in order to estimate the utility R. Prefiltering shrinks the number of items by means

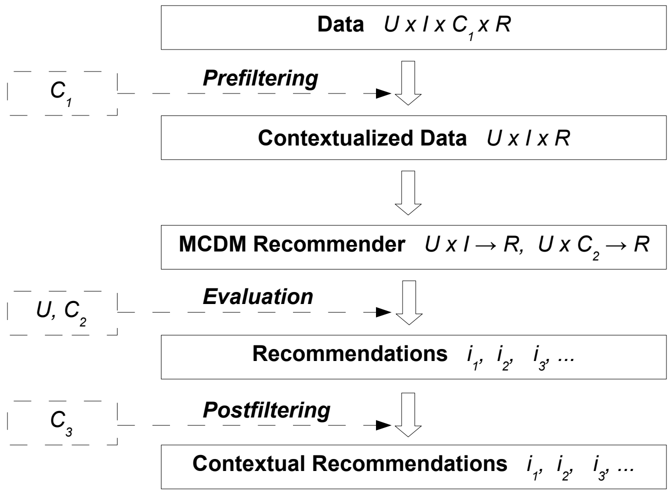


Figure 6.1.: Context-aware filtering combining prefiltering and postfiltering with a MCDM recommender

of context C_1 which reduces recommendation complexity. Postfiltering is crucial for in-car recommendations because the number of items that can be presented to a driver in these scenarios is small. It uses context information C_3 to select useful items for the user out of the set of evaluated items. All together, our filtering process contains three steps. First, non-compensatory heuristics eliminate a significant amount of items (prefiltering). Second, items are evaluated in a compensatory way (MCDM recommender). Third, a candidate set is built by comparing the evaluated items (postfiltering).

6.1.1. Prefiltering with Context

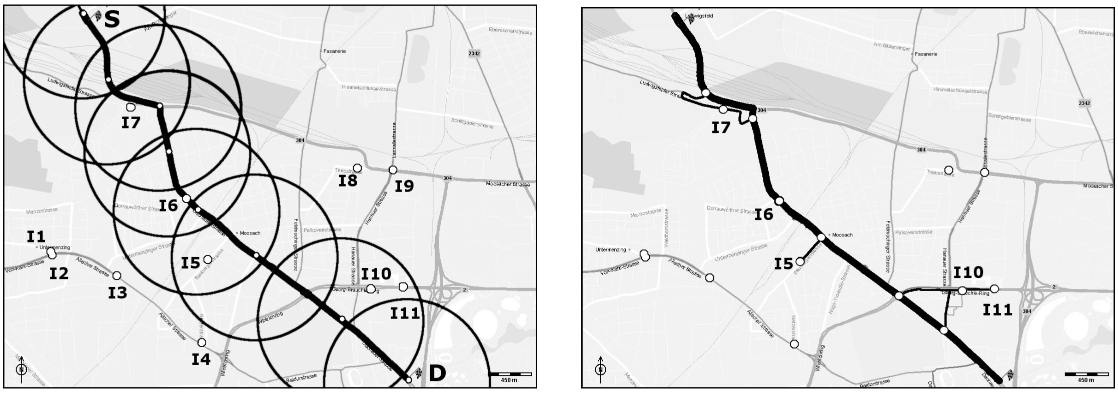
The task of prefiltering is to shrink the size of the set of available items with cut-off heuristics. This corresponds to an elimination-by-aspect (EBA) heuristic which people use for decision making facing large sets of available items (see Section 2.3.2). We apply three sequential prefilters ordered by importance and complexity. Two types of heuristics exist:

- Cut-off by Threshold: Exclude all items that exceed a predefined threshold (e.g., the location is too far away, the waiting time is too long or the facility is closed at arrival)
- Cut-off by Attribute: The item has to fulfill special properties (e.g., Diesel has to be offered or the POI needs to have a toilet)

A major attribute of POIs is their location. A cut-off by threshold heuristic for location may exclude all POIs that are either too far away from the current position or too far away from the route of the user. The first case is a special case of the second with the current position represented by a route that consists of exactly one node. We describe a method to determine the set of candidate items for the second case with several filter heuristics. The filter heuristics are arranged in a cascade beginning with the fastest method that excludes the most items and finishing with the most computational expensive method.

Filter Heuristics Sequence

First, we apply an **in-route nearest neighbor** (**IRNN**) query. This significantly shrinks the number of items for the following steps. An IRNN is able to determine those POIs that are nearest to the route of a user given a road network. Such queries are used with the assumption that the users want to keep their route as much as possible. Hence, we do not search the POI with the lowest overall detour but the closest POI to the route. The work of Shekhar and Yoo [SY03] compares four different methods for IRNN. We apply the IRNN method of spatial distance join-based filtering to find all interesting items along a route.



(a) Fast POI filtering with the Euclidean distance between the POIs I and the route

(b) Exact POI filtering with the real shortest path between the POIs I and the route

Figure 6.2.: Prefiltering POIs in two steps

Our method of spatial distance join-based filtering is shown in Figure 6.2 with an example. There is a set of items I that we want to shrink. The route of the user is described by nodes in a road network which are connected by edges. Edges represent real streets which a car is able to drive on and nodes are, e.g., crossings. In the first step a spatial distance join operation cuts off all items that do not fulfill the detour constraint in Equation 6.1.

$$\exists\, i \in I, r_l \in R : d(i, r_l) < c \tag{6.1}$$

The Euclidean distance d between item $i \in I$ and route node $r_l \in R$ have to be smaller than a constraint c for at least one pair (i, r_l). R is the set of all route nodes. A fast method to calculate this distances is to index the location of all items I and the route nodes R with a spatial index tree, e.g., a R^* tree, and calculate a join between both sets (comparable to a database). The index of all items I is precomputed because it is static and does not change often. The route node index is calculated on the fly because the number of route nodes is small enough to do it efficiently. The Euclidean distance is not based on the road network but is the lowest distance between the location of a POI and a route node (the air line distance). It is always less than or equal to the road network distance. It can be calculated faster than the road network distance. Figure 6.2a shows the result for an example route from start node S to the destination node D. The Euclidean distance with constraint c results in circular areas with radius c around the route nodes. Each item $i^* \in I$ that is located in the circle passes the first step (in this case $I5, I6, I7, I10, I11$). The join results in a set of route nodes R_i^* for each passed item i^*. The set contains all route nodes that have an Euclidean distance smaller than c to the POI location.

The application of the second step is simpler. It uses cut-off by attribute heuristics to eliminate all items in the set I^* that do not have the required attribute. A set of attribute constraints C determines which attributes an item has to have. For instance, an attribute can be a toilet or the business hours of a facility. If a POI has no toilets or is closed, it is removed from the set I^*.

The third step is the most complex one. Therefore, it is done with an already reduced set of candidate items I^*. The first step provides a subset of route nodes $R_i^* \subseteq R$ for each item i^*. These are potential departure points on the route for the item i^*. The third step uses the road distance from items i^* to the route nodes R_i^*. It is the shortest path in a road network $N = (V, E)$ where V are the nodes of the network and E are the edges that connect the nodes. Edges are directed because there can be one way streets. To find the shortest paths between R_i^* and i^*, we use Dijkstra's algorithm [Dij59]. The set R_i^* already corresponds to nodes in the road network N. For each i^*, we have to find the node $v_i \in V$ that represents the item in the network. There are several methods called map matching to do this (e.g., Quddus et al. [QON07]). Their discussion is out of scope of this work.

The shortest paths between the items I^* and the route of the user are shown in Figure 6.2b. They connect nodes of the route with the location of the POI. Dijkstra's algorithm calculates the shortest paths between node v_i representing the location of the item and the route nodes R_i^* (filtered in the first step). As there are one-way streets in a road network, the calculation has to be done two times. The first calculation determines the shortest paths from the item i^* to the route nodes (forward calculation). The second calculation results in the paths from the route nodes to the item. Instead of calculating n shortest paths to the same destination i^* with n nodes in R_i^*, Dijkstra's algorithm can be executed only one times with flipped directions of the edges (backward calculation). The results are two sets of shortest paths $S_{i,forward}$ and $S_{i,backward}$ from item i^* to the route nodes (forward) and vice versa (backward). With the lengths of the shortest paths, we need to evaluate the extended version of Equation 6.1 with Equation 6.2.

$$\exists\, s_{i,k} \in S_{i,forward}, s_{i,l} \in S_{i,backward} : d(s_{i,k}) + d(s_{i,l}) - d(v_k, v_l) < c \tag{6.2}$$

At least one path over the item (composition of two shortest paths, one from $S_{i,forward}$ and one from $S_{i,backward}$, minus the skipped route) has to fulfill the constraint c. The distance $d(v_k, v_l)$ is the part of the route that is skipped if driving to the POI. We call this composition the detour. Note that the constraint c is equal to the first step because the route to the POI is not necessarily the route from the POI. For instance, in Figure 6.2b, the way to item $I10$ from the route is not the same as the way from the item to the route.

Route Context

While prefiltering the items, we calculated shortest paths for all remaining items I^* after prefiltering. We use this information to build the route context of an item i^*. The route context is determined by criteria that the users take into consideration when they choose a POI concerning the path from the route to the POI and vice versa. Each POI has its own corresponding route context. The route context consists of criteria that can be built by the elements in Figure 6.3.

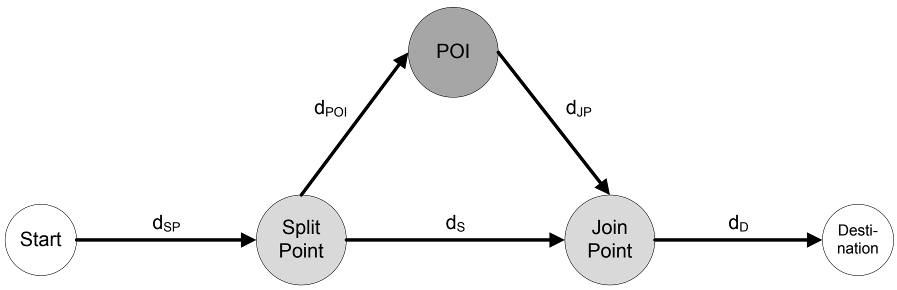


Figure 6.3.: Elements of the route context

For all distances d, the unit is determined by the route profile that the users select (e.g., fastest or shortest). The split point $SP \in R_i^*$ is a decision point on the route where the users leave their route to get to the POI. At the join point $JP \in R_i^*$, the users get back to their original route. Based on the different distances d available in this context, we define three criteria:

- Detour $= (d_{POI} + d_{JP}) - d_S$: The overall detour the users have to accept.
- Deviation $= d_{POI} + d_{JP}$: The new route the users have to drive additionally to get to the POI and back to their original route.

- Skipped route $= d_S$ if $d_S > 0$ else 0: The part of the original route the users leave out when driving to the POI.

The detour stands for minimizing the additional distance that is needed to drive to the POI. To minimize the deviation means to find POIs closest to the intended route. Many user groups, e.g., commuters, select a route in advance and may not want to deviate from it much. Therefore, deviation is important for the drivers who do not want to use other routes than the originally planned one. Analogously, minimizing the skipped route corresponds to the intention to drive as much as possible on the path of the original route. Generally, the larger one of the criterion becomes, the larger are the others. In Figure 6.2b, there are some examples of different behavior. The skipped route of $I5$ is 0, but the deviation is larger. Deviation and skipped route of $I7$ are similar. Hence, the detour is small, although the drivers have to leave their route. Depending on the preferences of the driver, each of the presented criteria can be weighted differently in the system.

6.1.2. Evaluation of Items with a MCDM Recommender

The MCDM recommender brings user preferences, context and item attributes together. Due to the requirement to avoid cold start problems, we apply a multidimensional knowledge-based approach. It is flexible to include collaborative or content-based filtering as further dimensions to support hybrid recommendations.

Score Calculation

In contrast to a common multidimensional context integration paradigm (e.g., in Adomavicius and Tuzhilin [ASST05]), our approach uses loose coupling. User preferences U are not represented relative to context C_2. Preferences are either independent from context, e.g., brand of a gas station, or apply to specific context information, e.g., preferences for the route context. We distinguish between quality and context of an item. Quality corresponds to the classical approach to calculate ratings for an item ($U \times I \rightarrow R$). We do not use ratings explicitly but abstract scores. Scores can be ratings but also utilities. Hybridization of several recommender methods is supported by regarding the predicted rating of each method as score. It corresponds to a weighted hybridization method because of the MCDM approach. Besides quality, the items are also evaluated for context information ($U \times C_2 \rightarrow R$), i.e., the users have preferences U for context information C_2. The set of context information C_2 that is used for the recommender can be different to the set of context information C_1 that is used for prefiltering.

Figure 6.4 shows our three level approach. The attributes $a_{i,j}$ belonging either to the item I or to the context of the item C_2 are reduced to abstract scores s_i in the range $[0, 1]$ on level 2. Level 2 scores are described by a predefined hierarchy of attributes such as the route context, i.e., the attributes $a_{i,j}$ that are reduced to score s_i are determined

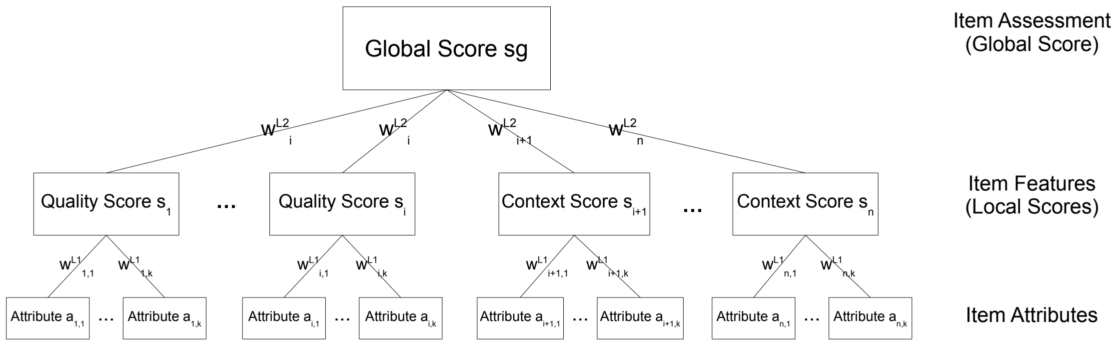


Figure 6.4.: Integrating context into a recommender system with two types of scores to assess each item

in advance. For the reduction, we use either general user preferences or specific user preferences. General user preferences are represented by a predefined utility function for each attribute, e.g., for each criterion of the route context. Specific user preferences are used to integrate classical recommendation systems, e.g., predicted ratings in a collaborative filter, or preferences explicitly stated by the user, e.g., for a specific gas station brand. The weights W^{L1} are static during the run-time of the system. They can be adapted afterwards. e.g., by machine learning mechanisms. We call second level dimensions item features. There is one score for each item feature. The score is only valid for a specific item. Therefore it is local. Local scores are aggregated to one global score $sg \in [0, 1]$ for each item by means of MCDM methods (Section 2.3.3). The global score can be used to compare items with each other or to rank them. The weights W^{L2} represent long-term user preferences that are given by the users explicitly and may be adapted depending on the situations of the users.

The selection of item features should be done carefully. Generally, they should cover decision dimensions towards an item. For instance, all monetary aspects should account for one score. They should be understandable for the user to make a choice among alternative items. We assume that decision problems are represented by a few significant features on level 2 for most application scenarios. Findings in decision theory (Section 2.3) show that people only take a few features into account to lower the effort of decision making, especially for comprehensive decisions.

The advantage of generic dimensionless scores on level 2 is that well-known MCDM methods can be used for further calculation and item features become more easily comparable. Having several scores for one item in different granularity (local and global) allows post-filtering methods to select a set of items to recommend more accurately. Other selection methods than ranking the items are possible. Furthermore, explanations can be created by selecting arguments based on item features because these features represent human understandable information.

User Profile

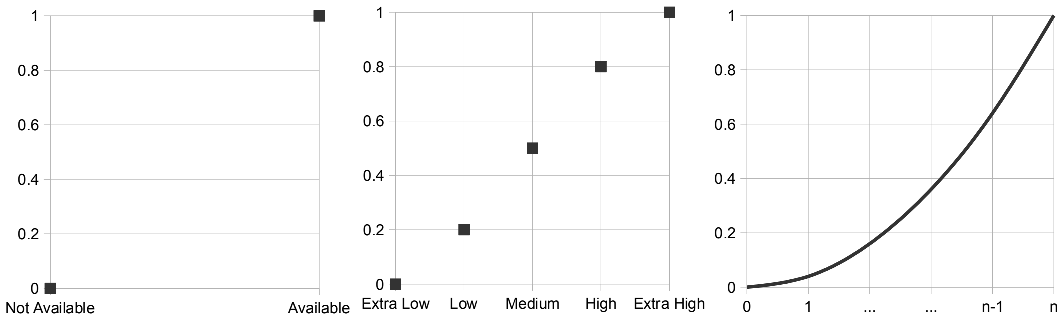
User preference elicitation should be able with low effort for the driver. In case of our MCDM recommender, the users have the possibility to set long-term preferences for each task TA manually. This involves evaluating a set of predefined (level 2) item features for an item category that is associated with a task. The actual items to recommend only have to provide a subset of these features. Usually, the evaluation should be done before driving, e.g., while standing still or at home. The drivers do not evaluate attributes on level 1 but experts estimate the weights of the attributes W^{L1}, e.g., based on questionnaires. The weights for n item features are stored as normalized weights $U(TA) = w_1, \ldots, w_n$. They are regarded as low-level situations to be able to adjust them to the situations of the users. A comprehension model like described in Section 5.1 derives high-level situation weights $W^{L2}(TA)$. For example, if the user prefers a specific brand of a POI, then the weight for that feature is in general high. If the situation awareness component recognizes that the user has time pressure, then the comprehension model may lower the weight for this feature. If there are no preferences for that feature, then time pressure has no influence. Some of the features are modeled by predefined utility functions that represent how much utility the feature of an item has to the user. For all other (discrete) features that cannot be predefined as they are user-specific, e.g., the brand of a POI, the user is able to prefer values of the features. The final user preferences determine the utility functions that are used by the MCDM recommender. Defining a default user profile (e.g., based on data mining) and adapting the weights in the user profile may reduce user interaction for preference input.

Utility Functions

To express the utility of a context item feature for a user, we use utility functions. Utility functions may also be applied to quality item features in case of utility-based recommendations. We identified three generic utility functions that allow describing all the dimensions we need (Figure 6.5).

The utility function for properties has either a utility of 0 if a property is not available or of 1 if it is available (Figure 6.5a). Note that this is different to the cut-off by attribute heuristic. The last requires the availability of the property and the utility function represents only one decision dimension among others, i.e., items may be recommended which do not have the property. Examples for property utility functions are available services, the brand of gas stations or the type of a restaurant.

The utility function for interval-based criteria (Figure 6.5b) is used for information that is not described numerically, e.g., after discretization of numerical values. An order from "low" to "high" describes the general occurrence of crisp values for this utility function. Intermediate steps such as "very low" may also be used. Outside the regular occurrence of values, we define the steps "extra low" and "extra high" to describe superlatives. For instance, the service in a restaurant may be "bad" or "good" for most of the restaurants



(a) Property (b) Interval (c) Numeric

Figure 6.5.: Generic utility functions

but some are outstanding. Assigning the highest score to these restaurants like for regular values would underrate the restaurants. If the user strongly prefers service, then we want restaurants with an extra ordinary service to be recommended. This utility function covers, among others, waiting time, price, food quality, service or atmosphere of a restaurant. The specific utility values need to be set up for each feature separately by experts.

This also applies to numerical utility functions. They map real valued numbers to a utility (Figure 6.5c). The utility function for money is an example. They can either be described with a mathematical function or with a set of sample points and associated utilities. Utility values between the sample points are interpolated. The price of gas, the detour caused by a POI and the remaining gas level at the gas station can be described by numeric utility functions.

6.1.3. Postfiltering with Context

After evaluating items with the MCDM recommendation approach, the final step is to select items that should be delivered to the driver. Two different approaches are possible for postfiltering. Postfiltering either classifies a recommendation set of k items or ranks the items. A combination is also possible. For P-IVRS, we want to build small recommendation sets of items that are recommended to the user. Postfiltering extends our recommendation approach by being able to find some good items and build a small recommendation set. We describe three different methods for building these recommendation sets. The decision process actually uses these methods to build a recommendation set (Section 4.3.3). It depends on the implementation of the decision process which methods are used and how they are combined. Our focus in this section is on the postfiltering methods. They allow the system to evaluate conditions on a recommendation set, e.g., the maximum number of items.

Dominance-based Recommendation Set Building

The idea behind dominance-based recommendation set building is to select items by means of local scores of the filtering hierarchy (Figure 6.4). Items that are dominated by other items in the recommendation set are eliminated. In Section 2.3.3, we described different kinds of dominance filter.

Ranking-based Recommendation Set Building

The idea of ranking-based recommendation set building is to process the evaluated items from the recommendation process in a specific order and to decide if each item should be included in the recommendation set. The iteration either terminates if the recommendation set reaches its maximum size n_{max}, no item is available anymore or the items are too "bad", i.e., if their global score sg is smaller than the threshold α. As an alternative, it is also possible to simply take the first "top-k" items. However, we believe that this would lead to items that are more difficult to distinguish by the driver because item features are more similar. We describe two alternative similarity metrics that determine whether an item should be included in a recommendation set.

Two items are similar in their structure if they perform similar on item features. For instance, if both items perform better in gas price than in detour and better in detour than in brand. We also incorporate the weights W^{L2} in the calculation. To calculate the correlation between item x and item y, we regard both items as vector $\overrightarrow{x}$ and $\overrightarrow{y}$ with their local scores as values. We assume that weights are normalized: $\sum_{w_i \in W^{L2}} w_i = 1$. The weighted mean for x is $m(\overrightarrow{x}, W^{L2}) = \sum_{x_i \in \overrightarrow{x}, w_i \in W^{L2}} w_i x_i$. The same applies to y. Equation 6.3 shows the weighted covariance.

$$COV(\overrightarrow{x}, \overrightarrow{y}, W^{L2}) = \sum_{x_i \in \overrightarrow{x}, y_i \in \overrightarrow{y}, w_i \in W^{L2}} w_i(x_i - m(\overrightarrow{x}, W^{L2})) \times (y_i - m(\overrightarrow{y}, W^{L2})) \quad (6.3)$$

Equation 6.3 leads to the weighted correlation in Equation 6.4.

$$COR(\overrightarrow{x}, \overrightarrow{y}, W^{L2}) = \frac{COV(\overrightarrow{x}, \overrightarrow{y}, W^{L2})}{\sqrt{COV(\overrightarrow{x}, \overrightarrow{x}, W^{L2}) \times COV(\overrightarrow{y}, \overrightarrow{y}, W^{L2})}} \quad (6.4)$$

Note that only items that have the same structure have high correlation. For instance, $x = (0.1, 0.2, 0.3, 0.4)$ and $y = (0.7, 0.8, 0.9, 1.0)$ strongly correlate but x is much better than y.

Besides the similarity of the structure, we can also use the similarity of the content of two items to decide whether they should be included into the recommendation set.

This accounts for real similarity based on the attributes that are relevant in decision making. The metric is useful if we only use an estimated value of an attribute and if there is a similar item that has the value of the attribute. Incomplete information is common in user-generated data or data from Internet sources, especially if the validity of information is volatile. For instance, prices for restaurants and gas stations are often not available or outdated. In some cases, we can only estimate the attribute based on other attributes. If there is an item in the recommendation set that has a price but is slightly worse, we replace the item with the missing attribute. Thus, more accurate information can be presented to the user, e.g., for justification.

To calculate content similarity, we use the weighted Euclidean distance in Equation 6.5. As attributes are not in the same range, they have to be normalized first.

$$d(\overrightarrow{x}, \overrightarrow{y}, W^{L2}, W^{L1}) = \sqrt{\sum_{x_i \in \overrightarrow{x}, y_i \in \overrightarrow{y}, w_i \in W^{L2}} \sum_{a_j \in x_i, b_j \in y_i, w_j \in W^{L1}} w_i \times w_j \times (a_j - a_j)^2} \quad (6.5)$$

Score-based Recommendation Set Building

Finally, we can also build recommendation sets by selecting all $\binom{n}{k}$, $\binom{n}{k-1}$ until $\binom{n}{1}$ permutations of recommendation sets with $k = n_{max}$ and the number of evaluated items n. This means that we build all possible recommendation sets with a maximum of k elements. For all recommendation sets, a score is calculated that determines the quality of the recommendation set. The recommendation set score can incorporate similarity and diversity of the items in the recommendation set or the number of items. As calculating the binomial coefficient can be computational costly ($\binom{100}{3} = 161700$) heuristics need to be applied to decrease n.

6.1.4. Discussion

The main difference between our method and context integration methods described by Adomavicius and Tuzhilin [AT05b] [AT05a] or Baltrunas and Ricci [BR09] is the representation of items and user profiles. Their approaches are based on a classical recommender system matrix with user preferences stored as utility (e.g., a rating) for items ($I \times U$). Context C is added as further dimension and the multidimensional matrix is solved in different ways, e.g., reduction of dimensions. Our presentation of use cases for P-IVRS in Section 4.1 indicates that items in most applications are better represented by item attributes rather than ratings (e.g., gas stations or parking lots). The decision for these items is based on objective criteria rather than taste. Furthermore, we believe that sparsity problems would arise because the drivers are probably less willing to give ratings for such items.

Another aspect of P-IVRS is cold start. Classical context integration with a multidimensional recommendation matrix would need time to make suitable recommendations. As proactive recommendations are not requested by the users, they are probably less tolerant to irrelevant items. Therefore, we use a knowledge-based approach that is flexible to integrate classical recommender methods. We put initial engineering effort in the setup of utility functions. The user should have only little interaction effort with the system in advance. The result is a representational view on context, i.e., we predefine all relevant context information, in contrast to an interactional view such as described by Yap et al. [YTP05], Palmisano et al. [PTG08] or Brunato and Battiti [BB03] where relevant context is learned from data.

Hybrid recommendation approaches are a common technique to solve cold start problems in proactive recommender systems. Beer et al. [BHZR06] describe how a hybrid recommender of collaborative filtering, content-based filtering and knowledge-based filtering can be used to gather as much as possible knowledge about the item. A similar approach can be found in Wörndl et al. [WG07]. Like our prefiltering, the idea is to shrink the amount of items with context or with one type of recommender and to apply further recommenders to make the assessment in a sequence. Ducheneaut et al. [DPH$^+$09] present the idea of using models instead of classic recommenders. Models may represent a classic recommender but can also comprise context aspects or learned models. This is similar to the reduction of context from the first to the second level in our model but we focus on meaningful features on level 2 instead of arbitrary predictive models.

The users of our system are able to define weight-based preferences for item features. User preference elicitation for features is common in decision theory. In regard to context, some approaches such as Agrawal et al. [ART06], Park et al. [PYC06] or Stefanidis and Pitoura [SP08] use context-aware preferences, i.e., user state their preferences for a feature in all relevant situations. As proposed by Park et al. [PYC06], this may result in utility functions. We set up utility functions with collected user data in advance to make preference elicitation easier. This results in non-personalized functions but the effort of preference elicitation is much lower. Furthermore, there are decision dimensions with an obvious tendency of utility, e.g., lower price is in general preferred over higher price.

Another characteristic of our user preferences is that they are adapted to the situation if necessary. The idea is similar to user splitting by Baltrunas and Amatriain [BA09] where context-aware profiles are generated at run-time. It is also similar to Park et al. [PYC06] where the situation (mood) determines which user preferences are applied. Schmitt et al. [SDB02] cover the complex domain of car buying with advanced mechanisms of user preference elicitation. The users are able to combine item features with operators such as OR or AND in their preferences. We believe that our domains can be covered with simpler preference elicitation.

Using the location for prefiltering is common for mobile guides (Krueger [KBH$^+$07]) or proactive recommenders (Oppermann et al. [OSJ99], Schmidt-Belz et al. [SBNPZ02], Brown et al. [BCB$^+$05], Aras et al. [ALWM10] or Modsching et al. [MKHG07]). In addition to these approaches, we make use of the route of the user as filtering space. The probability to find better items in a large space is higher than based solely on location. The route as context is mainly described by the detour. POI recommenders seldom regard detour (e.g., Bachfischer et al. [BBH+07] or Ablassmeier et al. [APRR07]). Horvitz et al. [HKS07] and Cho et al. [COYO06] take the detour for a POI into regard. They calculate routes via the POI and use the difference to the original route as detour. We believe that the route context is more complex and also comprises aspects such as deviation or which kind of streets have to be used to get to the POI. Moreover, our route context is represented as a dimensionless score with a utility function. It is generic and can be compared to other item features.

Scoring is a promising approach for context-aware recommendations in a proactive setup. Other methods described by Wörndl et al. [WHBGV11] and Cena et al. [CCG+06] also calculate dimensionless scores. Their idea is to calculate scores for aspects such as context, quality (user preferences) or proactivity and to fuse these scores (mostly by a linear model). Our approach is slightly different. Although we calculate one global score for an item, we also calculate local scores for item features and use them for item selection and for explanations. A global score is reasonable as long as the number of features is comprehensible for the user. Otherwise, the balance between quality and context becomes biased towards one of the aspects, e.g., with only one quality dimension facing many context dimensions. Furthermore, we are able to use aggregation methods beyond linear weighted models. Weighted models are common for the aggregation of information in proactive information systems (e.g., in Choeh and Lee [CL08], Ducheneaut et al. [DPH+09], Bellotti et al. [BPR+08], Wörndl et al. [WHBGV11] or Cena et al. [CCG+06]).

Besides calculating a global score, we apply dominance-based item selection methods in postfiltering. Our approach uses dominance filtering with generic context information in contrast to specialized approaches such as the location-based Skyline filter of Kodama et al. [KIGI09]. Our model is similar to the AHP proposed by Yeung and Yang [YY10]. Although the authors focus on AHP, their approach could be extended to other MCDM methods as well. In contrast to our approach, they do not rely on utility functions and the features in the AHP are fixed for all application scenarios. Our features depend on the type of application itself. Our approach also resembles Guttmann's Tate-A-Tate [Gut98]. Although the system is targeted to desktop users with intensive interaction and a lot of preferences, the author also establishes general utility functions and exploits assumptions like that costumers prefer faster and slower delivery time. Like in the work of Felfernig et al. [FMS+10], information is aggregated by means of constraint satisfaction methods. We use MCDM methods instead. Felfernig et al. [FMS+10] use a similar idea of few comprehensible features but their approach is targeted to the domain of complex financial product recommendations.

Our postfiltering approaches focus on building small sets of items to recommend. Only a few approaches regard the whole set of items as recommendation instead of individual items. Aguzzoli et al. [AAM01] describe recommendation set building for music recommendations. However, music recommendations are rather a different application domain. Teppan and Felfernig exploit the idea from decision theory that different items in a recommendation set may change the whole perception of the recommendation set. They use decoy items to increase sales. However, the domain of e-commerce is different to POI recommendations. Finally, small recommendation sets in our solution are restricted to the few items that should represent the solution as much as possible. If results are too similar, the users may have to add additional interaction effort to find other solutions. Price and Messinger [PM05] share this view. The authors calculate the utility of a recommendation set towards diversity of items.

6.2. Determining Utility Functions

Our goal is to put as much as possible effort into the description of the user model to lower initial effort for the drivers. Therefore, we derive general utility functions for as many item features as possible. In Chapter 5, we showed the usefulness of the driver's route for situation awareness in our proactive gas station recommender. In case of the recommendation process, we use the route as context because we think that it has a primary effect on the assessment of gas stations and other POIs. The location of a POI relative to the route determines how much effort the user needs to get to the POI. Intuitively, we think that no driver would drive long distances for a gas station when there are some close by. Also, a restaurant is probably not of interest if the user is in Munich and the restaurant is in New York. On the other hand, if many POIs have similar context relative to the route, it is harder to derive how the users perceive such a detour. Therefore, we investigate in this section how much detour a user would take into account for a POI. Gas stations are our main focus. As there are many other relevant types of POIs, we want to draw a comparison to gas stations. We select restaurants, as a popular kind of POI often applied in recommender systems. Restaurant selection differs from gas stations as individual taste and social aspects are more influencing in this case.

6.2.1. Study Design

The study is designed as an online questionnaire (Appendix C.1). The questionnaire consists of two parts. The first part deals with user perception of detour and the second with components of the route context. In the first part, three different imaginary routes are described. The routes consist of a short route to work (20 min or 20 kilometer), a medium route to friends (90 min or 120 km) and a long route on vacation (200 min or 400 km). To derive the perception of detour, two POIs A and B have to be compared by the participants of the study. The scale for comparison ranges from 0 to 10:

0	Does not correspond to my preferences.
3	Corresponds to my preferences partly.
5	Corresponds to my preferences moderately.
8	Corresponds to my preferences broadly.
10	Corresponds to my preferences fully.

Preferences are an abstract measurement for criteria that the user puts importance on by selecting POI. At the end of the assessment for each POI type, the participants are asked about their preferences. They see POI A that is located on the route and does not cause any detour. POI A rather does not correspond to their preferences (assessment of 1). The participants should give a rating on the presented scale for POI B. POI B is located off the route and causes a detour. It is presented with several different detours in the range of 1 to 15 minutes (or kilometer). The questionnaire includes gas stations as well as restaurants.

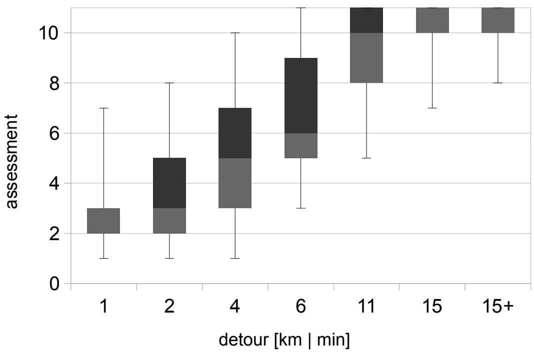
In the second part, we ask the participants how they assess the location of a POI relative to the route. They have to give a rating on a 10-point differential scale from "important" to "not important" or "don't know". Detour, deviation, skipped route as well as the street type have to be assessed.

We had 42 participants of which 39 are men and 3 are women. Most of them are between 18 and 30 years old (35), 6 are between 30 and 40 and 1 is between 40 and 50. 16 of the participants use their cars at least once a week and 11 even every day. 10 use it at least once a month and 5 less than that.

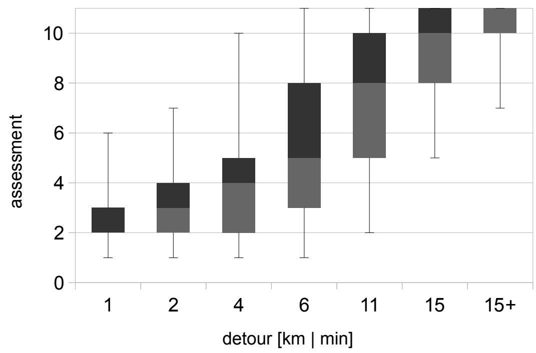
6.2.2. Results

Detour

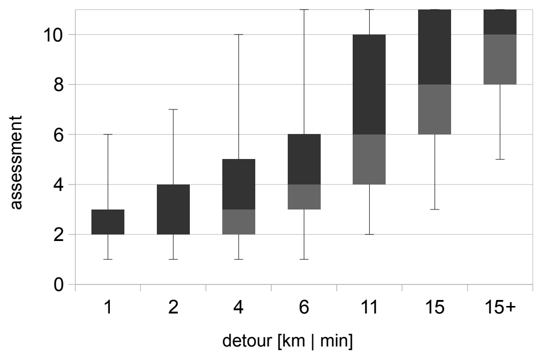
The perception of detour for a gas station depends on the preferences for criteria describing the quality of the gas station itself, e.g., price of gas or facilities. Figure 6.6 shows how the assessment for quality varies depending on the detour. 33 user data sets out of 42 are used for the figure because 9 are incomplete, inconsistent or faulty. The box plots show the median, the lower 25% quartile and the upper 75% quartile along with the absolute minimum and maximum for the assessments of the participants of POI B. Note that POI A has a predefined quality assessment of 1 ("does not correspond to my preferences"). Hence, an assessment of 3 ("corresponds to my preferences partly") for POI B means that POI B should be partly better than POI A. Assessment values above 10 represent unacceptable detour because it corresponds to the case that POI B should meet user preferences more than fully. Therefore, we classify the detour with a median rating of 10 as upper border of acceptance. This is the case for around 11 minutes for short routes, 15 minutes for medium routes and more than 15 minutes for long routes. For short detours of 1 minute, gas station B only has to be slightly better for short, medium and long routes (median rating of 2 or 3 respectively). The acceptance



(a) Route length: 20min or 20km



(b) Route length: 90min or 90km



(c) Route length: 200min or 200km

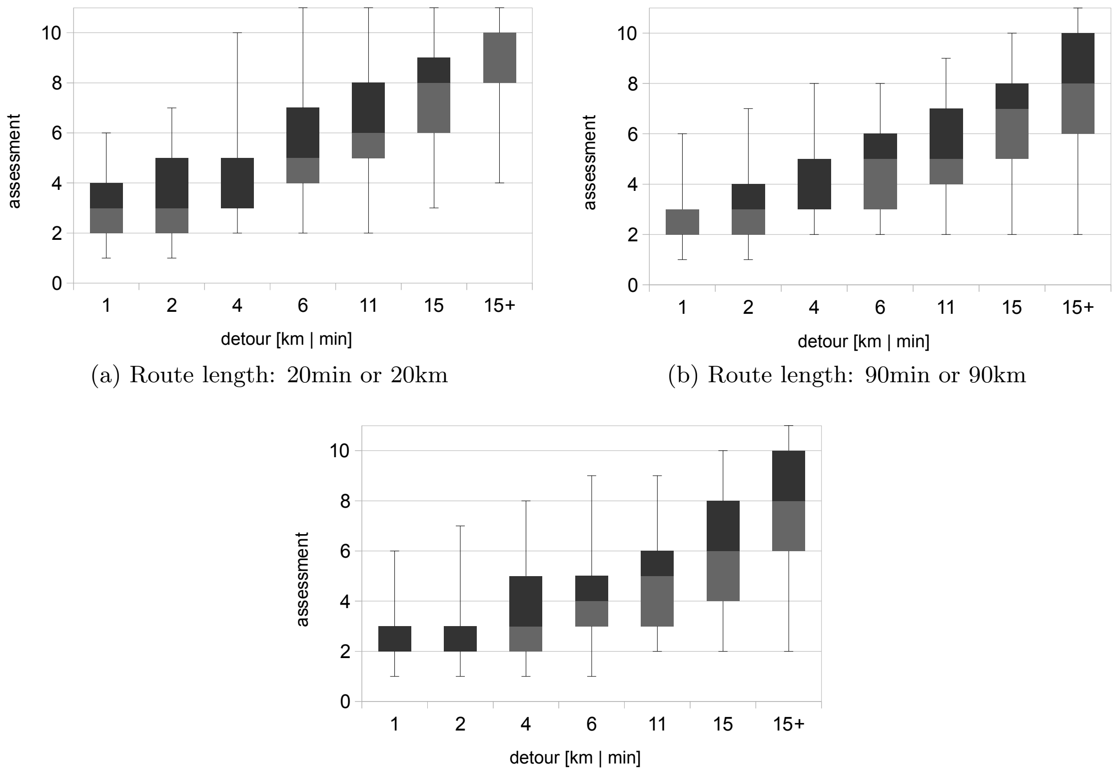
Figure 6.6.: Detour assessment of POI B representing a gas station

for longer detours varies much more. This can be seen by the range between the borders of the upper and lower quartiles. The longer the route is, the more participants are in agreement for short detours. This applies for up to 2 minutes for short routes until 6 minutes for long routes.

Our results of detour for restaurants are depicted in Figure 6.7. In this case, 12 user data sets are excluded from the analysis which leads to 30 used for the figure. It is striking that in all cases the median has a maximum of 10. This means that all kind of detour might be acceptable. The variance is lower for restaurants than for gas stations and the median of all values is also below the ones for gas stations. In contrast to gas stations, the difference between the route lengths is lower.

Decision Criteria

For the perception of detour, we use abstract assessments. However, we want to know which criteria are behind the assessments. For each type of POI, the participants could state their decision criteria in a free text form. This allows us to estimate if other



(a) Route length: 20min or 20km

(b) Route length: 90min or 90km

(c) Route length: 200min or 200km

Figure 6.7.: Detour assessment of POI B representing a restaurant

dimensions except detour have to be regarded for POI selection. 83% of the participants incorporate the price of the gas station in their decision. Available services (bistro, toilets or an ATM), the location of the gas station relative to the route and the brand are important for around 31% of the participants. The remaining gas level at the gas station that refers to the amount of gas that can be refilled at the gas station is of importance for 12%, the waiting time at the gas pump for 10% and the opening hours for 7%. For restaurants, there are three decision criteria that are mentioned by at least 50% of the participants (food quality, type of food, e.g., Italian, and price) and one which is close to 50% (service). In case of detour, 33% of the participants confirm that detour is important for gas stations. However, no participant mentions detour for restaurants.

Route Context Elements

In the previous section, we defined the elements of the route context. Here, we want to know how the participants assess the elements (Figure 6.8). The overall detour is the most important aspect with a median of 2 and 80% of the ratings in 1 to 4. The deviation of the route has a median of 4, hence it is rather important. The length of the route skipped for a POI is with a median of 8 rather unimportant. The participants are mostly not indifferent about the elements with around 10% in 5 and 6 for every element. The street type is with a median of 3 another important or at least rather important element of the route context.

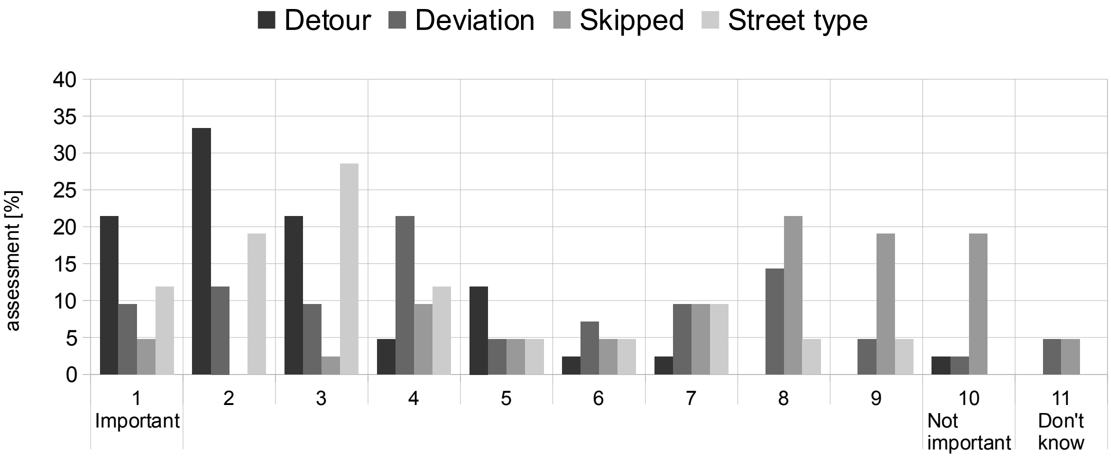


Figure 6.8.: Assessments of route context elements

6.2.3. Discussion

Our results show that the longer a route, the more detour is accepted. Especially in the transition from acceptable to unacceptable detours, the ratings vary among the users. The characteristics of the curves resemble slightly exponential or linear curves. This indicates that detours up to a threshold (the longer the route, the higher is the threshold) are perceived similar and above the threshold detour acceptability decreases faster. Compared to gas stations, more detour is acceptable for restaurants. Even more than 15 minutes is acceptable for all lengths of routes. In contrast to gas stations, driving to a restaurant nearly does not depend on the route length. Note that our example of a restaurant that does not correspond to user preferences is just a seldom case. Only if the users are hungry and it is the only place to eat, they may take it. The difference to a perfectly fitting restaurant is much higher than between a gas station that does not match preferences of the user and one that strongly match the preferences.

The decision criteria for each POI type have to be interpreted carefully. In general, the participants differently give answers in free text form. First, the participants who tend to write more also state more criteria. Second, mostly criteria that come to the mind of the participant are listed. Depending on how much they think about the question, different numbers of criteria are stated. Third, the participants tend to give obvious criteria and do not think about other criteria that unconsciously affect their decisions. Thus, criteria that are mentioned more often should be considered in any case by a recommender system. Other criteria that are mentioned sometimes need further research.

For gas stations, only two participants mentioned that they always use one particular gas station. That means that selecting a gas station is mostly a decision that is made ad hoc, not predetermined. Therefore, recommendations for gas stations can help to optimize this decision. The criteria also show that mostly hard facts such as price and detour are taken into regard. Regarding criteria for gas stations, the price is next to the detour a predominant dimension. Other criteria such as services, location, brand, remaining gas level at the gas station and waiting times may be of importance as well. Some of them are situation specific. For instance, an ATM is not always required when a gas station is approached. Decision making for restaurants seems to be more versatile. Next to type and price, also soft criteria, e.g., quality of food or service, play a role. As no user mentioned detour as a criterion, we assume that it plays less a role compared to gas station selection.

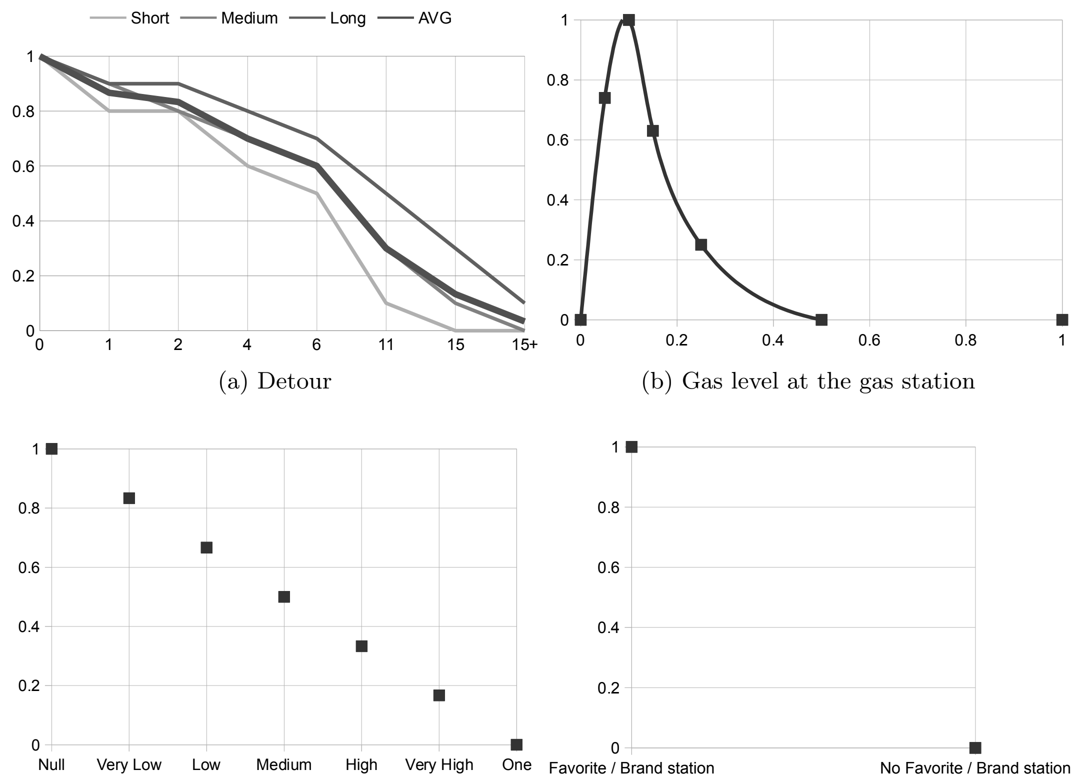
For the interpretation of the results for the elements of the route context, we should take into consideration that information such as the overall detour or the street type are much easier to imagine than other elements of the route context. The most participants state detour as important element. Street type is important as well. However, it is not clear how it is influencing participants' decisions because it cannot be measured with a numeric value. The deviation is only of minor importance but it can be used to differentiate routes with similar overall detour. A detour with lower deviation should be preferred. The skipped route can either be omitted or also be used to differentiate routes with similar overall detour and deviation.

6.2.4. Utility Functions Setup

Based on the user study, we derive four decision criteria that we want to use in our gas station recommender. The criteria need a representation as utility function to be integrated as an item feature on level 2 in our context integration model. The study shows that gas price and detour are mandatory criteria for gas station selection. Services are highly dynamic and divers. Therefore, we do not include them in our prototype. The same applies to the location of the gas station. Further investigation is needed to clarify which criteria define user preferences for the location, e.g., in town or out of town. This investigation is out of scope of this thesis. Brand and remaining gas level at the gas station are also integrated. The waiting time at the station is complicated to measure and opening times are used in prefiltering. Our utility functions can be seen in Figure 6.9.

Route

The user study shows the importance of the different route context elements. To derive a utility function for detour we use the formula in Equation 6.6.



(a) Detour (b) Gas level at the gas station

(c) Gas price (d) Brand

Figure 6.9.: Utility functions

The weight w of each element e in the route context where $E = detour|deviation|skipped$ is calculated with the $frequency$ of responses for scores $s_i \in [1, 10]$. It is normalized to receive weights in $[0, 1]$. The score s_i is squared to favor higher scores. We derive the weights $w_{detour} = 0,55$, $w_{deviation} = 0,32$ and $w_{skipped} = 0,13$ with Equation 6.6. These weights belong to the set W^{L1}. They are grouped to the feature "route".

$$w_e = \frac{\sum_{i=1}^{n} frequency(s_i, e) \times s_i^2}{\sum_{e_j \in E} \sum_{i=1}^{n} frequency(s_i, e_j) \times s_i^2} \quad (6.6)$$

For detour that is the most influencing part of the route context, we derive a numeric utility function $U_d(detour, trip)$ from the study (Figure 6.9a). It depends on the detour and the kind of trip (short, medium and long). As the aspects deviation and skipped route are of minor importance, a linear utility function is assumed for them. The values for U_d are the medians of the assessments of the participants. The median is less susceptible to outlier than the average. The longer the detour, the more the utility diverges. The average (AVG) of all kind of trips resembles a medium trip.

Remaining Gas Level at the Gas Station

The utility function for the gas level at the gas station $U_g(gaslevel, tank)$ is also numeric and depends on the remaining gas level at a gas station and the size of the tank (Figure 6.9b). These are the attributes on level 1 of our context integration model. The x-values are a percentage of the gas tank $\frac{gaslevel}{tanksize}$. This corresponds to the display of the current gas level on a car display. The drivers do not see the exact remaining amount in liter but the proportion of their tank. We derived the curve from the data collection in Section 5.2. The assumption is that the utility of a gas station regarding the remaining gas level at the gas station corresponds to the time of recommendation regarding the current gas level. It means that at the moment when the drivers need gas station recommendations most urgently, the utility of a gas station that is just around the corner would be the highest. Therefore, we take frequencies of subject responses with "Yes" (recommendation is useful) relative to the current gas level. The values are smoothed and supporting points for the utility function are extracted. For the curve in Figure 6.9b the supporting points $(0\%, 0.0)$, $(5\%, 0.74)$, $(10\%, 1.0)$, $(15\%, 0.63)$, $(25\%, 0.25)$, $(50\%, 0.0)$ and $(100\%, 0.0)$ are selected. A gas station with 50% and above and no remaining gas in the tank have no utility for the user. We receive the highest utility for 10% of the tank.

Brand

The utility of the brand $U_b(brand)$ is simply the availability of the property (Figure 6.9d). Either the users prefer large chains in general or a specific set of brands. For instance, the users have discount at a specific brand or they believe that the quality of gas is higher at large industry brands. The users are able to select one or more specific gas stations as their favorites. Favorite brands get the highest utility and all others no utility.

Gas Price

Gas prices are not generally available but several services on the Internet offer gas prices, e.g., *Clever-tanken.de*, *Oeamtc.at* or *Gasbuddy.com.* As these kinds of services collect gas prices in a community, we have to cope with outdated, missing or even wrong gas prices. Outdated and wrong values are handled by mapping gas prices in an interval with values "very low", "low", "middle", "high" and "very high". For missing values, an estimate is inferred from other attributes of the gas station, e.g., brand or location. This allows calculating with complete data. An interval is advantageous for estimated prices because a range can be estimated more accurate than a specific value. The interval-based utility function for gas price $U_p(price, gastype)$ maps the interval values to a linear increase of utility the lower the price is (Figure 6.9c). As the interval only covers the regular range of prices, we add "null" and "one" to react more appropriate on prices outside the interval range.

6.3. Evaluation

The evaluation comprises a performance analysis of our MCDM recommender in predicting the interest of a user in items. To collect test data, we carry out a user study. The participants have to assess gas stations in an online survey. We calculate scores on level 2 and 3 of our framework based on the utility functions we derived in the previous section. With common evaluation metrics described in Section 2.5.5, we compare our predictions with the ratings of the participants for the POIs.

6.3.1. Study Design

Because there is no test data freely available for our use case of a gas station recommender, we have to create our own test set. For this, we carry out an online user questionnaire in which the participants rate gas stations. We use real data for the setup to create a decision task as realistic as possible. The questionnaire presents four test sets with 10 gas stations each. The stations are characterized by the features detour (minutes), gas price per liter (Euro), the brand and distance to the gas station (kilometer). The distance to the gas station corresponds to the remaining gas level at the gas station. Some of the gas stations have no price information. The stations differ in these features. In all four scenarios, the current gas level is 6 liter (75 km remaining driving distance). The ratings are given on a scale from 1 to 5 (1=very bad, 5= very good). The participants are also asked to select one of the 10 gas stations that they would use to refill. Furthermore, they should rate on scale from 1 to 5 how complex this decision was. To get a user profile, we asked the participants for their preferences for the applied features of gas stations in the survey (gas price, brand, detour, gas level at the gas station).

35 participated in our study of which 32 are men and 3 women. 89% of the participants are between 23 and 34 years old, resulting in an average age of 28. 50% of them drive between 5.000 and 15.000 km a year, 30% between 15.000 km and 25.000 km and 20% drive more than 25.000 km. Most of the participants have experience with navigation systems (89%) and some of them also with driver information systems, e.g., a head-up display (37%). The preferences of the participants are similar compared to our previous study for detour assessment in Section 6.2.

6.3.2. Results

6.3.2.1. Example

The example in Table 6.1 with POI test set 3 gives an overview of the outcome of our approach. We use one of the user profiles with the normalized level 2 feature weights 0.33 for gas price, 0.0 for brand, 0.44 for route context and 0.23 for gas level at the

Gas stations				Calculated Scores					Sky-
Brand	**Price**	**Detour**	**Distance**	**Brand**	**Price**	**Route**	**Gas**	**Global**	**line**
Free		03:20	7.9	0	0.25	0.46	0.7	0.38	-
OMV	1.14	09:22	63.5	1	0.25	0.06	0.08	0.11	-
JET		05:07	8.2	1	0.25	0.3	0.7	0.29	-
ARAL	1.14	01:54	19.7	1	0.25	0.47	0.43	0.36	-
Free		04:18	8.3	0	0.25	0.35	0.7	0.32	-
Free		09:07	36.9	0	0.25	0.1	0.24	0.13	-
Free	1.09	06:40	50.2	0	1	0.15	0.14	0.45	×
ARAL	1.14	00:00	6.2	1	0.25	1	1	0.63	×
AGIP		09:00	63.3	1	0.08	0.06	0.08	0.01	-
AGIP		17:24	61.9	1	0.08	0.05	0.08	0.00	-

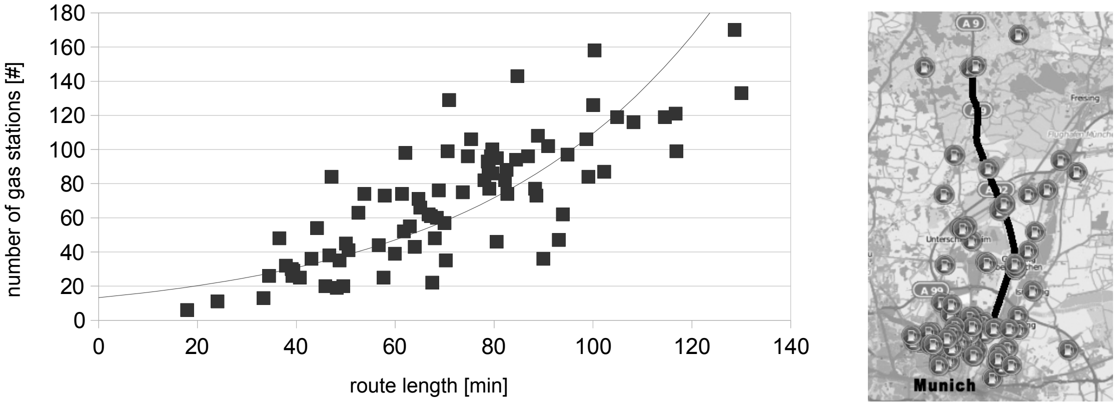
Table 6.1.: Example recommendation

gas station. This is a common user profile among the participants. We predict local scores with the MCDM method AHP on level 1 and global scores with TOPSIS without normalization on level 2. The Skyline filter has a fuzziness of 10%, $k = 3$ and no ω. As this user prefers gas stations without detour, we expect that the user chooses the one with zero detour. Regarding the global scores, this station would be exactly number one in a ranking with the highest value of 0.63. The Skyline contains two items. Besides this item with the highest global score, the Skyline also contains an alternative with low gas price because this user has preferences for price.

6.3.2.2. Prefiltering

We take a constraint c of 15 minutes for the detour in the prefilter. The value is derived from the utility functions study in the previous section. It is the maximum detour the drivers would accept for medium length routes.

Figure 6.10a shows the resulting number of POIs (gas stations) as a function of the route length. We generate 81 random routes around Munich. Routes through rural areas provide fewer gas stations as such through cities. Therefore, the number of prefiltered POIs varies across similar route lengths. For instance, there is a route that results in fewer than 40 prefiltered POIs and another route with more than 140 POIs at around 90 kilometers. The example in Figure 6.10b shows the difference between a rural and a city area. The northern part of Munich has a higher density of gas stations than rural area outside Munich. The prefilter around the route in the figure guarantees that only stations within a detour of a maximum of 15 minutes are shown.



(a) Number of prefiltered gas stations

(b) Example

Figure 6.10.: Resulting number of items after prefiltering relative to the length of a route for 81 different random routes (a) and one example for a route of 15 minutes [shown on *OpenStreetMap.org* maps] (b)

6.3.2.3. MCDM Recommender

To compare available MCDM methods (described in Section 2.3.3) for our approach, we calculate a global score sg with several combinations of the methods on the two levels (see Figure 6.4). We test the methods with and without normalization, i.e., to stretch the scores on the whole range $[0, 1]$ with a best score of 1. AHP only works with normalization because it uses comparison during calculation. WPM is not reasonable on level 1 because utility functions of 0 would lead to a score 0, even if there are other item features with higher weight (unless the weight for this feature is also 0). On the other hand, WPM can be useful on level 2 to cut off bad performing items for one feature. Overall, 27 combinations of methods on level 1 and 2 are tested. We discuss major evaluation metrics. Detailed numbers can be found in Appendix C.3.

Mean Absolut Error (MAE)

First, we calculate the mean absolute error (MAE) (Equ. 2.35). MAE is not the most important metric in our domain. We do not need to predict an accurate rating for all items due to the reduced information delivery in case of proactive recommendations, especially inside a car. Finding some good items is more important than to show the users how much they would like the item. On the other hand, predicted ratings could be valuable to make system decisions more comprehensible (see Chapter 7). In our test, the MAE values do not vary much over the POI test sets and also over the methods. For all tests, they are between 0.2 and 0.33 and most of the methods are below 0.27. It is striking that in the range 0.26 to 0.33 all nine methods use normalization on level 2. Methods on level 2 are TOPSIS, WPM and WSM. AHP is more accurate with the worst MAE of

0.22 on level 2, although its scores are normalized. Note that user ratings are discrete values in $\{0.0, 0.25, 0.5, 0.75, 1.0\}$ and the methods calculate continuous scores in $[0, 1]$. The MAE corresponds to a difference of one preference value (0.25) in average, i.e., if a user rates an item with 4, the system would probably calculate a 5. Furthermore, most of the methods calculate scores that are higher than user ratings. Especially normalized methods on the second level are always higher than user ratings.

Precision

The precision (Equation 2.37) is more important for our domain. Rhodes [Rho00] shows that a high precision should be preferred over high recall in the area of proactive information retrieval. Our relevance threshold is 0.65 including only good and very good rated items by the participants and in the prediction. That means that the true positives number tp counts items that have ratings larger than 0.65. In several tests, this threshold performed best for predicted scores, i.e., it results in the highest precision. This threshold also guarantees that only good (4) or very good (5) rated items are taken into regard because they correspond to 0.75 or 1.0 respectively in the range $[0, 1]$.

In the results, it is noticeable that normalized and regular TOPSIS performs completely equal on level 1. This is due to the structure of the attributes on level 1. Three out of four decision dimensions (price, brand and gas level at the gas station) only consist of one attribute on level 1. Hence, level 2 scores result only from the utility function for WSM, WPM and TOPSIS. AHP applies pairwise comparison for each attribute and therefore it acts different. Only detour consists of three attributes (overall detour, deviation and skipped route).

To build a bottom line for precision, we calculate 1000 times random scores for the features of the POIs. The calculation results in an average precision of the POIs with random scores of 0.25. Methods that have a lower precision than 0.25 are worse than taking random items. On level 2, normalized TOPSIS and AHP tend to vary less than the other methods with a variance lower or equal 0.008. Although the range of precision is high from 0.2 to 0.66, the methods perform differently depending on the combination. AHP on level 1 and 2 performs best with a precision of 0.66. AHP on level 1 and WPM without normalization on level 2 perform poorest with a precision of 0.20. In general, AHP performs best on level 2 with the worst precision of 0.60. Also WSM performs well with the worst precision of 0.53.

Mean Recommendation Error

In our application scenario of refilling, the users finally select one item because they need exactly one gas station to fill up. Therefore, we also ask the participants to select exactly one item out of the 10 available POIs in each test set. The participants are free to select any POI independent from the rating they gave to the POI. To assess how the selection fits to our prediction, we calculate a **mean recommendation error** (**MRE**) metric (Equation 6.7).

$$MRE = \frac{\sum_{i=1}^{N} R_{s_i} - R_{p_i}}{N} \tag{6.7}$$

The metric is similar to the MAE, except that only participant ratings are involved. The sum of the ratings for the item that is selected by the participant R_s and the item that we predict to be selected R_p (the item with the highest global score) is divided by the number of participants N. We receive MRE values in $[0, 1]$ because ratings of participants are normalized between 0 and 1. A MRE value of 0 means that a user selects an item with the same rating our predicted item has, even if our predicted item is not selected ($R_{s_i} = R_{p_i}$). Our assumption is that identically rated items have the same value for the user (independent from the actual selection of the user). A MRE value of 1 means that we predict an item to be selected that the participants rated the worst ($R_{p_i} = 0$). At the same time, there is at least one item that is rated best ($R_{s_i} = 1$). We exclude inconsistent selections from the investigation. Our assumption is that it makes no sense that the participants select an item that they have not rated highest. Hence, R_s is always larger than or equal to R_p.

All methods have a MRE between 0.13 and 0.17. This means that the predicted item either receives the same rating as the selected one in approximately 50% of the cases or a rating that is next best in the remaining cases. TOPSIS performs best on level 2 and all other methods are distributed across the ranking. The MRE is in average lower for POI test set 3 and higher for POI test set 4.

Irrelevant Items

We are also interested in how many of the items predicted by our method to be selected are rated as irrelevant (normalized participant scores of 0 or 0.25). That means we calculate the highest score for a POI that the participant rated as irrelevant. The amount of irrelevant items is between 6% and 10% over all methods. AHP performs worse than the other methods on level 2. The result for each POI test set correlates with the MRE.

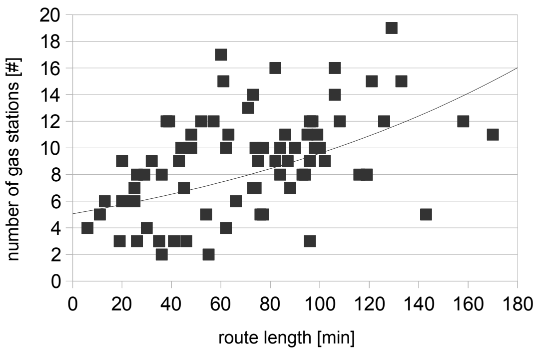
Hit Rate

Regarding the hit rates, i.e., the probability to find the item the participant selects, there is a clear distribution of the methods on level 2. Normalized and regular TOPSIS perform best with a probability of 0.47 to 0.49 to find the item selected by the participant. AHP is average with 0.41 to 0.44. WPM and WSM are the worst with a probability between 0.36 and 0.41. The only exception is WPM with AHP on level 1. This method has a hit rate of 0.46.

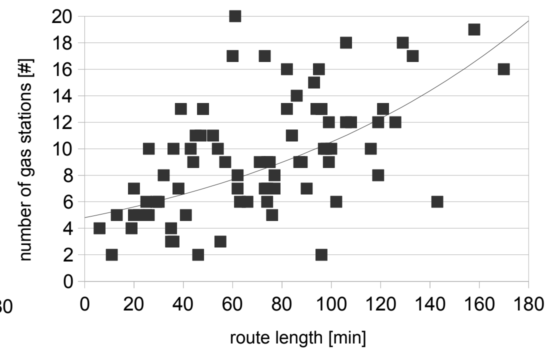
6.3.2.4. Postfilterig

For postfiltering, we test how well different settings of the Skyline filter perform in selecting items. This is comparable to the analysis of precision which we run for the MCDM recommender. The global score recommendation set, i.e., all items above the relevance threshold, can also be regarded as a postfiltering step of building recommendation sets based on a ranking. The Skyline filters vary in fuzziness, k and ω. $k = 2$ is combined with ω from 0.1 to 0.9. During testing, a k Skyline without weighting showed only reasonable results with profiles of equal weights for every item feature. Because in our application this case is seldom, we do not test it in detail. $k = 3$ is a regular Skyline and is tested with fuzziness from 0% to 50%. As a Skyline depends on the number of features, we fuse gas price and brand to one feature named quality. Hence, we only have one dimension that describes the quality of a gas station. The feature scores for price and brand are aggregated by their W^{L2} weights and the weight for the resulting feature quality is the maximum of both weights. On level 1, MCDM methods are used with normalization for AHP and TOPSIS and not for WSM.

Skyline Recommendation Set Size



(a) Regular Skyline on WSM



(b) Regular Sykline on AHP

Figure 6.11.: Comparison between the number of items for different types of Skyline filters relative to the amount of items prefiltered

For proactive recommendations, the amount of items in a recommendation set is crucial. We compare WSM without normalization and AHP with normalization as these methods use different kind of item scoring (single and pairwise) and differ in normalization. All methods with fuzziness 50% select too few items with either 0 or 1. The same applies to $\omega < 0.7$. A ω of 0.9 behaves like a regular Skyline. Figure 6.11 shows two examples and detailed results can be found in Appendix C.3.

We run the tests with the same 81 routes as for the prefilter in Figure 6.10a. The results show that AHP tends to select more items in the Skyline. Normalization cannot be the reason as scaling the score range does not affect the dominance condition. Like prefiltered items, Skyline recommendation sets also have a wide range. In Figure 6.11b, the Skyline selects two items in one case and in another case 18 at around 100 km. The amount of filtered items also depends on the prefilter. Generally, the fewer items are available after prefiltering, the smaller is the Skyline recommendation set. The size of the Skyline recommendation set again varies across different sized prefiltered candidate sets. It seems that the size of the Skyline recommendation set also depends on the structure of the items.

Loosening of the dominance condition significantly reduces the Skyline recommendation set. The size of the Skyline for WSM with a fuzziness of 10% is mostly below eight and for AHP below 11. In seven cases, the Skyline for WSM only selects one item. Regarding user preferences, $\omega = 0.7$ and $k = 2$ reduces the size of the Skyline recommendation set to mostly a maximum of five items for both AHP and WSM. The settings also more seldom select only one item for WSM. $\omega = 0.7$ and $k = 2$ seem to be a reasonable choice concerning user preferences. In the user study, for 31 participants out of 35, the weight of one feature is below 0.3 which makes this feature the less important one. Hence, two features remain with a summed weight higher 0.7. In 19 cases the gas level at the gas station, in seven cases the price or brand and in five cases the detour is the less important feature.

Skyline Recommendation Set Precision

We calculate the precision for the Skyline also with Equation 2.37. In this case, true positives tp are items in the Skyline recommendation set and rated as relevant by the participants (above the relevance threshold of 0.65). False positives fp are those items in the Skyline recommendation set that are not rated as relevant by the participants (below the relevance threshold of 0.65). Detailed results can be found in Appendix C.3.

The tested Skyline settings behave different across the POI test sets and have higher variance compared to the precision of the MCDM recommender. This underlines the dependency of the Skyline to the structure of the items. The precision for POI test set 3 fluctuates around 0.72 without difference across the methods and for POI test set 1 the range is from 0.0 to 0.571. The precision of the Skyline recommendation set is in average about 15% lower than for global score recommendation sets. The Skyline with $k = 2$ and $\omega = 0.7$ performs overall better (precision from 0.47 to 0.53) than the other Skylines. A ω below 0.3 shows bad results (precision below 0.43). This is related to the fact that either zero or one item is selected in this case. A fuzziness of 10% performs best (precision = 0.54) with AHP on the first level but performs also bad with normalized TOPSIS on level 1 (precision = 0.37). A fuzziness of up to 20% shows mostly reasonable results, whereas a fuzziness of 50% is already too high and shows bad results. The best performing regular Skyline has an average precision of 0.428. This is because the number of elements in a regular Skyline is high including medium performing items. Overall, the precision does not depend on the MCDM method on level 1.

It is surprising that $\omega = 0.9$ performs better than a regular Skyline. Due to the structure of the user profiles, all feature weights are taken into regard in this case. Hence, the number of items in the Skyline recommendation set is nearly the same as for the regular Skyline. The difference can be explained with the dominance condition with ω. It includes all features in the sum of weights with equal and better scores. However, with $k = 2$ it is enough that two features dominate and one is equal.

Recommendation Set Building

Finally, we investigate the performance of recommendation sets instead of single items. We compare between the global score recommendation set and a Skyline recommendation set with a relevance threshold of 0.65. For this, we create a recommendation set and test if the item that the participant selects is in the recommendation set. It corresponds to the probability that the selection of the user is included in the recommendation set.

Normalized methods on the second level perform better. WPM is an exception. It performs generally bad. AHP performs worse on the second level with an average probability between 0.71 and 0.79. Normalized TOPSIS and WSM show better results with probabilities between 0.83 and 0.95. For a Skyline recommendation set, a Skyline filter that tends to select more items (e.g., with high ω or a regular Skyline) performs best. The highest probability has a Skyline setting with $k = 2$ and $\omega = 0.9$ (0.857). Every other Skyline setting is in average below a probability of 0.58. Hence, the performance of a Skyline recommendation set is only comparable to a global score recommendation set with Skyline settings close to a regular Skyline. At the beginning of this section we showed that in this case the size of the Skyline recommendation set might become large. Other Skyline settings perform worse than global score recommendation sets in general.

6.3.3. Discussion

We are able to derive from the results of our analysis how drivers rate gas stations. The MAE indicates a difference of approximately one rating step in the score. Our system is mostly higher, especially for methods with normalization. Thus, the users seem to use highest scores less than the system. Furthermore, the users seem to act more in a comparative way because AHP is the most precise method. A comparative rating behavior means that a station that is rated good in one scenario may be rated bad in another scenario because more good stations are available in the last. Differences of precision values over the POI test sets underline our assumption as well.

Over all tests, AHP has the best performance on level 2. Especially in precision where AHP is also more constant over the POI test sets. Although the probability of recommending an irrelevant item is highest for AHP, the probabilities are in general still low. TOPSIS also performs better than WSM and WPM. It has the best MRE and hit rate. WSM is average and WPM shows the poorest results on level 2 in all tests. Results for the methods on level 1 vary across the analysis. Their evaluation is limited because there is only one feature with more than one attribute (route context). Further investigations may lead to clearer results.

The results show that it is unlikely that the users rate the highest ranked item (according to the global score on level 3) as irrelevant. It has either the highest rating that the participants used or the next best (according to the MRE), even if the hit rate is generally lower 50%. This means that it is possible to build reasonable recommendation sets of POIs in postfiltering with the global score. In addition, the probability to recommend an irrelevant item decreases with more items. An open question is whether one relevant item is enough to make the recommendation perceived as good. Maybe, the existence of irrelevant items has negative influence.

Recommendation set building with Skyline filters shows that only methods that loosen the dominance constraint produce reasonable amounts of items and good precision. On the other hand, these methods tend to shrink the recommendation set to one item in some cases. This is undesirable because the user has no choice any more. Skyline filters also tend to perform differently across different POI test sets. It is possible that the precision for a specific POI set is low, i.e., there are relevant items but we cannot find them. Among the loosening techniques, k and ω perform best. This is probably due to the incorporation of user preferences. For instance, if there are two important features d_1 and d_2, we can use $k = 2$ and $\omega = W^{L2}(d_1) + W^{L2}(d_2)$. Furthermore, Skyline is more unstable than a ranking, e.g., with AHP. It also depends on the structure of data. This is shown by the variance across POI test set 1 and the number of items in a Skyline relative to the number of items prefiltered. Finally, it is difficult to predefine the number of items solely with Skyline (e.g., a recommendation set of two or three items). Skyline techniques with loosening heuristics produce fewer items but also only one item in some cases. A regular Skyline needs more refinement of the Skyline recommendation set before it is presented to the user.

6.4. Summary

This chapter described our approach for context integration into recommender systems, a deeper investigation of the route as context and a final evaluation of different MCDM methods. Our context integration method is a combination of the well-known paradigms in context-aware recommender systems (CARS), prefiltering and postfiltering. Prefiltering uses simple cut-off heuristics in combination with filtering along the route to remove irrelevant items. Our knowledge-based recommendation approach is based on multi-criteria decision making (MCDM). It combines qualitative dimensions with contextual dimensions. The approach uses utility functions to assess items based on their attributes. These attributes either describe qualitative aspects such as the brand of a gas station or contextual aspects such as the detour that a gas station causes. To make heterogeneous attributes comparable, they are transformed into dimensionless score-based item features. Item features are decision dimensions of the users to select an item. User preferences are represented by a list of user weights for each decision dimension. Finally, postfiltering takes all assessed items and builds small recommendation sets that can be presented to the driver. In case of our P-IVRS, postfiltering is part of the decision making process. It combines assessed items with the situation awareness of the system. A recommendation set is not just built according to the performance of the items but involves the situations of the users, i.e., whether the item is useful for the users in their situations.

Our investigation involved a detailed analysis of the route as context. A user study shows how the drivers perceive detour towards gas station and restaurant recommendations. We define the route context as one important context parameter in our system. It contains the detour to a POI, the deviation from the route and the skipped route. The users assess deviation as important factor but less important than detour. The skipped route is of minor importance. Based on the results from the user study, we derive a utility function for detour depending on the length of the route. We also derive utility functions for further decision dimensions: the price of gas at a station, the brand and the remaining gas level at the gas station.

The main focus of this chapter was to evaluate different MCDM methods. We implemented our context integration approach with different MCDM methods and utility functions. A final user study collected ratings for gas stations which are compared to the prediction of the methods. We showed that comparative models such as AHP or TOPSIS mostly outperform the simple weighted model. The global score calculated by MCDM methods can be used to build small recommendation sets in prefiltering because the users seldom rate higher ranked items as bad. Besides calculating a global score, we applied dominance-based item selection methods in postfiltering (Skyline filter). Their results vary depending on the specific decision problem. We conclude that building recommendation sets with a global score is more suitable than with dominance-based filtering.

7. Recommendation Justification in P-IVRS with Explanations

Besides the competence to select good items, a P-IVRS also requires functional competence to make system decisions comprehensible for the driver. The challenge is to justify the decisions of the system because the system makes decisions automatically without user request. We apply explicit explanations to support comprehensibility. In this chapter, we present our approach to generate justifications for recommended items and for the reason why a recommendation is delivered. We use our gas station recommender to implement and evaluate our methods. As we do not know which arguments are relevant in our use case, a preliminary study is carried out. Justification is not the only explanation goal in context of P-IVRS. We classify several possibilities of explanations towards the user interface of a P-IVRS. However, the main focus of this chapter is to analyze explanations that justify system decisions. We describe our method for item explanations and carry out a second user study with an implementation of a proactive explanation interface. Finally, we describe our method for situation explanations and show example results.

7.1. Preliminary Study

The preliminary study should reveal the main requirements on generating arguments in our application scenario of a gas station recommender. Detailed numbers of the results are listed in Appendix D.1. In the following, we explain the most important results. An extract of the preliminary study was already published in [BKWL11].

7.1.1. Study Design

The user survey is carried out as online questionnaire. The participants have to rate different kinds of arguments and explanations on a 5-point Likert scale. We focus on properties of explanations which we described in Section 2.5.4. The questionnaire consists of 14 closed and one open question. In most of the cases, the participants receive different kinds of arguments and they have to rate them.

The first part of the survey tries to find out which arguments should be used in an explanation. First, the content of a single argument is investigated. The four item features that we already used in our context-aware gas station recommender are reused here: gas price, detour, brand and remaining gas level at the gas station. Also further information such as a present the participant gets for each refill, opening times or waiting times are used. We also investigate the usefulness of negative arguments in terms of bad item features in combination with positive arguments. Finally, negative arguments for gas stations that are not in the recommendation set are investigated.

The next part of the questionnaire is about the structure of an explanation, i.e., how many arguments should be in the explanation and in which structure. Few arguments may not be enough to understand the advantage of the item and too many arguments may provide useless information and cause information overload. Arguments can either be structured individually without reference to other items or comparative to other types of information or to other items of a set.

The third part investigates additional information as arguments. Besides explanations of the recommended items, the system may also deliver additional information to make its behavior more transparent. This can be the situation causing a recommendation, the current status of the system or additional information related to the source of information used in the recommendation process. Furthermore, we interview the users about additional arguments that contain the reliability of the data, the intelligence of the system and the context of the items. All of these sources bear more or less uncertainty. The uncertainty can be presented in form of likelihoods as degree of belief.

The rest of the questions deal with general aspects of recommendations in a proactive manner. We ask the participants how many gas stations should be recommended in different traffic situations. The situations range from challenging maneuvers in dense traffic to long-term standing still. The idea behind the question is to find out the expectations of the users concerning the amount of information they can handle in situations with different workload. We also ask the participants about other kinds of POIs for proactive recommendations in a car. Possible domains are classical item categories, e.g., restaurants or hotels.

Altogether, 89 participated in the online survey. Of these, eight questionnaires have been incomplete resulting in 81 data sets we use in our analysis. 70% of the participants needed 8 to 18 minutes to fill out the questionnaire with a peak around 11 minutes. The group of participants consists of 64 males and 17 females. 90% of the participants are between 18 and 38 years old. After excluding outlier outside this range, it results in an average age of 26 years. 80% of the participants have at least average experience with portable navigation devices, 63% with embedded in-car navigation systems, 37% with driver assistance systems such as head-up displays and 23% with additional in-car information systems such as BMW Assist. More than a half (51%) of the participants drives more than 10.000 kilometers a year.

7.1.2. Results

Supporting Arguments

The first question that we ask the participants is about information to use as supporting argument for a specific gas station. Supporting arguments underline the decision of the system for an item. The Likert scale for the answer ranges from *not useful (1)* to *useful (5)*. Arguments consisting of information based on detour, gas level at the gas station and gas price are rated as useful with a median of 5. A low variance indicates that the participants are in broad agreement. Slightly worse but still useful are arguments with the reachability of gas stations in general, i.e., how many stations are reachable with the current gas level. The waiting time at the station has a median of 3 but the participants are divided with this assessment. This is indicated by a variance of 1.16. Arguments containing the brand of the station receive low ratings with a median of 2. The participants rather agree in their assessment with a variance around 1 but approximately 66% give a rating of 1 or 2. Large uncertainty (variance = 1.42) can also be found in the assessment of a free drink for every refill. The median is 2. This makes the argument rather useless but 42% also rate the argument at least as medium useful.

A user who does not prefer a specific item feature would probably assess an argument with this feature as useless. Therefore, we investigate the correlation between user preferences and argument assessments. The highest positive correlation exists for arguments based on preference elements and the preference element itself. This means that participants who rate an item feature high also tend to rate the argument with that information high and vice versa. A low positive correlation also exists between price preferences and the free drink for every refill (both refer to a monetary aspect). High negative correlation exists between preferences for brand and arguments based on gas price and vice versa. A low negative correlation exists between preferences for gas level at the gas station and arguments with gas price.

Counterarguments

The next question aims to find out whether negative arguments are helpful. First, we investigate counterarguments. In each explanation for an item, a positive argument (supporting argument) is shown together with a counterargument. Only the combination of a positive argument with detour and a negative argument of brand is rated as useful (88% of the participants give a 4 or 5). However, the median of the argument with detour decreases to 4 compared to its individual use as single argument in the previous question. This even applies more to the combination of a positive argument for gas price and a negative for detour. The explanation receives a median rating of 3, whereas the gas price individually receives 5. Although the participants are not strongly in agreement (variance = 0.96), 70% give a rating of 3 or worse. The same applies to the combination of a positive argument for gas level at the gas station and a negative argument for gas price. The ratings are slightly higher, but overall the median decreases from 5 to 3. The ratings for a positive argument with brand and a negative argument with detour stay low compared to the brand individually.

Although the ratings for the explanations with a counterargument decrease compared to the individual use, the correlation between the positive argument and the user preference with this information remains nearly equal. Between the negative argument and its corresponding user preference, there is nearly no correlation.

Arguments for Items that were not Recommended

The second types of negative arguments are arguments for items that were not recommended. The participants are told in the description of the question that they have the possibility to retrieve explanations about gas stations that were not recommended by the system. High ratings (median = 5) with strong agreement (variance <= 0.4) are given to arguments with hard constraints. Items that were not recommended due to the opening times and due to the reachability with the current gas level is useful information for arguments. This does not apply to the availability of the type of gas "Super 98". We pretend that "Super 98" is the only type of gas which is possible for the car of the user in our study. The ratings for "Super 98" have a median of 3 but a high variance of 1.72 indicates that the participants are not in agreement. 75% of the participants rate the argument with 3 or lower. The argument with gas price again receives mostly positive ratings (75% of the participants give 4 or 5).

Following the question about arguments for items that were not recommended, we ask the participants whether they think that this kind of explanation is useful in general and should be provided on request. The explanations are perceived as rather useful (median = 4) but the participants are not strongly in agreement (variance = 1.16). 86% of the participants believe that the explanations are at least medium useful (rating of 3) or more useful.

Number of Arguments

Besides the content of arguments, we also want to know how many arguments are appropriate in an explanation for gas station recommendations and whether it makes a difference what kind of information is used. Therefore, we present the participants explanations with a different amount of arguments which they should rate on a scale from "too few information" (1) to "too much information" (5). The results are depicted in Figure 7.1.

A rating of 3 represents not too much and not too less information (i.e., exactly the right amount). Explanations consisting of one argument comprise arguments with gas price or detour. The median is a rating of 2, i.e., rather too few arguments are presented. 66% of the participants rate one argument with 2 or worse. There is a low correlation of

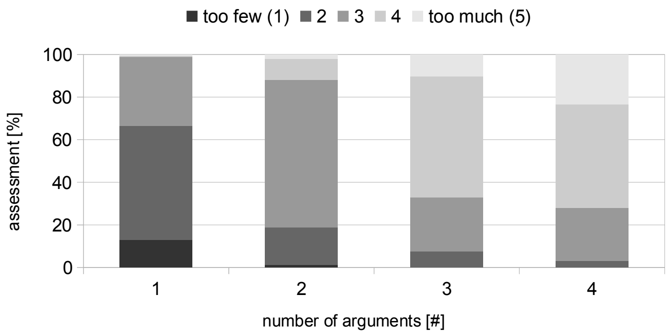


Figure 7.1.: Assessement of the number of arguments

0.26 between preferences for gas price and the rating for the explanation with gas price and no correlation between preferences for detour and the corresponding explanation. Explanations containing two arguments are either a combination of gas price and brand or gas price and detour. 69% of the participants rate these combinations as exactly right. The remaining 31% are distributed over "too few" and "too much". Explanations with more than two arguments have a median rating of "too much". Four arguments are slightly more "too much" but already three arguments are rather "too much".

Combination of Arguments

For the investigation of the most appropriate type of combination of arguments, we present three different types of combinations each with the same type of arguments (price, detour and brand). Comparative combinations are able to contain more information than individual combinations because they can make a statement about the domain in general, e.g., "This gas station has the lowest price in the area". The results show that the participants are not in agreement in their ratings and that there is no combination that is clearly better. All combinations receive a medium rating of 3. Individual combination is rated slightly higher with 85% of the ratings above 3. A comparative combination to an average has 72% and comparative to the set has 67%.

Regarding the correlation between the combinations, each correlation is positive but not high. Individual combination and comparative to an average have a correlation of 0.28, individual and comparative to the set a correlation of 0.11 and the two comparative combinations 0.26. This means that the participants who rate one combination high do no necessarily rate the other combinations low.

Situation-aware Explanations

We investigate the delivery of additional explanations concerning the situation in which a user is and how this affects the recommendations. For that, the participants should imagine a situation consisting of a trip to an urgent meeting where the destination cannot be reached without refilling the tank.

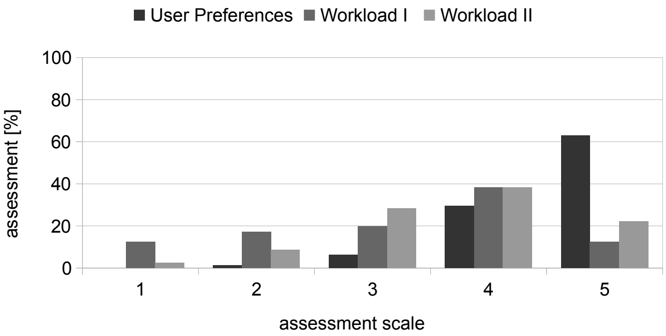


Figure 7.2.: Assessment of situation-aware explanations: Adaptation of user preferences and reduction of workload

In Figure 7.2, the ratings for the three explanations are shown. The explanation concerning user preferences comprises the adaptation of preferences to avoid loss of time due to an urgent meeting. The other two explanations describe the delivery of recommendations because of a situation with workload. Workload I refers to low workload while standing still at a traffic light. In Workload II, the driver faces high workload while handling a complex traffic situation. Situation explanations are overall rated at least as rather useful with a median of 4 for the workload and 5 for the user preferences. The situation with the complex traffic situation is slightly rated more useful as the traffic light. The participants are in agreement that explanations of the adaptation of user preferences are useful in this situation. 91% gave a rating of 4 or higher.

Status Explanations

Status explanations describe actions of the system and its decisions. Explanations containing the decision and its resulting action receive the best ratings in this question with a median of 4. Although the assessments of the combination of gas level at the gas station and the retrieval of gas stations vary (variance = 1.12), most of them gave at least a rating of 4 (80%). The agreement to the reachability of gas stations and their retrieval is slightly lower but still positive. 64% of the participants give a rating of at least 4. On the other hand, status information without the reason of the decision receives much lower ratings. The explanation that describes the retrieval of gas stations itself is rated as not useful (median = 1) and the explanation about a positive search result receives a median rating of 2. The agreement with the last explanation is low (variance = 1.75) but only 30% of the participants rate this information as rather useful or better. The rest of the status explanations including the up-to-dateness of the gas prices and no search results are rated with a median of 3 but the agreement is low (variance>= 1.75).

Reliability of Data

Besides additional explanations including the situations of a user and the status of the system, we also ask the participants about the reliability in the completeness of the data, the matching between user preferences and the results and context information used for reasoning. To make the participants clear what is meant with the arguments, we give some examples. The example for completeness of data refers to the number of gas stations known by the system. For context information, the reliability of gas prices known by the system is the example. The assessment of the participants depends on the source of reliability. Completeness of data is rated as rather useless and context information is rated as rather useful. User profile matching is in between with a median of 3 but the variance is high.

Degree of Proactivity

Additional explanations can either be delivered automatically, e.g., with the recommendation, or requested by the user on demand. We ask the participants to give their opinion towards proactively delivered explanations of situations, status information and reliability on a scale from "automatic" (1) to "on demand" (5). The median of ratings for situation specific explanations is 3 but the variance is very high (2.05). 88% of the users are not indifferent between automatic and on demand. The same applies to other types of additional information. 91% of the participants are not indifferent concerning status information and 85% concerning reliability. Status information is requested to be rather automatic with a median of 2 and the reliability is requested to be on demand with a median of 4.

Further Results

The question about how many items should be delivered proactively shows that for most traffic situations three items are reasonable. In standing still situations, up to five items can be delivered. In complex situations, the participants request one or two items. Finally, some participants mention that information overload is annoying and explanations should not lead to more irrelevant information.

7.1.3. Discussion

The results concerning the content of an argument suggest that the perception of arguments depends on the perceived utility of the item. An item that is perceived as useful receives in general a better rating for the arguments. This aspect can also be seen in the assessment of counterarguments. Even if both arguments are relevant considering user preferences, negative explanations receive a much worse rating compared to the individual use of the positive argument. The more important the negative argument is, the worse the rating becomes. Actually the explanation should help to make a decision, even if the decision is against the recommended item. Because the participants probably do not like the expected item behind the explanation, the explanation is rated worse. However, it can be confirmed that the content of an explanation strongly depends on user preferences. The correlation between user preferences and supporting arguments is high, especially in case of user preferences for the brand of the station. These preferences are in general low but an argument with brand is useful if the users prefer a specific brand.

The usefulness of counterarguments itself could not be derived. The availability of a counterargument leads to worse ratings of the explanation. It may help the participants to make a decision. It is also notable, that the user preferences only correlate with the positive argument. This may indicate that decisions are made based on the first argument. Counterarguments only influence the decision. Furthermore, negative arguments for items that were not recommended are overall desirable. They are more popular if they are easy to understand (e.g., hard constraints such as opening times).

Concerning the structure of an explanation, we believe that two arguments are the perfect size for an explanation. This coincides with the two major item features that the most participants use to select a gas station: price and detour. It is also notable that the second argument in the explanation does not necessarily need to be a high rated item feature in the user profile. Note that this number of arguments does not need to be valid for explanations during driving because the workload is different there. The combination of arguments in an explanation does not seem to play a role in the construction of explanations. The participants rate all kinds of combinations mostly indifferent and slightly as useful. High ratings for one kind of combination do not imply low ratings for the other. This can be seen in the positive correlation. The simplest structure receives slightly better ratings than more complex ones. It indicates that the participants would prefer less and simpler information over more information. Overall, it seems that the selection of the structure is less crucial than other aspects.

In case of incorporating users' situations in an explanation, the arguments that deliver the most benefit to the task are rated best. The explanation for the situation with the user preference adaptation also covers another aspect. It explains the decision why the system carries out an action (in this case, the action "user preference adaptation"). Arguments concerning situations with the workload of the driver can also be useful in an explanation. The usefulness of situation-aware explanations can also be seen in the ratings for status information. Related explanations that do not contain the reason for the action are rated much worse than an explanation that describes the reason for the action. We also observe that status explanations have a positive correlation between their content and user preferences. Furthermore, the participants are not interested in general status information such as gas prices (e.g., downloading of current gas prices). This also applies to reliability. Overall, the usefulness of reliability information cannot be derived uniquely. The ratings are distributed over the whole range. We believe that the usefulness of this kind of information can only be tested with a prototype system.

Finally, our findings concerning pushing or pulling additional information are clearer than other results. As only a few participants are indifferent at this topic, we suggest making pushing and pulling configurable. Situation explanations can be pushed by default and status information can also be shown by default but the reliability should be retrievable on demand.

7.2. Explanation User Interface(UI)

The different types of explanations we investigated in the preliminary study should be provided to the driver in an appropriate way. These explanations need to be delivered by an in-car user interface.

7.2.1. Ramping Strategy

We use a ramping strategy like Rhodes [Rho00] to explain proactive recommendations, i.e., explanations are distributed over several levels of detail. Rhodes shows that ramping interfaces are an effective way to deliver pushed information. We particularly agree to Rhodes's view that information overload should be kept low in the beginning of the recommendation process. Additionally, it should be controllable by the users. If the users are not interested in recommendations for any reasons, they should have the option to ignore a recommendation. Furthermore, a proactively delivered recommendation can contain false positives. This should be anticipated when using an intelligent system. In an automotive scenario, false positives can cause additional workload which has to be kept at a minimum. A ramping strategy is able to cope with these circumstances. If the users are interested in explanations, then they should have the possibility to retrieve as many information as needed to make a decision. The more important a further detail is, the easier it should be to retrieve it.

7.2.2. Stages of an In-Vehicle Ramping Interface

Table 7.1 shows the stages of our ramping interface for explaining in-vehicle proactive recommendations. Note that not all explanations are necessarily delivered proactively. The proactive behavior of the system depends on the implementation. We classify the stages by cognitive effort to consume the explanations, their explanation goal and the amount of user involvement. Cognitive effort becomes higher with more information and if the user has to make a choice.

The first stage only indicates that the system has no recommendations, e.g., with an icon or if the system stays inactive. The cognitive effort to consume this information is very low because the user does not have to make a decision or think about provided information. The second stage allows the system to provide status information to make

Stage	Explanations	Cognitive effort	Goal	Involvement
1	No recommendations	very low	Transparency	No
2	Status information	very low	Transparency	No,Once
3	Why something was recommended	low	Justification	No
4	Benefits of items	medium	Justification	No,Once
5	Recommendation situation	high	Transparency	Once
6	Item details	high	Effectivity	Once
7	Explained items	high	Transparency	Once
8	All items	very high	Scrutability	Steady

Table 7.1.: Stages of a ramping interface for proactive recommendations

clear why the system is behaving that way (Transparency), e.g., why it has no recommendations or what it is doing right now. Low cognitive effort is imposed by this stage because no decision has to be made.

In the stages 3 to 8, a recommendation is presented to the driver. Stage 3 aims to provide a justification for the decision of the system concerning the recommendation, e.g., it describes why a situation caused the recommendation. This explanation has low cognitive effort because it should only help the driver to decide if the recommendation is relevant. It is important to keep this stage with low effort to minimize negative effects of false positives in an early phase of the recommendation process. In the next stage, the recommended items are presented with their benefits. The presentation comprises explanations with a set of arguments. It should persuade the users with a justification that the items are useful for them. Cognitive effort is classified as medium because the driver has to make a trade-off based on the features of the items. Again, the risk of false positives has to be kept to a minimum because the user has not shown interest for the items yet, only for the recommendation in general. Therefore, not all details but only necessary information is provided. In stage 6, more details about the items are presented to give the user the possibility to make an effective decision. This is the first stage that contains the whole details of the items. A lot of information may be involved in this stage. The information has to be compared among the recommended items. Therefore, this stage is classified with high cognitive effort. The same applies to stage 7 which supports stage 6. It supports the choice of the system for an item with additional explanations.

Finally, the system may deliver irrelevant items but the users are still interested in an item. Hence, the users should have the opportunity to access all available items in stage 8. This is classified with high cognitive effort because not only a small set of items is compared but a large one. Depending on the implementation, this stage may contain a detailed list of items or only descriptions, e.g., the benefits of other items.

Another characteristic of the ramping strategy is the amount of user involvement. The more user involvement it needs, the more interruption is caused. We believe that stages 1 to 4 can be realized without any involvement of the user, i.e., the user does not have to interact with the system by means of an input device. The involvement in stages 2 and 4 depends on the application. Status information and a short explanation of item benefits may be requested on demand. All further stages should only be accessible on demand because of their higher cognitive effort and the risk of false positives. The retrieval of information requires user interaction only once, e.g., details of an item. Browsing through a large set of items requires steady interaction (stage 8).

7.3. Item Explanations

Although all stages of the explanation interface for proactive recommendations are able to deliver useful information to make the behavior of the system more transparent, we focus on explanations for the reason of a recommendation (stage 3) and the benefits of the items (stage 4). Within these stages, we focus on item explanations. The design of the corresponding user interface is the topic of the next chapter. In this section, we describe our method for item explanations. It is designed for a small set of recommended items because the preliminary study showed that only a few items should be delivered proactively. Our argument selection method makes use of the output of the context-aware recommender described in Chapter 6. However, it is designed independently from an underlying recommendation method. According to Vig and Riedl [VSR08], this aspect classifies our explanations as justifications in contrast to make the system more transparent. Our approach requires a user profile with weights for item features, a global score for the overall performance of an item and local scores for the performance of each item feature as input. The output is one or two arguments for the set of items to be explained with the supporting argument in the first place. We already published the following method in [BKWL11].

7.3.1. Argument Selection

The arguments for the items are structured independently, i.e., no comparative explanations are used. The preliminary study shows that it makes no difference for the user but an independent structure enables shorter explanations. We use preference- as well as context-based arguments, starting with a supporting argument in the first place and adding a second one if necessary. The idea is that if an item is recommended, there has to be one significant feature that causes the selection of the system. This should be at the first place. If items in the set cannot be distinguished by the first argument, a second argument is provided.

Argument Strength

We generate arguments for items based on the context-aware recommender system for gas stations presented in Chapter 6. The recommender uses multi-criteria decision making methods (MCDM) to assess items I on multiple item feature dimensions D by means of utility functions. For example, these are price or detour of a gas station. First, the context-aware recommender aggregates all item attributes and context (level 1) belonging together to local scores $LS_{I,D}$ in the range $[0, 1]$ (level 2). Each score represents an item feature D. On level 3, all features are aggregated to a global score GS_I. The users are able to set their preferences for the item features explicitly which results in a weight w_D for every feature D. Note, with the performance of an item as global and local score as output, it can be any recommendation approach to produce these scores.

Our argument assessment uses two additional scores. The explanation score $ES_{I,D}$ describes the explaining performance of an item feature and the information score IS_D measures the amount of information of an item feature. The explanation score is calculated by multiplying the weight of a feature w_D with the performance of the item I in that feature (Equation 7.1).

$$ES_{I,D} = LS_{I,D} \times w_D \tag{7.1}$$

With this score, bad performing features as well as aspects not important for the user are neglected. The score corresponds to the product of user interest in a feature with the contribution of an argument to that feature described by Felfernig et al. [FGL+08]. Instead of a predefined argument phrase, we measure the performance of the feature. The problem of only using this score is that if every item performs well on a feature and this feature is important for the user, every item would be explained by the same argument. This decreases the opportunity of the user to make an effective decision because items are not distinguishable. Therefore, our information score measures the amount of information of a feature relative to the item set it belongs to (Equation 7.2).

$$IS_D = \frac{R + IN}{2} \tag{7.2}$$

The value $R = max(x) - min(x)$ is the range of the feature value x in the set for feature D. IN can either be Shannon's entropy (Equation 2.26) or simply $IN = \frac{n-h}{n-1}$ where n is the number of items in the set and h is the frequency of the most frequent feature value x in the set. Taking the feature value $x = LS_{I,D}$ is a good choice if local scores have a small value range, otherwise a utility interpretation of $LS_{I,D}$ would be better. A utility interpretation transforms the crisp score value into an interpreted value such as "low". The information score is low if either all x are similar (R is low) or same x appear frequently (IN is low), e.g., all gas stations are average priced.

Interpretation

Information for arguments in an explanation can either be interpreted feature values, e.g., current gas level is low, or facts, e.g., current gas level is 32 liter. Hereby, every feature is described by one attribute from level 1. For instance, the route context is described by the detour because it is the most important attribute in that feature. An interpretation is a mapping from a specific value to a discrete interval. We use a generic discrete interval with $\{One, Very\ High, High, Medium, Low, Very\ Low, Null\}$ to map values to a discrete value. Two kinds of values can be mapped. A utility interpretation maps the utility of an item. For instance, a current gas level of 32 liter at a gas station can be mapped to $Null$, because most people do not refill at this level. Therefore, the utility is 0 for that feature. The assessment of the utility is the task of the recommender system. The utility is represented as local score. Interpreting the attribute and context values leads to different results, e.g., a current gas level of 32 liter is $Medium$ if the tank has a capacity of 65 liters. This is called attribute interpretation.

7.3.2. Explanation Process

The scores from the recommendation process as well as the additional scores for argument strengths are used in the explanation process to build the final explanation. Figure 7.3 shows the process to select arguments. It follows the framework for generating explanations described by Carenini et al. [CM06] (see Section 2.5.4). The framework divides the process in the selection and organization of the explanation content and the transformation in a human understandable output.

Content Selection

In the content selection, our argumentation strategy selects arguments for every item I separately. A positive argument is selected first to help the user to instantly recognize why this item is relevant. For this, the best performing feature D based on the explanation score $ES_{I,D}$ is compared to the threshold α (1). A value larger than α means that the dimension is good enough as a first argument. The threshold α should be chosen so that the first argument is positive. If no feature is larger than α and thus no first argument can be selected, we look at the global score GS_I (2). If this score is larger than β, then the item is a good average, otherwise we suppose that the recommender could not find better alternatives. With a first argument, we look at the information score IS_D of the corresponding feature (3). A small information score (lower than γ) means that this feature provides low information. Therefore, a second argument is selected by means of the explanation score. The explanation score $ES_{I,D}$ of the second argument has to be larger than μ to make sure that the second argument is meaningful enough (4). In general, the threshold for the second argument is lower than the threshold for

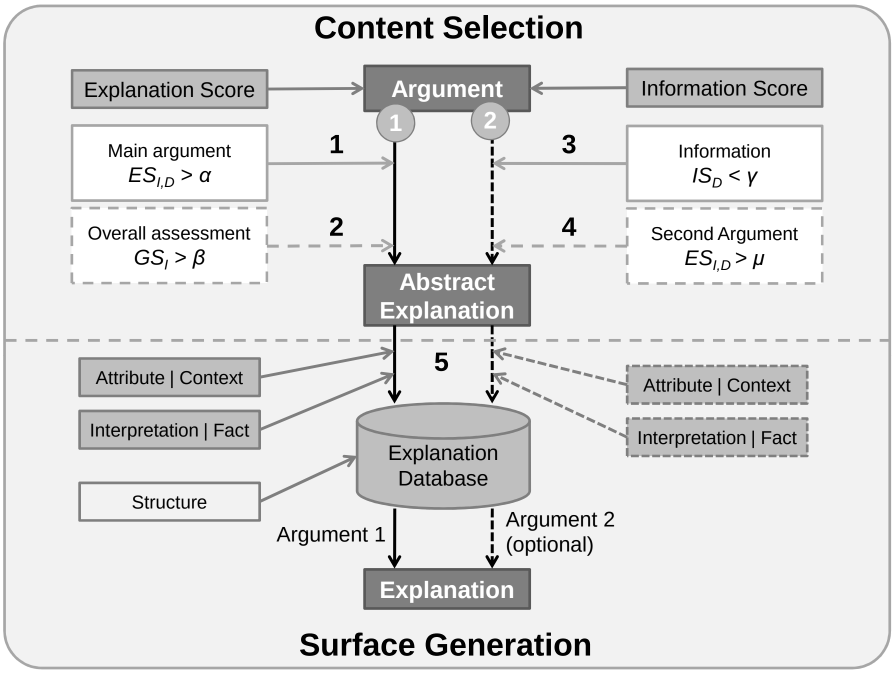


Figure 7.3.: A process to generate explanations by analyzing scores (content selection) and building the explanation (surface generation)

the first argument ($\mu < \alpha$) because the requirements on the second argument are lower. With the thresholds μ and γ, the amount of information can be controlled.

Surface Generation

The result of the content selection is an abstract explanation that needs to be resolved to something the user understands. This is done by the surface generation. We map a key value pair, e.g., $(gaslevel, low)$, to human understandable information, e.g., textual phrases or icons (5). Either facts or attribute interpretations can be used as values. Human understandable explanation information is uniquely stored in a database, e.g., in XML format. Also the structure of an explanation (icon, independent phrase, comparative phrase etc.) can be defined here.

7.3.3. Discussion

Argumentation theory as described in the guidelines of Carenini and Moore [CM06] and Toulmin's [Tou03] model suggests a specific order of arguments in an explanation to support the claim effectively. We ignore this order in our explanations for items because we are only able to use one or two arguments for each item due to information overload. With this exception, our approach follows the guidelines by delivering a strong argument at the first place, supporting this argument with a second argument if necessary and including negative arguments only optionally.

Our method for item explanations is based on the framework for generating explanations of Carenini and Moore [CM06]. The strength of an argument is the most important aspect in the framework. Carenini and Moore call the strength compellingness. We measure the strength with the utility of an item feature and user preferences for that feature. Our approach resembles the work of Felfernig et al. [FB08]. Felfernig et al. use weights for predefined explanations and user preferences towards item features. Their method orders arguments according to assumptions from argumentation theory. In contrast, we measure the argumentation strength of an item feature at the run-time and retrieve a predefined phrase for that feature.

Another difference between our approach and approaches like that of Felfernig et al. [FB08] is that our explanations are targeted to the users of a proactive mobile system and do not want to persuade the user to buy products. Product explanations should be comprehensive enough to persuade the user, whereas item explanations for proactive systems should be short enough to avoid information overload. In contrast to products, the use cases for automotive scenarios impose high physical involvement instead of high financial involvement and the time to make a decision is short.

We agree with Baltrunas et al. [BLPR11] that explanations in a mobile scenario have to be very short and easy to grasp. We share the same goal to justify system decisions with short explanations. However, our preliminary study shows that one argument as proposed by Baltrunas et al. [BLPR11] is not enough to make a satisfying decision for an item in our use case. Note, the use case of Baltrunas et al. and our use case are different. Tourists consume a series of attractions in contrast to drivers who only need one gas station at a time.

Finally, it could be useful to make comprehensive explanations available for the users who want to get a deeper understanding such as described by DeCarolis et al. [dCNPG09]. However, this kind of explanations cannot be presented to a driver while driving because of information overload.

7.4. Evaluation of Item Explanations

We evaluate the presented method for item explanations with a PC prototype in an offline setup. The subjects of our experiment are shown different scenarios of gas station recommendations. They have to assess the explanations that are provided along with the recommended items. We published an abstract of the evaluation with the major results in [BKWL11].

7.4.1. Evaluation UI and Deployment

We carry out the evaluation with a prototype of a gas station recommender on a PC. The subjects have the possibility to interact with the system with regular input devices (mouse and keyboard). They sit in front of a 19″ monitor. The PC interface is shown in Appendix D.2.

The evaluation interface implements our explanation UI concept (Table 7.1) in three levels of detail. The users can assess each of the levels separately. Stages 1 and 2 are omitted here. Stage 3 is represented with artificial information and stage 5 is also omitted. The first level represents a pop-up view that is shown to the user without request as soon as the system delivers its recommendations. A map is shown next to the pop-up. The pop-up comprises situational explanations in the header (stage 3), the list of items with explanations (stage 4) and buttons to enter the other levels or to zoom to the POI on the map. The map view shows the active route and the recommended items that can be matched to the items in the list by colors. It is based on a street map from *OpenStreetMap.org*. We implemented a route planer to be able to calculate real routes. In general, the whole route of the driver is shown. Zooming to a POI changes the view area to the selected POI and the detour from the route to the POI. The mouse can be used to browse in the map or to change the zoom level.

The second level represents the detail view. It gives the user the option to retrieve more details (stage 6) for the recommended items and all explanations (stage 7). This can be useful if a decision for a gas station cannot be made based on information in the first level. Besides the address and the name of the gas station, information about the item features is listed and the facilities that the gas station offers.

The third level represents the list view of all available gas stations along the route (stage 8). If the recommended items do not fit to users' expectations, they are able to retrieve a list of all stations along the route and order them by their feature or their names.

7.4.2. Study Design

We define six scenarios that are handed to the subjects along with a description. The scenarios cover in the area of Munich different lengths of routes with different intentions, e.g., work, vacation or visiting friends. The order of presenting the scenarios to the subjects is random. The scenarios are listed in Table 7.2. A detailed description of the scenarios can be found in Appendix D.2.

In all scenarios, the users are told that they cannot reach the destination without refilling and therefore a gas station recommender pops up proactively. The task of the subjects is to select a gas station. The pop-up shows a list of items with interpreted item features in the explanation. If more information is needed for a decision, the subjects are allowed to enter another level after selecting an item and indicating how satisfied they are with the decision. The satisfaction is given on a Likert scale from 1 (satisfied) to 5 (unsatisfied).

Scenario	Description	Route	Duration	Length
A, B	Ride to work	Gauting - Munich, Rosenheimer Strasse	30 min	21 km
C	Visiting friends	Gauting - Dingolfing	75 min	122 km
D, E	Visiting friends	Gauting - Bad Grönenbach	75 min	119 km
F	Vacation trip	Königsbronn - Chieming am Chiemsee	180 min	260 km

Table 7.2.: Routes used in the scenarios

Additionally, the subjects have to answer questions. We ask them questions about relevant, irrelevant and missing information in the view, the reason and satisfaction of their decision and why they want to enter another level. In the beginning of the study, they are introduced to the system and its interaction possibilities. After completing all six scenarios, a follow-up questionnaire is carried out including the number of items and arguments and whether interpreted or fact explanations are preferred.

We measure whether the system successfully justifies the items by interviewing the subjects about their satisfaction with their item selection in the first level and whether they need more information. If the subjects enter more detailed levels, explanations are not effective. If level 3 is entered, the items themselves are not relevant.

The interpretability of the study is limited by the offline setup. This does not take workload while driving into account. As described by Tintarev and Masthoff [TM11], the underlying recommendation approach and the recommendations that are delivered significantly influence the perception of the explanations. Furthermore, the map is a crucial element for POI selection and therefore it also has an influence on the assessments towards the explanations of the items. The study is based on a database with all available gas stations but only for one third of them gas prices are available. Hence, explanations without price do not necessarily mean that the system has classified the price as irrelevant but that the price is not available.

We had 20 participants with an average age of 27 in the range of 22 and 33 with two older subjects (38 and 45). The group comprises 17 males and three females. The participants were recruited from our research lab. 10 of them drive more than 10.000 kilometers a year, five between 5.000 and 10.000 and the rest less than 5.000. Three of them do not have a car. The participants have the possibility to set up a user profile at the beginning of the survey. The importance of the item features price, brand, detour and gas level at the gas station can be given (like in the studies we carried out in the previous chapters).

7.4.3. Results

We discuss the results for each scenario separately. Detailed values of the survey can be found in Appendix D.2.

Scenario A: An Additional Need

The subjects are rather satisfied with their selection in the first level (median of 2) but most of them wanted to know whether the station offers an ATM (due to the situation). 15 out of 20 subjects state that the situation influences their decision. Hence, 16 out of 20 subjects enter level 2 to look for the ATM. Their satisfaction increases significantly after the second level to a median of 1. None of the subjects needs to see the list of all stations. Arguments with price and detour are assessed as relevant in 14 cases for price and in 13 cases for detour. The brand is assessed by six subjects as relevant which is more than the subjects who state that this is an important feature. Additionally, also five subjects miss the brand. The gas level at the gas station is seldom used in the explanation and also not missed by the subjects. As an ATM is needed in this scenario, 14 subjects miss it on the first level.

Scenario B: An Urgent Appointment

In scenario B, the satisfaction with the items is generally high with a median of 2 (rather satisfied) and 40% of the subjects are satisfied. One subject needs to enter the second level and one the third level. Both subjects are at least rather satisfied after selection on these levels. The detour is stated in 17 cases as relevant information. All other arguments, also the price, are stated mostly as irrelevant in case it is used in the explanation at all. The results reflect the adaptation of the user profile where the importance of detour is set to highest and for all other features to lowest. Asking the subjects if this behavior of the system (adaptation of user preferences) is desirable, all affirmed.

Scenario C: A Regular Trip

In the next scenario, the satisfaction with the selection is in general low with a median of 3 and 80% of the assessments equal to or worse than 3. 11 subjects enter the second level and their satisfaction slightly increases but still has a median of 3. The three subjects who enter level 3 are significantly more satisfied. The detour is in this scenario relevant for most of the users. In seven cases, the price is missed. Brand and gas level at the gas station play a minor role in this scenario. In most of the cases, three gas stations are recommended that are closely located to each other but differ in detour. Missing differentiation in the item set is stated by most of the subjects as the reason for entering further levels.

Scenario D: A Regular Trip (Without Explanations)

In Scenario D, no interpreted explanations are used but all four item features are listed as facts. The results show that the main features (price and detour) are classified as relevant information by more than 50% of the subjects. The gas level at the gas station is stated as irrelevant by more than 50%. Only six subjects believe that it is relevant. In two of the four cases where the brand is stated as relevant, the preferences for the brand are high. Hence, also two subjects who do not prefer a brand state this information as relevant. Furthermore, the brand is not irrelevant for the most of the subjects. Note that the brand is presented as a brand logo in this case. Further text explanations such as "brand gas station" or "favorite brand" are not used. Due to the delivery of all information in the first level, no subject enters the second level and only one enters the third level. The subjects are rather satisfied with their decisions in the first level with a median of 2. As all necessary information is delivered in the first level, the results for satisfaction rather reflect the satisfaction with the recommended items than with the explanations.

Scenario E: A Regular Trip with Inverted User Profile

In the next scenario, the profile is inverted with $U(D)^* = U_{max} - U(D)$. D is the item feature, $U(D)$ is the assessments of the users towards this feature and U_{max} is the highest possible preference for this feature. Note, the closer the preferences are to $\frac{U_{max}}{2}$, the less their weight is changed. Thus, mainly the subjects who state a dimension as very important or very unimportant should recognize a change. The statements towards relevance show that six subjects missed the price and eight subjects rated the brand explanations as irrelevant. This corresponds to the user preferences where price is often important and brand not important. In contrast to scenario D, the brand is explained directly with the text "brand gas station". Eight subjects enter level 2 of which five say that the explanations are irrelevant. Furthermore, five subjects also enter level 3 because of irrelevant recommendations. The median of the satisfaction is 2.5 in the first level. However, the subjects are indifferent because ca. 50% are rather satisfied or satisfied and 35% are rather unsatisfied or unsatisfied. Satisfaction increases to a median of 2 with more information in level 2 and further increases (with the same median) in level 3. When asked, 12 subjects say that they expected different explanations.

Scenario F: A Vacation Trip

In the last scenario, an irregular trip to a vacation destination is presented to the users. The satisfaction has a median of 3 but it varies strongly along the range of assessments. 12 subjects enter level 2 and justify their decisions in nine cases with missing information about resting opportunities (WC,Bistro) or no recommendation is recognized as relevant, mainly because of the location. If asked, 10 subjects say that they desire an appropriate option to make a break. For this reason, five subjects also enter level 3. The satisfaction between level 1 and 2 only slightly increases but with level 3 significantly. All subjects who switch to level 3 are at least rather satisfied. It is particularly noticeable that besides price that is missed six times and brand that is irrelevant six times, the gas level at the gas station is missed by six subjects and stated as relevant by three.

Further Results

Besides the investigation of specific scenarios, there are some more findings worth noticing. We ask the subjects in a follow-up questionnaire if they would prefer that the feature that is most important for them (e.g., price) should always be integrated in an explanation. 13 of 20 subjects affirm this question.

Based on the results in the preliminary study, we want to know if the number of items with a maximum of 3 and the number of arguments with 2 is confirmed in this experiment. The system recommends in most cases three items and sometimes two. 14 subjects confirm that the amount is neither too less nor too much. Eight subjects state the amount of arguments as rather too less or too less and nine subjects as enough. This is due to the settings of our method. Sometimes one and sometimes two arguments are calculated omitting negative arguments.

When interacting with the system, the subjects always have the possibility to interact with the street map or at least see the locations of the gas stations relative to the route. When asked, the subjects assess the map as a relevant source of information for their decision. For some of the scenarios (D,E,F) the map is more useful than for others (B). In B, the predominant argument is the detour. In general, the location of the stations in the map does not contribute to the decision much. However, for a vacation trip, the location is one of the most important factors. In their comments during the survey, the subjects seem to prefer to get gas outside the city area, probably because gas stations in dense city areas are more difficult to access.

Except of scenario D, we use interpreted explanations throughout the scenarios. In the follow-up questionnaire, we ask the subjects if they prefer to have clear facts instead. Although some subjects would strongly prefer interpreted arguments (3) and some rather prefer them (4), the majority of 11 subjects strongly or rather strongly prefer facts. During the study, the subjects often complain about the interpreted wordings. The most complains are directed to the fuzziness of interpreted wordings and a patronizing interpretation of information. For instance, not all users have the same understanding of an inexpensive gas station. The subjects who prefer interpreted wordings state the possibility of a faster decision with less cognitive effort as reason.

7.4.4. Discussion

Explanations can be classified as useless in case the users enter the second level and are not satisfied in the first level. This means that the items are not well explained and the users hope to find more relevant information in the next level. If the user is satisfied in the first level but enters the second level, then information that is not expected to be part of the explanation is missed. If the user is still not satisfied in the second level, then it indicates that the recommendations are not relevant. Entering level 3 because of dissatisfaction indicates that the user does not believe in more relevant information in the second level. Hence, recommendations are irrelevant. We see the negative effect of irrelevant recommendations in the scenarios E and F where satisfaction goes up after entering level 3. In scenario C, both explanations and recommendations are perceived as irrelevant.

Overall, our method provides mostly relevant explanations but sometimes information to differentiate is missed. The subjects perceive the number of arguments as exactly right or rather too less. A comparison between scenario B and C shows that although detour is stated as relevant information in both scenarios, in one scenario it is enough to make a satisfying decision in the other scenario it isn't. Especially the missing differentiation to other features leads to negative assessments. We conclude that explanations should be distinguishable. Incorporating counterarguments is one possibility. Also a dominance filter can be used (Pu and Chen [PC06]). It guarantees that arguments are not presented in a way that one dominates the other in the explanation.

The availability of user preferences does not mean that arguments can be selected only based on them. Looking at the preference order in the user profile indicates that the price is the most important feature, then the detour follows close behind. The gas level at the gas station is barely important. The brand is only important for some of the users. Regarding the assessment of relevant and irrelevant data (especially in scenario D) unveils that detour seems to have a higher influence on the decision because it is mostly relevant and seldom irrelevant. The Price follows and the gas level at the gas station does not have any influence. Brand only sometimes has an influence which is correctly reflected by the user profile. We conclude that the user profile is an important source for generating explanations but it is not the only one.

Taking the amount of information into account (information score) is a promising method to generate explanations. The assessments of the gas level at the gas station confirm the usage of the information score. The gas level at the gas station is irrelevant in most of the scenarios. In scenario D where it is shown every time, most of the users classify it as irrelevant additionally. The users do not want to get recommendations when their tank is still full or already empty. In between this range, the gas level at the gas station varies only slightly for short routes, i.e., it has a low information score. Although the users expect the system to incorporate this information in the calculation, it is not a feature for the final decision. This does not hold for scenario F. In this case, the gas level at the gas station is stated as relevant or is even missed. In this scenario, the gas level at the gas station varies much stronger because of a long route, i.e., it has a higher information score. Hence, the gas level at the gas station becomes relevant for the decision making.

Other features such as an ATM also have an influence on the decision of the user but do not lead to unsatisfied choices in scenario A. The subjects seem to understand and expect that the system is only hardly able to infer on the need for money. They accept that in this case accessing details on other levels is necessary. Hence, details should be made accessible very easily. But there are also situations in which the system is expected to handle additional needs. In scenario F, we have a long route in the vacations. This is comparable to the additional need of an ATM in scenario A. In this case, the subjects seem to expect that this circumstance is incorporated by the system. Hence, they are more unsatisfied. Especially the location of the gas station relative to the route and facilities such as a toilet play an important role.

Scenario B shows that we can confirm our finding in the preliminary study. The users like the intelligent incorporation of context information, e.g., appointments. It gives the system an unexpected intelligence in comparison to regular scenarios and it increases the benefit of such a system. We assume that there are more special situations that are suitable for such kind of intelligent filtering, e.g., the overall reachability of gas stations in not densely populated areas.

7.5. Situation Explanations

The second kind of explanations on which we focus in this thesis are situation explanations. These are explanations to justify the decision of the system to deliver recommendations based on the situations of a user. For instance, a low current gas level is an obvious situation for a gas station recommendation but there are some more situations that may lead to a recommendation. These situations can be a rather good gas station along the route (e.g., very low priced), a deserted area with few gas stations or an important appointment that leads to a recommendation only with gas stations on the route. Without an explanation, a proactive recommendation in these situations may result in confusion. In this section, we investigate how arguments can be extracted from the comprehension model described in Section 5.2. The goal of these models is to enable intelligent behavior of an advisor that decides what to do in a specific situation to assist the users in their tasks. A justification by means of explanations should reveal the reasons of the advisor for its decision. Reasons are situations that lead to the decision. We focus on the Bayesian network comprehension model in this section.

7.5.1. Explanation Goals

The goal of a situation explanation is to make the decision of the system comprehensible for the user. In contrast to our item explanation method, the reasoning instead of the resulting output should be explained. Hence, the explanation method depends on the applied method in the comprehension model. Our method explains the states of the high-level situation for the proactive recommendation ("Yes", "No", "Too late" and "Later") because the decision of the system is made upon these states. In general, we explain a binary decision of the system. Either it decides not to recommend in a situation or to deliver recommendations (notifying is regarded as delivery).

Our preliminary study shows that proactively delivered recommendations should rather contain less than more information. Therefore, our method extracts one or at most two significant situations that account for the decision of the system. The method delivers these arguments automatically. Additionally, our method extracts all situations that contribute to the decision, ranks them and makes them accessible for the user. We explain why recommendations are delivered by generating arguments for the decision of a system. This corresponds to a positive decision in case the system decides to deliver recommendations. A negative decision occurs if the system decides against delivery (see Section 4.3.3).

7.5.2. Explanation Task

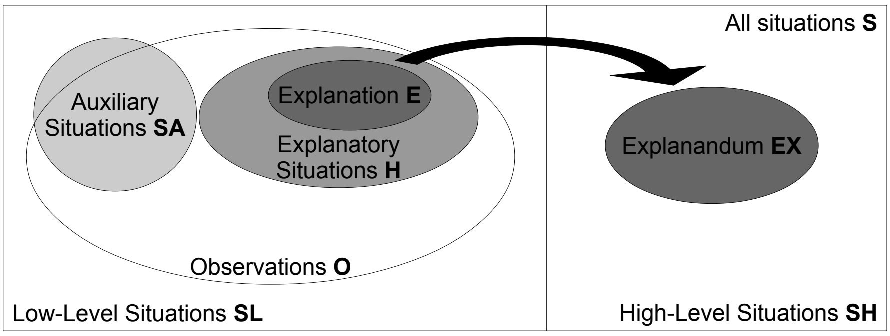


Figure 7.4.: Building an explanation E for those situations that should be explained (explanandum EX)

Our explanation task for the comprehension model is to select a set of situations for an explanation E out of the set of possible explanatory situations H (Figure 7.4). Explanatory situations H explain the situations in the explanandum EX. They correspond to criteria that the users incorporate in their decision towards items. Explanatory situations H are always observed situations O because of uncertainty of unobserved variables. Explaining with unobserved situations may confuse the user. For instance, if the believe of the system does not correspond to the experience of the user. Auxiliary situations SA are not taken as explanatory situations because decisions are not based on these situations. These situations can be observed and unobserved. The explanandum EX is a subset of the high-level situations SH that are not observable but inferred with comprehension models. Thus, we explain the reasoning on EX based on observations O. The explanation E is a set of situations and their states. This information is used as arguments. For instance, "current gas level is low" can be an argument with "current gas level" as situation and "low" as state.

7.5.3. Argument Selection

Argument Strength

Our idea is to measure the impact of an observation of a low-level situation on the posterior probability of a high-level situation. The impact is measured for each high-level situation SH separately. It describes the strength of an argument that is built on the low-level situation.

$$dy = P(SH|SL_1, \dots, SL_n, SA_1, \dots, SA_m) - P(SH) \tag{7.3}$$

For the impact, we define the increase difference dy in Equation 7.3. It is the difference between the posterior probability after observation of low-level situations SL and auxiliary situations SA and the prior probability. This corresponds to the change of the high-level situation after all observations.

$$dx_{i,k} = P(SH|SL_{i,k}, SA_1, \dots, SA_m) - P(SH) \tag{7.4}$$

To receive the impact of every single situation, we calculate the conditional probability of every state $SL_{i,k}$ and the high-level situation SH under all observations SA. The result is again compared with the prior probability of SH (Equation 7.4).

$$impact_{i,j,k} = |dx_{i,j,k}| \times (1.0 - |dx_{i,j,k} - dy_j|) \tag{7.5}$$

The final impact on each state SH_j is then defined as the product of the differences with Equation 7.5. The first term of the equation corresponds to the absolute difference that the k-th state of the low-level situation SL_i has on the j-th state of the high-level situation SH_j. As the change can either be negative or positive, we also need to incorporate the tendency of dy. The closer dx and dy are, the more impact the low-level situation has. For instance, "gas level = low" implies a high positive change to "Proactive Recommendation = Yes", i.e., the prior probability $P(''Recommendation = Yes'')$ is smaller than the posterior probability $P(''Recommendation = Yes''|''gaslevel = low'')$. If for any reasons, e.g., other influencing situations, we infer that the prior probability $P_{prior}(''Recommendation = Yes'')$ is higher than the posterior probability $P_{posterior}(''Recommendation = Yes'')$ and we make the observation "current gas level

= low", then the impact should be low because another situation causes the inference result.

$$score_{SL_i} = \sum_{j=1}^{h}(\sum_{k=1}^{l} \mu_{SL}(SL_{i,k}) \times impact_{i,j,k}) \times P(SH_j|SL_1, \ldots, SL_n, SA_1, \ldots, SA_m) \quad (7.6)$$

Additionally, we want to incorporate the membership μ_{SL} of the states of SL because they are fuzzy. We aggregate the impact over all states of SH to a final score in Equation 7.6. The score for each low-level situation SL_i is calculated over l states of the situation and its memberships μ_{SL}. It is weighted with the posterior probability of SH for h states. Weighting is important because we are more interested in the impact of more likely states of SH. These states contribute to the decision of the system. To determine the state k of the low-level situation SL_i that is taken as argument, we simply take the one with the highest membership in Equation 7.7.

$$SL_{i,k} = argmax_{\mu(SL_{i,k})} SL_i \quad (7.7)$$

Negative and Positive Arguments

For our final explanation, we have to distinguish positive and negative arguments. For instance, it makes no sense to present just a negative argument in case a recommendation is delivered. We determine a set of states of the high-level situation SH. The set corresponds to the action that the system performs. These are the two actions "recommend" or "don't recommend". We derive from the estimated benefit B that one is performed. The higher the benefit, the more likely a recommendation is given. Hence, rules that contribute to a positive benefit are positive and vice versa. The state "Yes" accounts for high benefit and the state "No" for low benefit. The state "Too late" is intermediate. Therefore, it can be disregarded. Finally, we simply have to compare $dx_{i,''Yes''}$ with $dx_{i,''No''}$ of situation SL_i for its state i with the highest membership (Equation 7.8).

$$positive(SL_i) = \begin{cases} true & dx_{i,''Yes''} > dx_{i,''No''} \\ false & \text{otherwise} \end{cases} \quad (7.8)$$

7.5.4. Explanation Process

Prefiltering

The situation states are processed in order of their importance for the user. The importance can be represented by the order of primary decision dimensions defined in the fuzzy logic comprehension model in Section 5.2. Situations that are more important in a decision are regarded earlier. With the importance and the impact, we select arguments for the explanation. If the observed situation has a larger impact than α, it is good enough to be included in the explanation. If its impact is even larger than β ($\beta > \alpha$), then we have a significant argument and can terminate the search for more arguments. Terminating can only be done in case there is already at least one argument that supports the decision. We also terminate the process if a maximum of n_{max} arguments are found. As the observations are processed in order of their importance, there can be a significant argument together with a more important argument in the explanation. In case the impact of no argument that supports the decision is larger than α, we search for the first argument in the observation set that supports the decision regardless of its impact. Only if one argument supports the decision, the search ends successfully. Otherwise, we suppose that something went wrong during the inference because it should be impossible that we recommend and no observation indicates that we should do so.

Finally, the observations are used to build an explanation. Different combinations are possible. For instance, they can be ordered by their impact or importance or the contribution can also be regarded. Supporting arguments can be given first, then the rest. The implementation of prefiltering of arguments is described in Appendix D.3.

Ranking

Depending on the value of α, explanations can become large. Especially in automobile and proactive scenarios, explanations should be brief and concise. Thus, we further reduce our explanation by considering more properties of an argument. To achieve this, we rank all the arguments and select the first k.

One further property of an observation is its obviousness. It expresses whether the user expects an argument. The more an argument is expected the less useful it is for argumentation. We measure the obviousness with the prior probability distribution. We assume that if the state of a situation is very likely, it is obvious for the user. However, taking only the probability of single states disregards the number of possible states a situation has. For instance, consider a situation with two and a situation with eight states. An observation o with $P(o) = 0.5$ is much more obvious in case of an eight state situation as the remaining probability $P(!o) = 1.0 - P(o) = 0.5$ is distributed over seven states in contrast to a one state situation. Hence, we need to regard the number of states in our assessment. We transfer the probability to a score with a function $f(x)$ where $f(0) = 0$, $f(1) = 1$ and $f(0.5) = 1 - \frac{1}{k}$ with k states. The function ensures that equally distributed observations are always mapped to an obviousness of 0.5. The memberships of the observations do not have to be considered for obviousness because only states with the maximum membership are presented to the user.

Another criterion that is taken into regard for ranking, is the prior knowledge of the user about the observation. Information that is already known by the user is less valuable than information that less known because it is determined or retrieved by the system itself. For instance, the current gas level is information that the most of the cars display clearly visible for the driver, whereas the estimated gas level at the destination is more valuable because it is calculated by the system. An argument based on information with strong prior knowledge can be more easily omitted in the explanation than an argument without prior knowledge. Automatic detection of prior knowledge is often unfeasible or involves high costs. On the other hand, experts can often easily estimate prior knowledge in many domains such as gas station recommendations. Therefore, we use a static mapping of situations and prior knowledge where we assign a value in $[0, 1]$ to each low-level situation in a profile.

Our final ranking task is to select a maximum of k observations out of a set of n prefiltered observations. Each observation is described by the importance IM, the impact IP, the obviousness OB and the prior knowledge PK. Hence, we have a multi-criteria decision making (MCDM) problem. We use an analytic hierarchy process (AHP) with pairwise comparison to rank the arguments. More details about our AHP can be found in Appendix D.3.

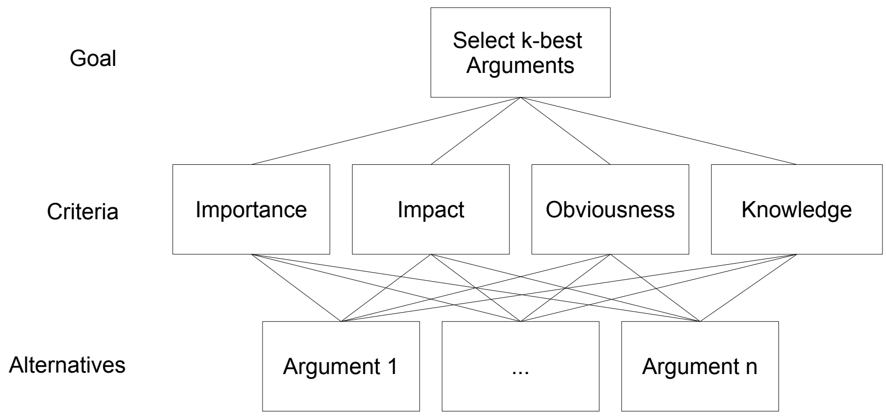


Figure 7.5.: Analytic hierarchy process (AHP) to select k-best arguments

Figure 7.5 shows our AHP hierarchy. Our goal is to rank the alternatives by a single numeric value and select k-best alternatives. On the criteria level, the weights of each criterion are determined by a manual pairwise comparison by experts. It resulted in a weight of 0.39 for importance, 0.39 for impact, 0.15 for obviousness and 0.07 for prior knowledge. On the alternative level of the hierarchy, every alternative is compared with each other on each criterion separately. As we do not want to do this manually, a mapping function from the difference of criteria $\triangle_c = A_i(c) - A_j(c)$ of alternative A_i and A_j to Saaty's Rating Scale is needed.

$$r_{i,j}(c) = 1 + 8f(\frac{A_i(c) - A_j(c)}{c_{range}}) \tag{7.9}$$

We use the function in Equation 7.9 to calculate the i, j-th rating value r in the pairwise comparison matrix for criterion c. The difference of the criteria is scaled with the range of the values c_{range} to make sure all values are in $[0, 1]$. The result is evaluated with a function $f(x)$ depending on the distribution of the values over the range of criterion c. Finally, we calculate the Eigen vector for each criterion but it does not need to be normalized. The ranking value is calculated with the linear function in Equation 7.10.

$$ranking(A_i) = \sum_{c \in C} Eigen(c) \times Eigen(A_i(c)) \tag{7.10}$$

Explanation Sets

In the final step, we create two explanations sets. A comprehensive explanation contains enough arguments m out of n prefiltered arguments to explain the situation in detail. However, in an automotive scenario we need a more condensed explanation as well. The second explanation called justification explanation contains k out of n arguments of the ranking where k is usually much smaller than m (one or two). The arguments in a justification explanation are also included in a comprehensive explanation. If no argument supports the decision in a justification explanation, e.g., the argument is negative in case a recommendation was given and vice versa, we simple take the next supporting argument in the rank. The prefiltering guarantees that there is at least one. With this information, we apply a surface generation described in the previous section for item explanations.

7.5.5. Discussion

Our argument strength is inspired by work of Suermondt [Sue92] and Jeffreys [Jef61] who measure the impact of a node in a Bayesian network to another node by the difference of posterior and prior probabilities. Our method also incorporates ideas of Shimony [Shi91] towards irrelevance of variables and their prior distributions by de Campos [dCGM01] to extract only relevant arguments. In contrast to Flores [FGM05] and Nielsen et al. [NPE08], we do not allow explanations with unobserved variables because we believe that this would confuse the user in our application domain, especially if the prediction is not correct. Additionally, our method incorporates the fuzzy nature of nodes in the Bayesian network. Compared to complex explanation domains such as medical diagnosis, our explanation process is much simpler because we do not have to follow the whole reasoning thread.

We choose AHP to rank arguments according to their properties. In general, argument selection can be done by determining optimal solutions, e.g., with a variant of the Skyline filter, or by building a ranking and selecting the top-k alternatives. The last method is more suitable with a constraint of a maximum of k elements. Many MCDM methods exist to build rankings. Multi-attribute utility theory (MAUT) requires assessments of alternatives in the same value range for each dimension. Otherwise, it would compare the incomparable. Although our criteria values are in $[0, 1]$, each of the criterion uses the range in a different way. Importance and impact usually take on small values, whereas obviousness and prior knowledge often take a value of 1. MAUT calculation is biased privileging the criteria with high values. Therefore, we use pairwise comparison which is not restricted to the range of the values. Pairwise comparison calculates assessments for every dimension by comparing each alternative on a dimension based on a uniform scale that makes dimensions comparable.

It could be useful to make comprehensive explanations available for the users who want to get a deeper understanding of the situation such as described by DeCarolis et al. [dCNPG09]. Their approach could be an extension of our comprehensive situation explanations. We simply list the situations that lead to a recommendation in order of their impact. DeCarolis et al. use comparative explanation structures and formulate whole sentences.

7.6. Evaluation of Situation Explanations

We run our method of selecting arguments for situation explanations in preselected situations to investigate the plausibility of generated explanations. We take $\beta = 0.15$ for significant arguments and $\alpha = 0.05$ for arguments that are good enough. The maximum number of arguments n_{max} is 5. For the justification explanation, we select $k = 2$ arguments. Table 7.3 shows the results. The first row of each example depicts the raw value of the context information that is used for the situations in the Bayesian comprehension model. The second row gives a possible comprehensive explanation and the third row a reduced justification explanation. The arguments are ordered by their impact IP. Arguments that support the decision are listed first. The decision threshold is $t = 0.55$, i.e., benefits B above the threshold lead to recommendations and values below the threshold are against recommending. The explanation text is created according to the state of the situation with the maximum membership. The table shows text examples that we created by hand because no surface generation is implemented. Decisions that support arguments in the beginning are separated by "but" or "however" from arguments against the decision. The examples do not claim to be sophisticated sentences because this is not the scope of the thesis.

7.6.1. Examples

The first example mainly explains the reason for a recommendation with "current gas level" and "gas level at the destination" because both are low. However, the reduced explanation contains "gas level at the destination" and "reachability of good located and low priced stations" because "current gas level" is a more obvious observation.

<table>
<tr><th>Sit.</th><th>B</th><th>GL</th><th>GAD</th><th>M</th><th>RL</th><th>RT</th><th>POR</th><th>A</th><th>REA</th><th>REL</th><th>REP</th></tr>
<tr><th>Unit</th><th>[0,1]</th><th>L</th><th>L</th><th>0—1</th><th>m</th><th>[0,7]</th><th>%</th><th>0—1</th><th>#</th><th>#</th><th>#</th></tr>
<tr><td>1</td><td>0.78</td><td>5.72</td><td>3.80</td><td>1</td><td>40</td><td>7</td><td>0.40</td><td>0</td><td>31</td><td>4</td><td>1</td></tr>
<tr><td></td><td colspan="11">Your gas level is low at destination [0.15] and now [0.13]. There are some well located and inexpensive gas stations [0.12], but only a few well located stations are reachable [0.05].</td></tr>
<tr><td></td><td colspan="11">Your gas level is low at destination and there are some inexpensive gas. stations.</td></tr>
<tr><td>2</td><td>0.58</td><td>8.92</td><td>7.40</td><td>1</td><td>20</td><td>7</td><td>0.05</td><td>0</td><td>38</td><td>2</td><td>1</td></tr>
<tr><td></td><td colspan="11">There are some inexpensive and well located stations [0.15], only few well located gas stations are reachable [0.09] and you are in the beginning of the trip [0.07]. However it's a regular route [0.06] and many gas stations can be reached [0.06].</td></tr>
<tr><td></td><td colspan="11">There are some inexpensive and well located gas stations.</td></tr>
<tr><td>3</td><td>0.38</td><td>8.84</td><td>7.40</td><td>1</td><td>20</td><td>7</td><td>0.10</td><td>0</td><td>37</td><td>2</td><td>0</td></tr>
<tr><td></td><td colspan="11">There are many stations reachable [0.08], you have enough gas at destination [0.07] and you are on a regular route [0.07]. However, there are only few gas stations well located [0.06] and you are in the beginning of the trip [0.05].</td></tr>
<tr><td></td><td colspan="11">There is enough gas at the destination, but there are only few well located stations.</td></tr>
<tr><td>4</td><td>0.27</td><td>14.04</td><td>12.60</td><td>1</td><td>30</td><td>7</td><td>0.40</td><td>0</td><td>31</td><td>1</td><td>0</td></tr>
<tr><td></td><td colspan="11">There is a lot of gas at the destination [0.27].</td></tr>
<tr><td></td><td colspan="11">There is a lot of gas at the destination.</td></tr>
<tr><td>5</td><td>0.45</td><td>13.56</td><td>8.60</td><td>1</td><td>80</td><td>7</td><td>0.23</td><td>0</td><td>34</td><td>2</td><td>0</td></tr>
<tr><td></td><td colspan="11">You have much gas now [0.18] and enough at destination [0.09].</td></tr>
<tr><td></td><td colspan="11">You have much gas now and enough at destination.</td></tr>
<tr><td>6</td><td>0.80</td><td>3.12</td><td>-2.40</td><td>0</td><td>80</td><td>7</td><td>0.14</td><td>0</td><td>36</td><td>2</td><td>0</td></tr>
<tr><td></td><td colspan="11">You are out of gas at the destination [0.22].</td></tr>
<tr><td></td><td colspan="11">You are out of gas at the destination.</td></tr>
<tr><td>7</td><td>0.89</td><td>3.92</td><td>3.60</td><td>1</td><td>5</td><td>7</td><td>0.20</td><td>0</td><td>35</td><td>2</td><td>0</td></tr>
<tr><td></td><td colspan="11">Your gas level is low at the destination [0.15] and now [0.15], there are only few well located gas stations [0.06].</td></tr>
<tr><td></td><td colspan="11">Your gas level is low at destination and currently.</td></tr>
<tr><td>8</td><td>0.36</td><td>7.08</td><td>3.80</td><td>1</td><td>65</td><td>2</td><td>0.37</td><td>1</td><td>31</td><td>5</td><td>0</td></tr>
<tr><td></td><td colspan="11">You are on a business trip [0.15], you can reach many well located stations [0.10], you have enough gas [0.10] and you are also on a regular route [0.09]. However, you are low on gas at the destination [0.15].</td></tr>
<tr><td></td><td colspan="11">You are on a business trip, but you are low on gas at destination.</td></tr>
</table>

Table 7.3.: Generated example explanations with task benefit (B) and situations: current gas level (GL), gas at the destination (GLAD), modality (M), route length (RL), route type (RT), position on route (POR), appointment (A), reachability of gas stations (REA), well located gas stations (REL) and well located and inexpensive gas stations (REP); [argument strength]

In the second example, the system decided to recommend. The explanation shows that the reasons do not include primary decision dimensions. Both "current gas level" and "gas level at the destination" are in state "enough" which generally leads to no recommendation. However, in this situation there are some well located and inexpensive gas stations along the route.

In Example 3, no recommendation is given. Many reasons such as enough reachable gas stations or enough gas at the destination confirm this decision. Just a few well located gas stations that are reachable do not lead to a recommendation in this case. Together with "gas at the destination" it forms the reduced explanation because the knowledge of many reachable gas stations is much more obvious than their location relative to the route.

In the examples 4, 5 and 6, primary decision dimensions clearly dominate the reasoning. No recommendation is given in Example 4 because of much gas at the destination (the impact exceeds α). Example 5 shows that "gas at the destination = enough" does not explain solely that no recommendation is given because there might be other reasons. On the other hand, the situation "current gas level = much" explains the decision (it also exceeds α). Note, although "current gas level" exceeds α, "gas level at the destination" is also included in the explanation because it is more important in the decision for refilling. In Example 6, an empty gas tank at the destination exceeds α but this time in favor of a decision for recommending.

In the seventh example, the decision to recommend is mainly based on "current gas level" and "gas at the destination". The decision is underlined by only a few well located gas stations that are reachable.

In the last example, a low gas level at the destination would usually account for delivering a recommendation. However, in case of a business trip, the system makes the opposite decision. This is the case because there is enough gas in the tank.

7.6.2. Discussion

The examples show that our method generates reasonable justification explanations which should be easy to understand for the driver. The explanations are dominated by primary decision dimensions such as current gas level and gas level at the destination in most cases. However, other situations such as well located and inexpensive gas stations or a business trip with an appointment are also sometimes the main argument. The results for the longer comprehension explanations are often complicated sentences. Simply listing the arguments by their rank and contribution to the decision often does not generate clear explanations. Guidelines from the argumentation theory concerning the position of an argument may help to improve the results. For justification explanations, most guidelines are not necessary because the explanation is too short. Based on the examples that we discussed, it is not clear if comprehension explanations provide more useful information than justification explanations.

Our method strongly depends on the quality of data for parameter estimation in the Bayesian network. The values for the strength of an argument are low for all situations, even if the situation is clear, e.g., the current gas level and the gas level at the destination. Noise in data may have an influence on the ranking of arguments. As a result, arguments may change frequently along a route.

Obviousness and prior knowledge are relevant criteria to generate useful arguments. However, their value is static and only an estimate. Different users of the system may perceive these criteria differently, especially with experience with the system.

7.7. Summary

To justify system decisions, we presented methods to generate explanations for recommended items and for the reason why a recommendation was given. The evaluation is carried out for a gas station recommender. For this use case, we do not know which kind of information should be used to generate arguments for the explanation. A preliminary study investigated arguments, counterarguments, arguments for items that were not recommended, number of arguments and the structure of the arguments for item explanations. The results show that only a few arguments (one or two) should be used for each item and the arguments should be selected according to the user preferences. The study also indicates that the users like explanations with situations that are a reason for a recommendation.

Several different kinds of explanations are imaginable in a P-IVRS. They follow different explanations goals, impose different amounts of cognitive load and may require involvement of the user to be retrieved. We classified relevant explanations for a P-IVRS towards the user interface. The approach corresponds to several stages of a recommendation process in a mixed initiative with the driver. Our focus is explanations justifying system decisions, i.e., why something was recommended and why exactly these items.

We presented a score-based method for item explanations that uses the output of our context-aware recommender. The method adds two scores for strength of an argument and its amount of information. The strength is based on the performance of the item features and user preferences. Amount of information comprises how obvious an argument might be. Our explanation process selects suitable arguments (content selection) and builds the final explanation from its selection (surface generation). First, a major argument is selected. A second argument is necessary for item explanations in case not enough information is in the first argument. In a second user study, we implemented parts of the explanation user interface and let the participants assess generated explanations in various scenarios. In most cases, our method generated relevant explanations. However, only one argument was often not enough for an item. Although user preferences have a major influence on the relevance of arguments, also further aspects seem to be important. In general, the quality of the explanations depends on the items that are recommended by the system.

Our explanation method for situations focuses on our Bayesian network comprehension model. In contrast to the item explanation method, the situation explanation method is tailored to the method of inference. In case of items, the scores contain all necessary information for an explanation. There, we do not need to explain the method to generate scores. In contrast to that, the posterior setting of the comprehension model defines the reasons for inference, i.e., which probabilities are propagated. The goal of the situation explanation process is to select observed situations that have an impact (cause change of probability) on the output of the comprehension model. This corresponds to the argument strength. The process also involves further dimensions such as obviousness of information and prior knowledge about the information. It ranks potential situations for an explanation. Finally, a justification explanation is built with one or two arguments and a comprehension explanation with more arguments. Examples for justification explanations show reasonable results. The usefulness of comprehension explanations should be regarded in a further analysis.

8. User Acceptance of a P-IVRS

The previous chapters dealt with functional aspects of P-IVRS. We described how situation-aware behavior is enabled, context-aware recommendations are selected and explanations are generated. The evaluation of the presented methods was carried out offline or on a desktop system. However, a car offers a restrictive environment for recommender systems where interaction and information delivery is limited due to information overload and driver distraction. Hence, we cannot tell solely based on its functionality whether a P-IVRS would be accepted while driving. In this chapter, we implement an integrated P-IVRS for gas station recommendations. Our main focus of investigation is the usefulness of the system. As the prototype is in an early stage of development, ease of use plays also an important role. Therefore, we first design the user interface of the P-IVRS according to requirements on proactive recommender system. To adjust the recommender to the requirements of an automotive environment, we ask automotive experts in interviews with our prototype. With the final user interface, we carry out a user study using the method of "think aloud". We published our expert study, the user interface design and an abstract of the evaluation of usefulness in [BSW11].

8.1. Ease of Use

The technology acceptance model (TAM) that was introduced in Section 2.5.5 is used as theoretical framework for our investigation. Our focus in this chapter is to derive perceived usefulness (U) of a proactive recommender system inside a car. In TAM, perceived ease of use (EOU) is the other important factor influencing user acceptance and usefulness (U). As our prototype is in an early stage, EOU is a crucial aspect when the users interact with the prototype. Therefore, we design an in-vehicle interface for proactive recommendations in this section.

8.1.1. Design Guidelines

We carry out expert interviews with BMW engineers to incorporate their experience in our interface design. We show the experts an early version of our prototype and ask them about their opinion. The experts are selected based on their expertise. One is a specialist for HMI aspects in cars, the other for POIs in the navigation system of the car, the next for driver assistance systems and the last for customer marketing. Some example statements of the experts are listed in Table 8.1. They are grouped according

Guideline	Description	Examples of expert statements
Minimal Information	Too much information as well as complex information should be avoided	The display of a detour route provides no information while driving
Scanability	The information should be comprehensible by as few glances as possible	Long-term action sequences on the map, e.g., animated zooming, should be avoided
Consistency	The effect of actions, icons and colors should be consistent during the interaction	A push to the left should lead to more details; Unfamiliar information presentation, e.g., gas level in liter, and icons known in other context should be avoided
Expectation	The system should be controllable and structured the way a driver expect it to be or is familiar with	Pressing the controller is always coupled with an action
Metaphors	Information delivery and interaction should be mapped to known metaphors	A checkered flag should indicate that the item can be integrated in the current route
Controllability	The driver should have full control over the system at all times	Every action should deliver feedback, e.g., a confirmation for a selected item
Input	As less input from the driver as possible should be required while driving	The pop-up should easily be escapable
Understandability	Text and graphics should be understandable by laymen	Usage of colors for assessment should be transparent; Icons should be easy to understand or familiar; The current position should always be displayed

Table 8.1.: Design guidelines derived from expert interviews

to common guidelines for in-vehicle information systems (IVIS) that are described by Green [GLPS95]. Green's work comprises visual, speech, haptic and acoustic in- and output guidelines for in-car information displays.

We also incorporate assessment criteria for driver assistance systems summarized by Breuer [Bre09] . First, the adaptation phase should be as short as possible. Second, the system should react like the driver expects it does and it should be consistent in similar situations. Third, information systems should have a clear human machine interface (HMI) and should be easy to control, e.g., turning off and on. These guidelines are based on conditions that the automotive environment imposes (see Section 2.1). In general, the less cognitive effort and distraction an information system causes, the better.

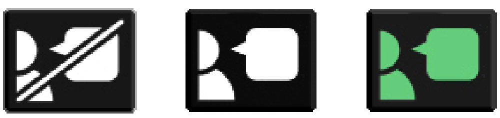
To achieve the collected design goals without comprehensive user studies, we design our interface as closely as possible according to the current HMI concept of current BMW cars in 2011.

8.1.2. User Interface Design

Our recommender is integrated in the central information display (CID). The driver is able to control the recommender with the iDrive controller. Other interfaces such as the instrument cluster or the head-up display are out of scope of this thesis. The same applies to other channels of output such as speech or haptic. The iDrive controller can be pressed, turned and pushed to the right, left, top or bottom. The recommender is mainly designed according to the three main requirements on P-IVRS concerning ease of use (unobtrusiveness, accessibility, transparency). It takes requirements on IVIS into account which are derived from literature and our expert interviews. The stages of interaction correspond to our theoretical concept of an explanation user interface for a P-IVRS described in Section 7.2.

Unobtrusiveness

If a recommendation system has information for the drivers, they should notice this while driving. On the other hand, distraction from the primary task of driving with information overload should be avoided. For this reason, we use a two-phase proactivity approach. A small icon in the right lower corner of the screen represents the recommender (Figure 8.1). It can either be inactive (Figure 8.1a) or active without (Figure 8.1b) or with recommendations (Figure 8.1c).



(a) Inactive (b) Idle (c) Notify

Figure 8.1.: Recommender icon with different states

In the first phase, the driver is notified about available recommendations by a change of color (Figure 8.1c). This gives the driver the option to look up the recommendations in advance without time pressure. The driver is able to use a less workload intensive situation such as a traffic jam or standing still at a traffic light to review available recommendations by pressing the recommender icon. The second phase covers the pop-up view (Figure 8.2). It is either shown automatically by the system itself or is invoked by the driver manually by pressing the recommender icon.

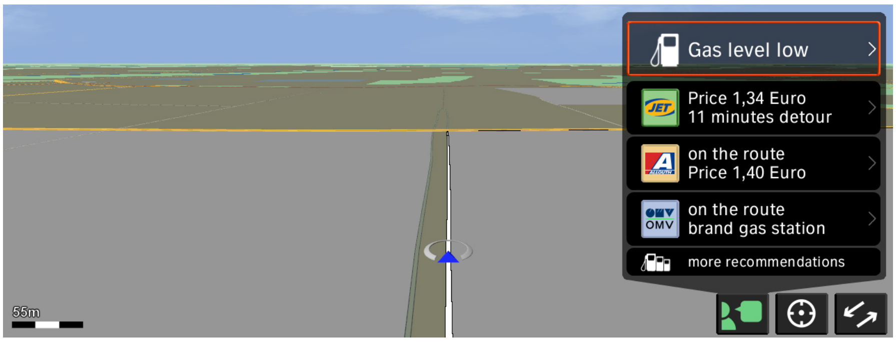


Figure 8.2.: Navigation mode: Pop-up view of the proactive recommender next to the navigation map

The pop-up captures one third of the map view to leave enough space in the viewing area to navigate or to orientate on the map. It is placed on the right side to cover the area that is farther away from the driver. This is less distracting as on the left side which is closer to the driver. After the pop-up appears, the map stays in the same mode as it was before (navigation mode). No interaction is required from the driver at this moment. After a certain amount of time (e.g., 30 seconds), the pop-up starts to fade out and finally disappears. We assume that the driver is not interested in recommendations in this case. Any kind of interaction, except the closing action, changes the map to exploration mode where the location of items and the current position is shown.

Accessibility

Proactive recommendations appear without request of a user. For efficient decision making, relevant information should easily be accessible. Therefore, interaction patterns from the BMW HMI are used. The drivers are already used to these patterns. The pop-up in Figure 8.2 is designed as list which matches the turn movement of the iDrive controller. Also, pushing and pressing of the controller behaves as the drivers would expect. A push to the left leads to less details or to close the recommender on the lowest detail level. A push to the right reveals more details if available. A small arrow depicts allowed directions of pushing for novice users (e.g., in Figure 8.2). The users are able to escape the pop-up whenever they want with an extra button. Pressing the controller on a list entry leads to a change of view. For instance, pressing the list entry of an item adds the item to the route and automatically plans a route via the item. Important details of an item and a list of more recommendations can easily be accessed. Figure 8.3 shows the detail view for items. It contains item attributes (e.g., facilities of the gas station) and features with explanations (e.g., a detour of 11 min is a "long detour").

Figure 8.3.: Item detail view with features and explanations for the features

The drivers may also dislike the items because they are interested in recommendations for another task. Other tasks can be accessed by a left push on the header of the pop-up view. Figure 8.4 shows a list of tasks for which a driver is able to receive recommendations. The list may be ordered by the benefit of a recommendation for a task. For instance, the task "refueling" can be on top of the list in case the gas level is low and the task "parking" is on top in case the drivers are close to their destinations.

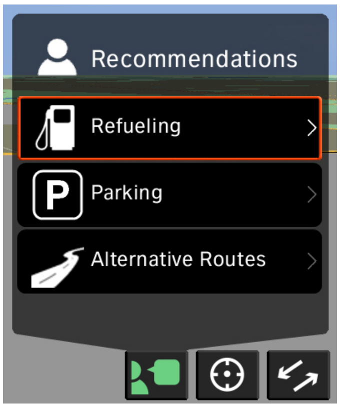


Figure 8.4.: Task view that shows all possible tasks

Transparency

IVIS should be understandable for the users in general. This also applies to proactive recommendations as information is delivered without the request of a user. The decisions of the system are justified by means of explanations for the reason of a recommendation and the strength of each recommended item. We use our method from Chapter 7 to extract such explanations automatically. On the first level of the pop-up, the most influencing reason of a recommendation is explained in the header of the pop-up view (the justification explanation). If more detailed information about the situation is needed, it can be accessed with a push to the right on the header. This results in a comprehensive explanation (Figure 8.5) that lists the arguments of the situation explanation. The order is chosen according to the impact that a situation has to the recommendation decision.

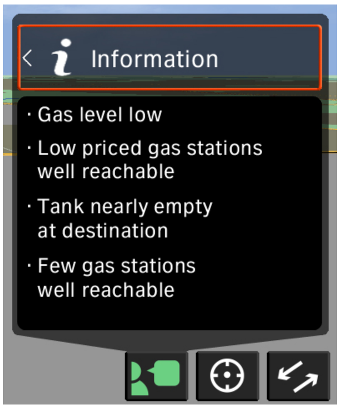


Figure 8.5.: Situation explanation view with a detailed explanation

Recommended items are explained on the first level of the interface with facts, e.g., "Price 1,34 Euro", or interpreted information, e.g., "extra low priced". On the second level, further details can be accessed with a right push (Figure 8.3). We only use minimal sentences like "current gas level low" in the surface generation and avoid additional explanations like "These items are recommended to you because ..." to reduce information overload while driving. An icon in the header indicates the type of a recommendation.

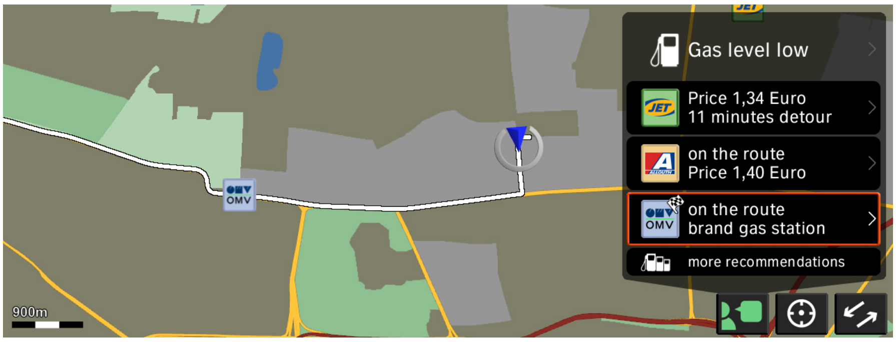
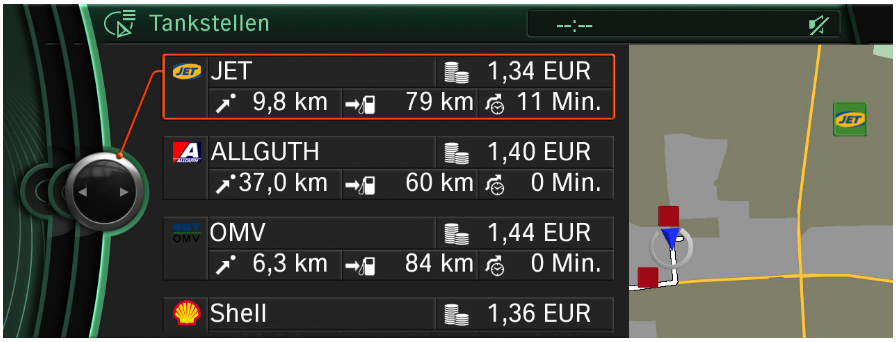


Figure 8.6.: Exploration mode: map view with the item location and the current position

Because the offline study in the last chapter showed that the map view provides a rich information source for quick orientation, the items in the list of the pop-up can be matched to the POIs on the map by the brand of the item and a unique color. For a better orientation in the exploration mode, the zoom level of the map is adjusted automatically if an item is selected. The new zoom level shows the current position of the car and the location of the item (Figure 8.6). This makes it easier for the users to estimate the location of the item relative to their routes. Every further item is displayed as a colored icon on the map like in Figure 8.7a. If further information is needed, a list of all items (Figure 8.7b) can be accessed as a third level of the interface by pressing the controller after the last item in the list (”more recommendations” in Figure 8.6).

(a) More recommendations as small icons on the map



(b) List view with item features: price for gas (e.g., 1.34 EUR), distance to the station (e.g., 9.8 km), remaining distance to drive when arriving at the station (e.g., 79 km) and detour (e.g., 11 min)

Figure 8.7.: More recommendations shown on the map (a) and as list view (b)

8.2. Usefulness

The previous section addressed ease of use (EOU) of the prototype based on expert interviews and requirements from literature. We carry out a second study with potential users to investigate perceived usefulness (U) in more detail.

8.2.1. User Study

We set up a prototype-based approach for evaluation instead of a scenario-based approach with paper prototypes. Van der Heijden et al. [vdHKK05] suggest using the prototype-based approach for mobile recommender systems. To limit costs and time, we combine the prototype-based approach with predefined scenarios. Our subjects use the system in these scenarios. The prototype is deployed in a BMW 550i GT. The CID of the car is connected with our prototype which is installed on a regular PC that is located in the trunk. The drivers control the prototype with the iDrive Controller. A common street map is used with about 15.000 gas stations. We update the gas prices in advance of the study by means of an online service. All routes are calculated by the route planner which we already implemented for the study in the last chapter. As we neither use artificial data nor mock-up functionality, the setup is close to a real world system.

Metrics for Perceived Usefulness

We determine four external variables to measure perceived usefulness (U) based on findings in literature. The expectations of the users are an important criterion because the prototype is an early stage where the participants of the study do not know the functionality in advance. The more a system meets the expectations of the users, the more valuable it is perceived (derived from expectation confirmation theory by Oliver [Oli80]). In case of proactive recommendations, transparency also influences U. The better the system explains its reasoning, the stronger relevance and utility are perceived. It was already shown by Cramer et al. [CER$^+$08] that transparency has a positive effect on how the users perceive the competence of a system. Relevance of items is in general an important factor for recommender systems. Additionally to the relevance of information, Rhodes [Rho00] suggests measuring the utility in case of proactive information systems. Utility comprises context-awareness as well as accuracy.

Predefined Scenarios

In the first scenario the subjects are asked to imagine that they picked up their new BMW recently and now they want to explore the functionality of a new feature, the proactive recommender for gas stations. We follow an explorative strategy where the subjects become familiar with the system by themselves. If features are not discovered or icons and words are not comprehensible, the interviewer explains everything that is not clear.

For the second scenario, we choose a route from the north of Munich to Herrsching with a current gas level of six out of 80 liters where refilling the tank is necessary. The subjects should imagine that they are on a trip in their leisure time. In most of the cases, the recommender pops up right after the beginning of the ride because there are many gas stations in this area and the current gas level is low.

In the third scenario, the current gas level is set to 13 liters where refilling is not necessary but the drivers may perform it on occasion. The subjects should imagine that they drive home from the city after a shopping tour. It is a half an hour ride from the north of Munich to Dachau. On the way to Dachau, there is an inexpensive gas station along the route and all other available gas stations are much more expensive. Depending on the settings for price in the user profile, some subjects are notified that there is a recommendation. Only this item is recommended.

In the fourth scenario, the current gas level is set back to six liter and time pressure due to an appointment is added. User preferences are adapted proactively. For instance, if price is preferred, it becomes less important in the assessment.

Study Design

Each subject is interviewed individually within our test vehicle in a standing still situation and while driving. After a socio-demographic questionnaire, we interview the subjects about their expectations of a proactive recommender in general and especially of a gas station recommender. They also have to enter their preferences for gas price, detour, brand and remaining gas level at the gas station. Then, we successively give them descriptions of the predefined scenarios. After they finished reading a description, the users are allowed to drive off. The method of "think aloud" is used while selecting a gas station to analyze the decision making process. For each of the scenarios, the current gas level of the car is adjusted accordingly. In the first scenario, the car is in a parking position to make the subjects familiar with the prototype. In all other scenarios, the subjects drive the car and follow guidance instructions to a specific destination. To shorten each run, the scenarios are interrupted after item selection.

ID	**Statements**
Q1a	The recommendation fits my given profile
Q1b	The recommendation fits my current situation
Q2	The number of items is enough
Q3	I can make a decision for a gas station based on the given information
Q4	I understand why I got these items
Q5	I am satisfied with the recommendation

Table 8.2.: statements for measuring usefulness (U)

To measure the usefulness of the recommendations, the subjects have to assess five statements (Table 8.2) on a 5-point Likert scale after the pop-up view appears and before they are allowed to interact with the system. The scale ranges from "strong disagreement" (-2) to "strong agreement" (2). The preferences in the second scenario do not depend on the situation. Therefore, we use Q1a here. The study is completed by a final survey. The subjects assess statements (Table 8.3) about how easy it was to use the system and if it is useful in general. Additionally, a semantic differential is established.

ID	Statements	TAM
S1	It is simple to use this system	EOU
S2	The information is presented clearly	U
S3	It is difficult to learn to use the system	EOU
S4	The system helps me to find a suitable gas station	U
S5	The system does not bother me while driving	EOU
S6	The system has all functions and capabilities I expected	U
S7	Overall, I am not satisfied with this system	U
S8	I trust the information the system presented	T
S9	I do not feel restricted by the system in my agency	EOU
S10	The recommendations match my user profile	U
S11	I would use this system in my car	A

Table 8.3.: statements for ease of use (EOU) and overall usefulness (U) and attitude to use (A) and trust (T)

Finally, we ask the subjects to assess the combinations of wordings for item explanations with paper prototypes of the variants. We use two kinds of wordings in the study. Fuzzy wording maps a subjective assessment of crisp values to vague statements such as "low price" and crisp wording shows the facts.

After the first five subjects it was clear that the system provides too few recommendations and all subjects prefer crisp wording instead of fuzzy wording. Therefore, we slightly adjusted our settings after the fifth subject to address this observation.

The study involves 15 subjects, five females and 10 males. Five are between 20 and 39 years old, six between 40 and 49 and four over 50. All of them own a car and 12 even a BMW, hence they are familiar with BMW interaction logic. Six of the subjects have much driving experience (25.000 to 50.000 kilometer a year) and 9 have average experience (5.000 to 15.000 kilometer a year). Only one subject has no experience with navigational systems. The others use it daily (6), several times a week (4), once a week (2) or less than once a week (2). They usually use it for guidance to a destination or for orientation on the road map. Five of the subjects already used POI selection and guidance to a selected POI.

Overview of the Results

The resulting average assessments of the subjects are shown in Figures 8.8, 8.9 and 8.10. We discuss them in detail in the following. A detailed view of user assessments can be found in Appendix E.

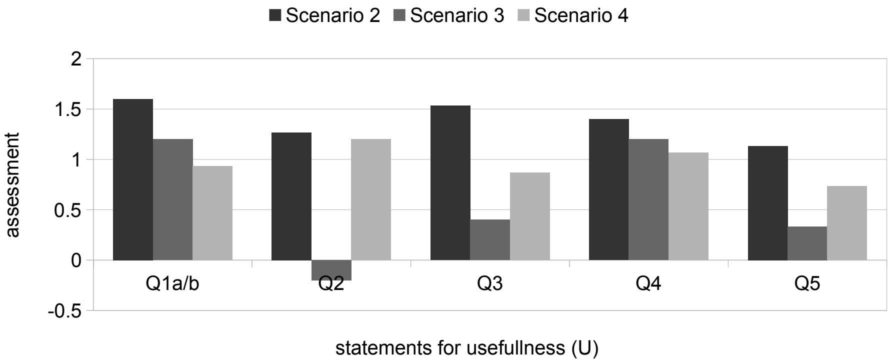


Figure 8.8.: Assessments of scenarios 2 to 4 for the statements in Table 8.2

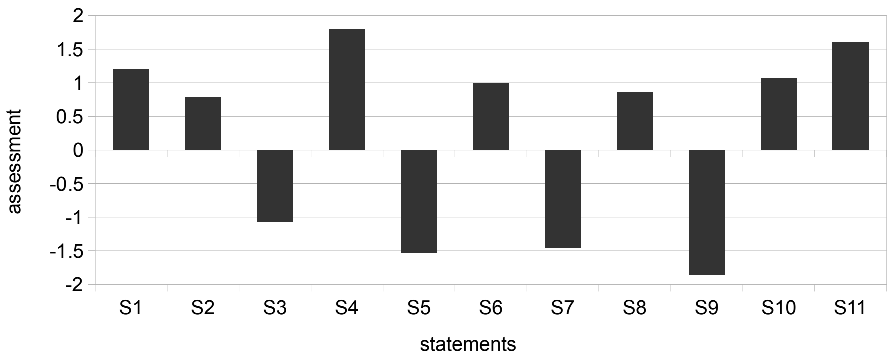


Figure 8.9.: Assessments of statements in the follow-up survey in Table 8.3

8.2.2. Results for Usefulness

Figure 8.9 shows that subjects perceive the system as useful in general because their assessment of satisfaction is between agreement and strong agreement (S7). In this section, we investigate how satisfaction is distributed over our determined external variables for U. These attributes are the relevance of items, the utility, the transparency and the expectations of the users.

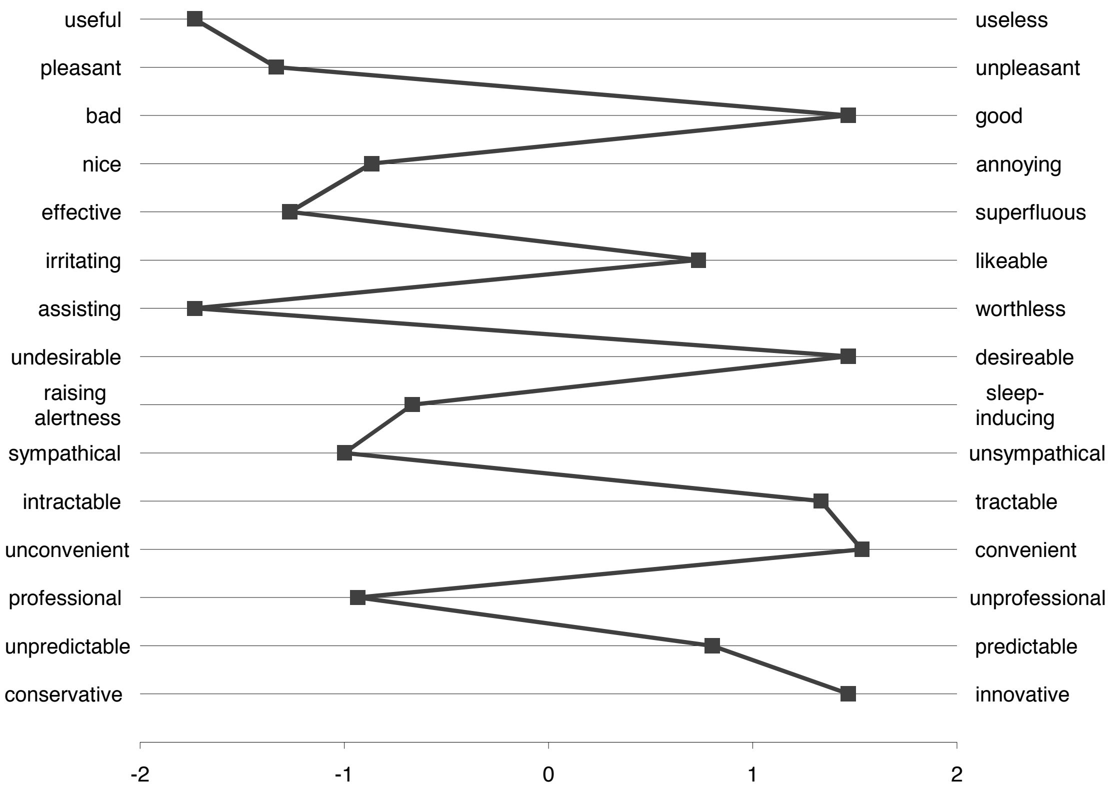


Figure 8.10.: Semantic differential

Expectations

The initial question in our study is about the expectations of the users for a proactive in-vehicle recommender system. The most frequent expectations are the quality of data (completeness, correctness, up-to-dateness) (7 out of 15 subjects) and easy interaction (5 out of 15). More than three subjects also expect personalization, situation awareness (e.g., the pressure of the tires) and clarity of information. In the second question, we interview the subjects about their expectations for a gas station recommender. The subjects mention the incorporation of several data sources as the most expected property. This includes static (location, brand, facilities, type of gas) as well as dynamic sources (gas price, detour, opening times). Price and detour are mentioned by more than two thirds of the subjects. A proactive gas station recommender is also expected to be sensitive to situations (primarily current gas level and current route) and to make intelligent calculations, e.g., an economic optimum of gas price and detour. The subjects agree that the system altogether meets their expectations (S6 in Figure 8.9).

Transparency

Transparency comprises the quality of item and situation explanations and their comprehensibility based on the wording.

At the end of the study, we showed the subjects paper prototypes of the pop-up view (Figure 8.2) with four different variants of wordings (fuzzy price and detour (A), fuzzy price and crisp detour (B), crisp price and fuzzy detour (C), crisp price and detour (D)). 10 subjects prefer D and eight subjects say that A is the worst. Less amount of information and ambiguous understanding of fuzzy interpretations are the most serious complaints about fuzzy wording. Most subjects mention that just in case of the same amount of information, e.g., "on the route", they prefer fuzzy wording. We used A in the beginning of the study and switched to D after interviewing five subjects because the tendency of preferences became quickly clear.

The quality of explanations is investigated with statements Q4 and Q3 (Figure 8.8) in each scenario. The subjects agree that they understand why items are recommended to them (Q4). On the other hand, they less agree with the overall clarity of information delivery (S2 in Figure 8.9). The subjects mention that they prefer a fixed order of arguments to be able to scan the attributes of an item quickly. Our method orders the arguments dynamically. Overall, the transparency is the highest in scenario 2. The subjects strongly agree in that scenario that they can make a decision with provided information (Q3) and agree less to that statement in the more advanced scenario 4.

Although the subjects are neither satisfied nor dissatisfied with the recommendation (Q5) in scenario 3, they understand why these items are recommended (Q4). Hence, it is transparent for the subjects why they received a recommendation. Note that the reason of the recommendation delivery is chosen to be obvious in all scenarios except the third scenario. There, all subjects see the explanation "Inexpensive gas stations along the route". The assessment of Q1b (Figure 8.8) for that scenario shows that this short explanation helps the subjects to understand the reason, although they are not satisfied with the recommendation itself (Q5). However, the subjects never used the detail view (Figure 8.5) for comprehension explanations. The subjects preferred to ask the interviewer for details about the reason or they are not interested in further details for the reason.

Relevance

Relevance is measured by regarding subjective assessments towards the perceived quality of items and by analyzing selection behavior of the subjects. First, we regard the position of items that are selected in case of more than one item in a recommendation. In 52% of the cases, the first item is selected. The second item is selected in 35% and the third in 13%. Most of the subjects associate a ranking with the list of recommendations. In this ranking, the first item is perceived as the best item. In 95% of the cases, an item out of the recommendation set is selected. Overall, the system is considered as effective in the semantic differential (Figure 8.10) and the subjects agree that recommendations match their user profile (S10 in Figure 8.9) in general.

Utility

Besides relevance, we also investigate whether the recommendations are useful in the situations. In the semantic differential (Figure 8.10), the subjects strongly agree that the system is useful and assisting. The system also helps them to find a suitable gas station (S4 in Figure 8.9).

In the second scenario, the subjects assess the number of items as too few because in three cases only one item was recommended before we changed the settings after the fifth subject. The assessment changes after at least two items are recommended in the following runs. Independent from the number of items, the subjects state that the recommendations match their user profile (Q1a). Seven out of 15 subjects use the detail view and six even use the list view. 11 subjects select a gas station based on provided information, two subjects confirm their selection in the list view and two subjects cannot make a decision at all. All in all, the subjects are more satisfied with the results (Q5). Those who are less satisfied, interact most with the system.

All subjects are noticeably confused that only one item is recommended in the third scenario. The assessment of Q2 reflects that more items are expected. The subjects also expect more items because there have been more items in the second scenario. 12 subjects use the list view to check if there are other recommendations. In the list view, most of the subjects realize that only one item makes sense to be recommended in this scenario. As reason for their confusion, they mention missing trust in the system in this early phase of usage. Only five subjects can make a selection without more information, five subjects confirm the recommended item in the list view and five subjects do not select a gas station under these circumstances. The subjects are neither satisfied nor dissatisfied with the recommendation (Q5) and can neither decide nor not decide for an item (Q3). However, the recommendation rather matches their user profile in that situation (Q1b).

In scenario 4, the first five subjects only receive one item before we changed the settings. All other subjects receive more recommendations. The effect can be seen in their assessment. The first five subjects are neither satisfied nor dissatisfied with the recommendation and with the number of items. Four of them use the list view and just two select an item. The other 10 subjects behave differently. Just one of them uses the list view to compare the selection. All 10 subjects select an item out of the recommendations. They state that the items rather match their user profile in that situation (Q1b). They are rather satisfied (Q5) with the recommendation and they can rather make a decision for an item (Q3). None of the subjects complained about the change of their user profile due to the situation. Most of the subjects expect this to happen.

8.2.3. Other Results

Choice

In the study, we observed that not giving the driver a choice by recommending only one item confuses them. When asked about the reason, most of them mention missing trust in the functionality of the system in this early phase of usage. The assessment of the statements relative to the number of items (see Appendix E) underlines the observation. It shows that in case of one item the satisfaction with the system (Q5) is much lower than for two or three items and the subjects state that one item is not enough (Q2). Thus, decision making becomes harder (Q3) even though the subjects understand why they receive the recommendation (Q4). In 70% of the cases, it also leads to the usage of the list view. On the other hand, the subjects rather trust the data source (S8 in Figure 8.9).

Perceived Ease of Use (EOU)

Ease of use (EOU) is not explicitly part of the study but we check if the subjects also perceive the interaction as easy as we attempt it to be with our design. Overall, the subjects do not seem to have trouble with the usability of the system (Figure 8.9). It is easy to learn system functionality (S3) and to use it (S1). Also, they are not distracted from driving (S5). Similar results can be found in the semantic differential (Figure 8.10). Strong agreement exists for tractability, pleasantness and convenience.

Attitude to Use

The subjects strongly agree (S11 in 8.9) that they have the intention to use our recommender in their own cars. In the semantic differential (Figure 8.10), the subjects strongly agree to adjectives such as good, desirable and innovative. It reflects a positive attitude towards the system. They also rather agree that the implementation is professional.

8.2.4. Discussion

We derive from the results of our study that a positive attitude towards using proactive in-vehicle recommendations is available. All external variables for the usefulness (U) (relevance, utility, transparency and expectation) are rated mostly positive. The subjects also confirmed ease of use (EOU) of the system. Because the subjects are confused by recommendations with one item, we believe that trust is an important indicator for long-term usage of the system. Results from the area of mobile services confirm our assumption (Kaasinen [Kaa05]). The investigation of trust is out of scope in this thesis. The position of the selected item in the recommendation list and the reaction of the users to one item show that it makes sense to recommend more than one item, i.e., to provide choice. Only half of the selected items are the first one in the list and most subjects mention that they associate a ranking with the list. Overall, our system is less seen as a feature that raises alertness but as useful and assisting.

The justification explanations for the reason of a recommendation seem to be enough to make the system comprehensible as the detail view for comprehension explanations is never used and the subjects mostly understand why they receive the recommendation. Information in this view may become useful in long-term usage where the subjects are on their own. A clearer and more precise information delivery may improve our proactive recommender, which may also contribute to more trust. The follow-up questionnaire shows that almost all subjects would prefer clear facts instead of fuzzy interpretations. Moreover, some subjects feel patronized by assessment such as "short detour" or "low priced". The results confirm our observation towards interpreted arguments in the offline study in Chapter 7. Additionally, showing only the most significant argument at the first place seems to be too dynamic for the subjects. They prefer a fixed order of the arguments to be able to scan the attributes of an item quickly. This is only observable in the driving study and was not clear in our offline study in Chapter 7.

It is interesting to see that the interaction with the map changed significantly compared to the user study in Chapter 7. In general, users are familiar with map interaction with online services such as Google Maps. Therefore, our subjects browsed the map during our offline study frequently. While driving, browsing of the map was never carried out. No user asked for the functionality to browse the map. On the other hand, information on the map is probably relevant because it shows the location of the items. Our automatic adjustment of the map area seems to provide enough information towards the map.

We confirm the observation from the user surveys in the previous chapter that gas station decisions are made by comparing the price and detour of a gas station. It becomes clear by interviewing the subjects about their expectations of a gas station recommender and also during the scenarios. Even those subjects with low preferences for the price also look for the price during decision making. The gas level at the gas station is also important in the preferences but is no primary decision dimension. The subjects expect the system to consider this information in the assessment but they do not need the information to make a decision. The same accounts for an economic optimum between price and detour. It is expected to be calculated by the system but is not missed in the decision making.

We are able to confirm the results of Iyengar and Lepper [IL00] towards choice in the context of proactive recommendations. The preliminary study in Section 7.1 shows that only a few items are desired by the users to be recommended. However, our observation in the user study reveals that only one item is not enough. Providing two or three items shows better results than only one item. The difference in the assessment between two and three items is not significant. Therefore, we think that more than three items are not needed.

8.3. Summary

To investigate user acceptance of P-IVRS, we used the common technology acceptance model (TAM) and adapted it to the conditions of P-IVRS. The model describes user acceptance by the aspects perceived ease of use (EOU) and usefulness (U). Our main focus of investigation was the usefulness of the system. We implemented our concept of an explanation user interface in a real user interface that is located in the CID of the car. The use case was a gas station recommender system. The user interface is designed according to the three major requirements on an interface for proactive recommender systems: unobtrusiveness, accessibility and transparency. As our prototype is in an early stage of development it is crucial to take ease of use into regard. Otherwise, it may has negative influence on usefulness. Therefore, we carried out an expert study towards the design of the interface.

In a user study, the drivers used the prototype inside our test vehicle. We gave the drivers different refilling scenarios and let them choose a gas station while driving as soon as the recommender pops up. The method of "think aloud" was used along with statements the drivers had to assess. We determined relevance of items, utility of items, expectations for the system and transparency as external variables for usefulness. The results show that the drivers have a positive attitude towards using such a system. They confirm that the system is easy to use and useful. However, information should be delivered more clearly and providing choice is an important issue. We observed that it is better to provide choice among two or three items instead of no choice with one recommended item.

9. Conclusions and Outlook

In the final chapter, we summarize the major findings in this thesis. Based on these findings, we give an outlook to future research that could not be solved in this thesis but is worth to regard.

9.1. Summary and Conclusions

Our research is targeted to drivers in an automotive environment. The primary task of the drivers is driving. In general, they should be as little as possible be distracted from this task. Therefore, consuming information inside a car is challenging. Interaction with in-vehicle information systems (IVIS) is restricted in contrast to desktop systems. Additionally, information overload can also be another reason for driver distraction in case of IVIS. Making use of context information contributes to more intelligent behavior inside a car. Intelligent behavior is able to lower information overload and driver distraction. Context information describes situations of the drivers that mainly influence how relevant information for the drivers is. Filtering information according to situational knowledge is a promising way to reduce the amount of information. On the other hand, the users should be given a choice between alternatives.

Besides context, the preferences of the users are another indicator for the relevance of information. Recommender systems bring together user preferences and context with the items of interest. They distinguish useful and useless items for the users. Classical approaches are collaborative filtering, content-based filtering and knowledge-based filtering. Exploiting context in recommender systems became popular in recent years.

Although recommender systems are able to lower information load, the interaction inside a car still stays a challenge. Enabling recommender systems to act proactively is able to contribute to less driver distraction through interaction. Proactivity turns recommender systems into assistants that show initiative by themselves to help the users, e.g., by recommending something useful that the user has not anticipated. However, making systems proactive is a challenge. A proactive system should be unobtrusive, useful and accessible. To accomplish these requirements, the research on intelligent systems delivers some well-approved methods, e.g., Bayesian networks and fuzzy logic. Though, acting proactively bears the risk that the users do not understand the behavior of the system anymore. Hence, they may become dissatisfied. Adding more transparency to the system may improve the comprehensibility of the system. Additional explanations are a popular method to add transparency.

The goal of this thesis was to investigate recommender systems that behave proactively to assist the driver and lower information load and driver distraction. We analyzed related approaches from the area of IVIS, proactive desktop and proactive mobile recommender systems to see if there are already solutions that can be applied to our problem. The result of the literature review is that available solutions only cover our problem to some extent. Especially the automotive environment restricts the application of available solutions. The focus of proactive systems in the area of IVIS is either on the user interface itself or on information filtering. Proactive desktop recommender systems mostly try to monitor the users. This requires a lot of interaction with the system. If machine learning is used, ramp up and cold start problems arise. Proactive mobile recommender systems are less interaction intensive and use context but mobile users are different to drivers. Mobile users are able to focus their full attention to the mobile device, whereas the drivers are not able to interrupt driving. Transparency is either out of scope of the proposed proactive recommenders, implicitly solved by design or targeted to special groups such as shoppers to persuade them to buy, e.g., *Amazon.com*. Finally, only a few approaches are investigated towards user acceptance. We believe that it is important to address all of these issues to design a proactive recommender system inside a car.

We approached the problem more formally by giving a definition of the scope of proactive in-vehicle recommender systems (P-IVRS) and by describing use cases and requirements. The use case of a gas station recommender was followed throughout our investigation. We distributed functional requirements on different parts of a software system to be able to investigate them separately. These components are for situation awareness, calculation of recommendations and generating explanations. A central part of our system is decision making. It combines the proactivity of the system with the recommendation engine. We proposed a combination of a satisficing method with optimizing items. The idea is to recommend optimal items in a situation that is good enough to recommend. This lays more importance to the quality of the items than on the situation.

In the first part of our research, we investigated the software component for situation awareness. Situation awareness represents the knowledge base upon which the system makes decisions and selects items. It provides information about the benefit of a recommendation in a situation and how user preferences are in this situation. We proposed a four-level framework based on human situation awareness. It comprises the perception of context information that determines a situation, the comprehension of the influence of a situation, the prediction of the development of this situation and the resolution of information that can be used by other parts of the system. Our framework differs from other approaches by the goal of the inference. Some approaches are task-oriented, i.e., they monitor the user while performing a task and infer on the need of information based on user's actions. Other approaches infer on the current or upcoming task of a user and deliver information associated with that task. Our framework aims to infer on information need directly. We proved that our framework is applicable to our problem by implementing the framework with models based on fuzzy logic and Bayesian networks. Many approaches in this research area use Bayesian networks to model the intelligent behavior of the system. Fuzzy logic is often used to represent context or situations. With few situations and a clear understanding of the behavior of the system, fuzzy logic showed good results. With more situations and dependencies between the situations, rules became more complex. Bayesian networks showed better results in this case. Although our situations are modeled as fuzzy variables, they can easily be applied within a Bayesian network. We showed that inference in the network based on fuzzy variables is a promising combination.

Our second system component is the recommender system. The main requirements on the recommender are that it should avoid cold start problems and should be able process context information. Our literature review showed that proposed approaches of context integration are either tailored to a specific recommendation method such as collaborative filtering or they use simple linear models. Our approach combines the well-known paradigms of prefiltering and postfiltering for context integration. This shrinks the set of available items and is then able to incorporate different multi-criteria decision making (MCDM) methods. The knowledge of the usefulness of an item for a user is stored in utility functions. These utility functions are used to calculate scores for the evaluation of items. Predefined utility functions help us to avoid ramp up and cold start problems. For most use cases of a P-IVRS in the automotive area, the route of the driver is important context information. By means of a user study, we investigated the most important elements of the route as context in the perception of the users. The results reveal that the detour to drive to a POI is the most important element. The implementation of our context integration model for gas station recommendations made it possible to evaluate our approach and to assess different MCDM methods. Our results show that comparative models such as AHP and TOPSIS predict user preferences more accurate than linear models such as WSM or WPM. Comparative models also outperform variants of a dominance-based Skyline filter.

The third component of our proactive recommender tries to make the system more comprehensible for the users by means of explicit explanations. We designed two methods to generate explanations. One method was designed for the selected items and one method for the situation awareness of the system. A preliminary study reveals that the explanations should be short and based on user preferences and the situation. We presented a score-based method to generate item explanations. This ties in with our score-based recommendation approach. An offline user study with an implementation of our method showed that not enough arguments in an explanation make it difficult for the user to make a decision. It also reveals the dependency of the quality of explanations to user preferences and the items that are recommended. In contrast to our item explanation method, situations awareness is explained depending on the method that is used to model proactive behavior. We proposed a method to explain the reasoning of our Bayesian network. Our method results in situations that significantly influence the decision of the system towards the delivery of a recommendation. Hence, it justifies the decision of the system. With different examples, we showed that the generated justification explanations are reasonable.

After all necessary components have been investigated separately, we carried out an infield user acceptance study with a complete system based on the technology acceptance model (TAM). For that, we implemented all system components and added an in-car user interface. The system was deployed inside a test car. The user interface was evaluated with an expert study to ensure easy of use for the drivers. Though, our main focus was on the usefulness of the system. We found that the drivers have a positive attitude to use such a system. Furthermore, choice between more than one item and the presentation of information in a clear manner are important factors to make the system useful. Although our proactive recommender contains several complex models, the drivers are only able to recognize a small amount of this complexity during usage. Especially situation awareness was mostly hidden to the users. The drivers mainly noticed the items that are recommended to them. Therefore, we conclude that our optimizing-satisficing (OS) strategy correctly gives more importance to the items when making a decision about what and when to recommend. Furthermore, as the users tend to accept such a system, we believe that our system design is adequate to provide proactive recommendations in automotive environments.

This thesis presents an integrated investigation with all components that we think are important for a proactive recommender inside a car. These components are applied in the automotive area where proactivity is able to make a contribution to less driver distraction. The presented results are promising and we hope that they motivate further research in this area.

9.2. Outlook

Looking at current developments in automotive industry and consumer electronics, we anticipate that the amount of information inside a car will further increase. Car manufacturers have to react on future needs of users. This needs change rapidly with more powerful mobile devices such as tablets or smartphones. A major challenge is to integrate such devices seamlessly into the automotive environment. With cloud computing, the users are already used to have their information at any time and any place. This thesis regards itself as contribution to make relevant information better accessible for the drivers.

We chose a simple and obvious example to explain our approaches. Some of the approaches such as the four-level situation awareness framework might be too complex for a simple gas station recommender. We want to motivate for further research based on our system design with the discussion of a more complex example from Chapter 4. The use case of recommending solutions if drowsiness was detected became a popular research area in recent years. Compared to the gas level indicator it is not simple to derive an indicator for drowsiness. Some approaches use Bayesian Networks (e.g., Yang et al. [YMT+09]) to describe the level of drowsiness with one value. This value can be represented as a fuzzy situation variable with the states "very drowsy", "slightly drowsy", etc. However, there are more situations that determine the need of information in this case. These are the length of the route, the time and distance that a driver already drove in one, the time of day and drivers' intention at the destination. In case of drowsiness, several POI categories might be relevant, e.g. restaurants, hotels or resting places. Hence, the challenge is to derive a model that brings together the situations and the POI categories towards need of information. This is similar to the Lumiere Project of Horvitz et al. [HBH+98] where user actions are connected with help topics. Furthermore, fatigue develops itself slowly over a long period of time and a recommendation before the driver becomes seriously drowsy increases the security. Information like the circadian rhythm or body clock allows to predict how drowsiness might develop and can be described as a fuzzy situation variable.

Just like gas stations, the utility of resting places, hotels or restaurants also depends on context. Our approach with multiple scores has some advantages in case of several categories. With local scores, items within a category can be assessed. The global score allows an overall assessment which is able to compare items with different categories. Finally, a recommendation with several categories based on drowsiness is complex and might not be easy to understand for the users without explanation. Drivers who need to refill their tanks look automatically for gas stations. However, a tired driver does not necessarily expect recommendations. An explanation like "You received these recommendations because you are driving since five hours without stop" remembers the driver why a break is important.

Findings in this thesis provide a first tendency of the application of proactive recommender systems inside a car. Proactive recommender systems are one possible method to make information accessible for the drivers. Our field survey covered a time period of one hour for each test driver. For this, we created artificial situations in which we interviewed the drivers. Our test drivers accepted our proposed system in general and confirmed its usefulness. Therefore, we believe that is its worth to carry out further investigations to get a profound assessment of such systems. The drivers should use the system in their everyday life for a longer period. This will show whether the users keep using the system after the first impression. The challenge is to derive meaningful results because a survey without a guide contains more noise.

We think that the user interaction with the system has a major influence on the long-term acceptance of the system. Experts confirmed that our presented in-vehicle user interface was suitable for our investigation. As our investigation showed, the drivers would accept a P-IVRS in general. For a long-term acceptance, we think that further improvements on the user interface should be made. Especially the presentation of information seems to be a key factor. This was often noted by the participants of our survey. However, it is also possible to take other mediums inside a car into regard to display information. The instrument cluster and the head-up display are common output devices in the car that are able to support information delivery in the CID. Furthermore, an up-to-date research topic in car industry is to improve route guidance with augmented reality (e.g., Akaho et al. [ANY+12]). BMW [BMW11] presented at its in-house event "Innovationstage ConnectedDrive 2011" ideas for a "contact analogous" head-up display that should be able to place guidance markings into the environment. Also app developers from iOnRoad [iL12] show a similar approach. Recommendations for POIs can be placed into the environment on the location where the facility actually is.

Besides the display of information, alternative ways for the input of data should also be investigated. Our system design focused on the intelligence to find suitable items and a suitable situation to recommend. Input was possible with the iDrive Controller. However, unknown influence factors to the system may lead to different user needs as anticipated. The opportunities of our system to correct its decisions are very limited currently. A speech-based dialog between the user and the system would improve our system and help the users to find a suitable item in case of wrong recommendations.

The best interface becomes useless, if the recommendations are not suitable. We showed the applicability of MCDM methods to calculate suitable items. It would be interesting to compare our MCDM-based recommender with other context integration methods based on collaborative filtering or content-based filtering. For this, different use cases that also involve taste, e.g., restaurants, should be used. In case of our use case of a gas station recommender, statements from the participants of our surveys indicate that the experience of the user with the task and the items has an influence on the selection of items. The incorporation of users' experience by means of a content-based recommender may improve our recommendation results.

The accuracy of a recommender also depends on the quality of the underlying data. In the use cases that we presented, several sources of content can be community-based and are subject to inaccuracy, incompleteness or are outdated. For instance, gas prices or free parking lots are very dynamic. It should be investigated how much the quality influences the acceptance of a P-IVRS. After all, the users already have experience with such content nowadays and use them anyhow, e.g., on their smartphones. Another information source that drives the suitability of recommendations is user data. It would be interesting to see how much information the drivers are willing to give about themselves to improve the accuracy of the system. There is probably a limit where the system may become scary because it acts proactively in a very accurate way. This may happen in combination with information from social networks or sensitive user information such as drowsiness. Finally, there is more than one information source that is of interest for the users at a time. A mixture of several proactive recommenders that all have information for the driver separately, e.g., restaurants, parking lots and gas stations, is difficult. The challenge is to aggregate the goals of each proactive assistant to avoid disturbing and interrupting the driver several times.

We think that comprehensibility is an important factor for the acceptance of a P-IVRS. Our field and offline studies showed the usefulness of item explanations. However, it was not possible to investigate the usefulness of situation explanations with our simulated situations. The explanations were to obvious to be recognized as useful. However, we think that such explanations are useful. Therefore, further explicit investigations of situation explanations should be carried out. These explanations may become useful in a long-term survey where the users are on their own. The explanations can also be improved by incorporating temporal aspects. Furthermore, the display of text is very restricted inside a car. Therefore, we focused on short explanations. A more sophisticated explanation such as "Recommendations are given later because you have enough gas now" may improve the comprehensibility of the system further. However, the challenge is to present such explanations without distraction.

If proactive recommender systems are offered commercially inside a car, they may alter the behavior of the drivers in regard of the consumption of items. The ADAC [ADA12] carried out a user questionnaire in Germany towards the behavior of drivers when they refill their car. The questionnaire reveals that 41% of the drivers never or only seldom compare gas prices, 43% refill only when the tank is empty and 40% always use the same gas station. The usage of a gas station recommender may change such behavior. The benefit would be to save money without much effort. Especially, in the last couple of years, gas prices increased dramatically. We think that there is already enough content to cover user's information need in many cases. However, today it needs too much effort to access this content while driving. Leshed et al. [LVR+08] emphasize that behavior change occurred with in-vehicle navigation systems. People differently interpret their environment and navigate through it. This may also be possible with the consumption of items. A tendency can already be seen with commercial offers of recommender systems such as *Amazon.com*, *Last.fm* or *Yelp.com*.

A. Satisficing-Optimizing (OS) Strategy

The procedure in Listing 1 shows our optimizing-satisficing (OS) strategy. The method builds item sets SE out of all assessed items I^* by trying to add a new item i to a set recursively. When the set size reaches the maximum number of items num or an item does not fulfill any other constraints relative to other items in the set, it is not added. The same applies if the item cannot be delivered in a satisficing situation. Resulting sets are pushed in a priority queue Q that is sorted according to the expected utility eu. The expected utility is calculated over all items in the set. The procedure returns the set SE_{max} with the highest expected utility which is the first element of Q.

The satisficing procedure in Listing 2 analyzes the horizon H^* between the current context $o_{Current}$ of the user and the delivery context of the item $p_{Consumption}$. The delivery context is the last chance in the horizon H to deliver an item. For instance, the delivery context of a restaurant depends on its location or opening times. The delivery context represents the time period before the restaurant is passed or before it closes plus an additional time to eat. To check if the item is satisficing, the task benefit b and the delivery costs c for each context in H^* are calculated by means of the situation awareness model. Costs and benefits are calculated at context o relative to the current context $o_{Current}$. This allows for a decay inside the time frame. As soon as the maximum benefit b_{max} regarded so far is good enough ($\geq t_R$) and higher than the minimum cost of delivery $> \alpha c$ regarded so far, the procedure stops with a positive result. The cost-benefit analysis can be regulated by $\alpha \in [0, \infty)$. For $\alpha < 1$, the delivery mainly depends on the benefit and for $\alpha > 1$ recommendations are less delivered if the costs, e.g., annoyance or safety risks, are too high. Costs c and benefits b have to be in the same value range.

The actual delivery context o_d is in the horizon H^{**}. It lies between the current context $o_{Current}$ and $o_{min} = \min_{i \in SE} consumptionContext(i)$ where the minimum is the closest context in space or time to the current context. This guarantees that a recommendation is not delivered too late, i.e., no item in the set is irrelevant because it is closed or passed. The exact context of delivery depends on the recommendation strategy. A possible strategy is to take $o_{Current}$ if it exceeds a threshold in benefit ($b(o_d) > t$), is not higher than a threshold for the costs ($c(o_d) < t$) or both conditions are fulfilled. Otherwise, the next o that exceeds these thresholds can be taken. Depending on the threshold, there is a risk that no o_d is found. Alternatively, the maximum benefit b_{max}, minimum cost c_{min} or a combination of both ($bc_{max} = \max f(b(o), c(o))$) in H^{**} can be taken. A third possibility is to take another context than $o_{Current}$ if it is better. Again, the difference can be calculated by regarding the costs ($c(o_{Current}) - c(o) > t_b$) or the benefit ($b(o) - b(o_{Current}) > t_c$). Both terms have a maximum of $\max(c)$ or $\max(b)$ and

Procedure 1 optimizeItemSet

Input: I := Items $\wedge$ SE := Current item set $\wedge$ num_c := Number of items in SE $\wedge$ num_{max} = Maximum number of items

Output: SE_{max} := resulting item set with a maximum of $num_{max} items$

Q := Priority queue of item sets SE
for all $i \in I*$ **do**
  $add = true$
  $SE^* = SE \cap i$
  if $SE^* \in Q$ **then**
    $add = false$
  else if $satisficeBenefitCost(i) \neq true$ **then**
    $add = false$
  else
    for all $j \in SE$ **do**
      for all $constraint \in CON$ **do**
        if $constraint(i, j) \neq 1$ **then**
          $add = false$
        end if
      end for
    end for
  end if
  if $add = true$ **then**
    if $expectedUtility(SE^*) > expectedUtility(SE)$ **then**
      $Q = Q \cup SE^*$
      $num_c = num_c + 1$
      if $num_c < num_{max}$ **then**
        $optimizeItemSet(I, SE^*, num_c)$
      end if
    end if
  end if
end for
return $Q[0]$

Procedure 2 satisficeBenefitCost

Input: H := Prediciton horizon $\wedge$ i := Index of position in H $\wedge$ $o_{Current}$ = Context in $H \wedge t_B$ = Threshold for benefit

Output: $true/false$

$o_{Consumption} = consumptionContext(i)$
$H^* := subHorzion(p_{Current}, p_{Consumption}, H)$
b_{max} := Maximum benefit
c_{min} := Minimal cost
for all $p \in H^*$ **do**
    $b = benefit(o, o_{Current})$
    if $b > b_{max}$ **then**
        $b_{max} = b$
    end if
    $c = cost(o, o_{Current}))$
    if $c > c_{min}$ **then**
        $c_{min} = c$
    end if
    if $b_{max} \geq t_B$ **and** $b_{max} > \alpha c_{min}$ **then**
        return $true$
    end if
end for
return $false$

a minimum of $\min(b)$ or $\min(c)$. Hence, to incorporate costs and benefit, we can use Equation A.1. The thresholds t_b, t_c and t_{bc} need to be larger than 0 as we do not want to be worse than $o_{Current}$. For instance, with threshold t_c it is possible to guarantee that recommendations are only given in standing still situations in automotive scenarios (Equation A.1).

$$\frac{(c(p_{Current}) - c(p)) + (b(p) - b(p_{Current}))}{\max(c) + \max(b)} > t_{bc} \tag{A.1}$$

If more than one output channel is used for delivering recommendations, the output at delivery context o_d can be evaluated for all displays. For instance, in a car the head-up display, the instrument cluster and the CID could be used. For each display the threshold t_{bc} is set separately. The costs for a head-up display are much higher than for a central display. Therefore, the recommendation must provide more benefit to be shown in the head-up display, e.g., in case the recommendation is urgent.

B. Situation Awareness

B.1. Fuzzy Membership Calculation for Situations

With the procedure in Listing 3, we derive fuzzy boundaries for membership functions. It calculates the membership for each situation S separately. If the situation is already discrete, we take a singleton membership function. Otherwise, we calculate the boundaries of the membership functions for each state st taking into regard the constraints in Section 4.3. The result is either a trapezoid membership function or a triangle membership function depending on the range of a state. The procedure iterates over the borders of the discretized intervals and determines the x-value of the membership function. A left side with start and end x-value and a right side represents a state st. Each iteration step considers the right side of one interval together with the left side of the following interval. Constraints $R1$ and $R3$ for the memberships of situation states are fulfilled by setting the x-value of the start of the right boundary equal to the x-value to the start of the left boundary of the next state (Lines 21-27). The corresponding y-value is 1 for the right boundary and 0 for the left. Line 21 ensures that $R2$ is fulfilled because the boundaries are adjusted to the mean value of the next state.

Procedure 3 calculateMembership

Input: $SL :=$ Set of lower-level situations
Output: $MSL :=$ map of membership functions for all situations in SL

for all $S \in SL$ **do**
  $m :=$ Memberships
  $p \leftarrow p + dp$
  if $distribution(S) = discrete$ **then**
    $m = memberships(singleton)$
    $MSE \leftarrow < S, m >$
  else
    $SD = discretize(S)$
    for $i = 1 \rightarrow n_{States}$ **do**
      $i \leftarrow i + 1$
      $st_1 = SD(i), st_2 = SD(i+1)$
      if $i = 1$ **then**
        $x_{-1,start} = x_{-1,end} = x_{1,right,start} = \min(st_1)$
      else
        $x_{1,right,start} = (\max(st_1) - \min(st_1))/2$
      end if
      if $st_2 \neq \emptyset$ **then**
        $x_{1,right,end} = \max(st_1) + (\max(st_1) - x_{1,right,start})$
        $x_{2,left,end} = (\max(st_2) - \min(st_2))/2$
        $x_{2,left,start} = \min(st_2) - (\max(st_2) - x_{2,left,end})$
        if $x_{1,right,end} \leq x_{2,left,end}$ **then**
          $x_{2,left,start} = x_{1,right,start}$
          $x_{2,left,end} = x_{1,right,end}$
        else
          $x_{1,right,start} = x_{2,left,start}$
          $x_{1,right,end} = x_{2,left,end}$
        end if
        $m_1 = membership(trapezoid, < x_{-1,start}, 0 >, < x_{-1,end}, 1 >, < x_{1,right,start}, 1 >, < x_{1,right,end}, 0 >)$
        $m \leftarrow m_1$
        $x_{-1,start} = x_{2,left,start}$
        $x_{-1,end} = x_{2,left,end}$
      else
        $m_1 = membership(trapezoid, < x_{-1,start}, 0 >, < x_{-1,end}, 1 >, < \max(st_1), 1 >, < \max(st_1), 0 >)$
        $m \leftarrow m_1$
      end if
    end for
    $MSL \leftarrow < S, m >$
  end if
end for
return MSL

B.2. High-level and Low-level Prediction

Low-level Prediction Process

Procedure 4 predictLowLevel

Input: SL := Set of low-level situations
Output: SLP := Set of low-level situation predictions

$H^* = subHorzion(p_{Current}, p_{End}, H)$
for all $S \in SL$ **do**
  $dc_S = samplingRateOfSituation(S)$
  SLP_S := Set of situation predictions for S
  for $p = p_{Current} \to p_{End}$ **do**
    $c \leftarrow c + dc_S$
    $s_P = predict(S, p)$
    $SLP_S = SLP_S \cup s_P$
  end for
  $SLP = SLP \cup SLP_S$
end for
return SLP

Based on the three types of situations in Section 5.1, the prediction procedure is listed in Listing 4. The sampling rate depends on the situation itself and its type. Static situations have a sample rate equal to the horizon because they do not change. The more dynamic a situation is, the more often it has to be sampled to get a precise picture of its development. The sampling rate also accounts for computational complexity and imprecision. The rate can be increased for less complexity with a tolerance of error. If the prediction is unreliable, only a few samples with high confidence are enough. For instance, the route that the drivers take is uncertain, if they do not enter it into the system. However, the current as well as the destination position can be predicted with more confidence. The underlying prediction method for context itself is not determined here. Prediction uncertainty is handled by fuzzy situation variables. As the memberships to situations are subject to constraints, it is possible to reflect probabilities as measure for uncertainty in the prediction process. The procedure returns the developments of all situations $S \in SL$ where SL is the set of low-level situations relevant for the task TA.

High-level Prediction Process

With low-level predicted situations, it is possible to infer on high-level situations and resolve benefit B and the user preferences U. The procedure for high-level prediction in Listing 5 works similar to the procedure for lower-level prediction. Again, we regard the sub horizon H^* from the current context in the horizon $c_{Current}$ to the end c_{End}. Each low-level situation prediction SLP_S is then sampled with the same rate dp. For

Procedure 5 predictHighLevelandResolve

Input: SLP := Set of lower-level situation predictions $\wedge$ CM := *Set of Comprehension models* $\wedge$ DM := *Set of Resolution models* $\wedge$ SH := *Set of high-level situations*

Output: BP := Set of benefit predictions $\wedge$ UP := Set of user preferences predictions

$dc = samplingRateOfBenefit()$
$H^* = subHorzion(p_{Current}, p_{End}, H)$
$/*$ High level Prediction $P_{High} */$
for $c = c_{Current} \rightarrow c_{End}$ **do**
    $c \leftarrow c + dc$
    for all $SLP_S \in SLP$ **do**
        if $type(S) = TEMPORAL$ **then**
            n := number of stages
            for $q = -n \times dc \rightarrow n \times dc$ **do**
                $q \leftarrow q + dc$
                $st_{Low} = state(SLP_S, q, p)$
                $setEvidence(st_{Low}, CM)$
            end for
        else
            $st_{Low} = interpolate(SLP_S, c)$
            $setEvidence(st_{Low}, CM)$
        end if
    end for
    $/*$ Resolution $P_D */$
    for all $S_H \in SH$ **do**
        $st_{High} = evaluate(S_H, CM)$
        $setEvidence(st_{High}, DM)$
    end for
    $b = benefit(DM)$
    $BP = BP \cup b$
    $u = userPreferences(DM)$
    $UP = UP \cup u$
end for
return $SB \wedge UP$

some situations, their states have be be interpolated. If all dc_S are equal to dc, low-level and high-level prediction could be computed in one procedure. We set each state at a prediction step c as evidence for our comprehension models. Temporal situations are specially handled. For every state of a temporal situation, we also set n states behind and after as evidence in the comprehension model. Finally, the comprehension models infers on each high-level situation SH. This is the input for the resolution model which results in the benefit b and user preferences u for each c.

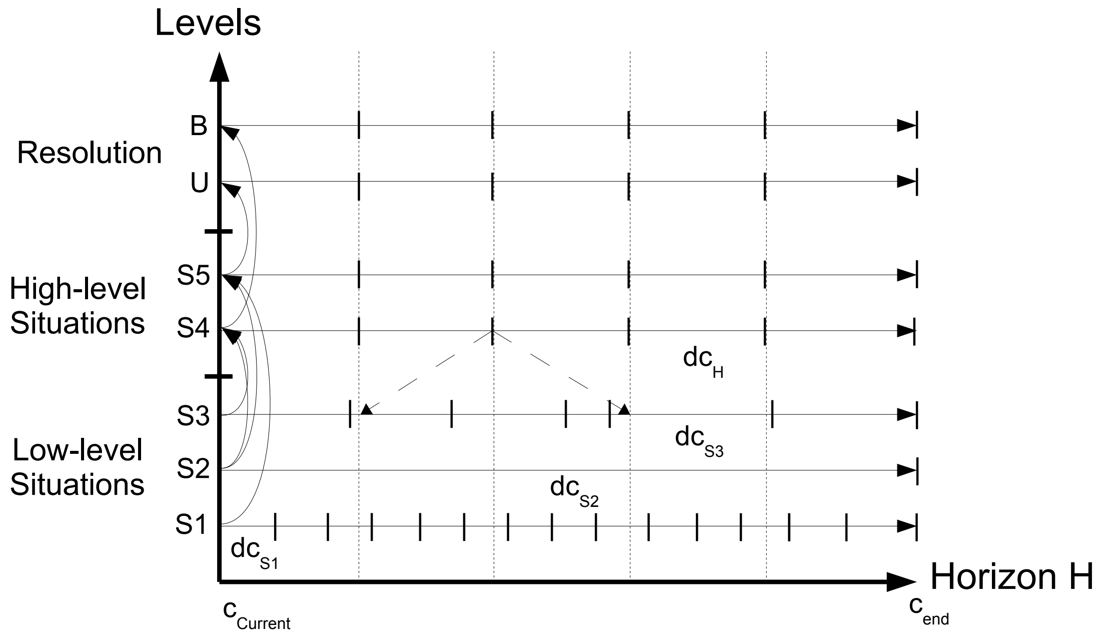


Figure B.1.: Situation awareness based on three situations (S1,S2,S3) along a horizon H with different sampling rates dc

In summary, the whole process is depicted in Figure B.1. Low-level prediction for three situations with different sampling rate is shown in the figure. Situations $S1$ and $S2$ are comprehended towards $S5$ and $S2$ and $S3$ are comprehended towards $S4$. The situation $S3$ is a temporal situation that is evaluated with its states before and after in the high-level situation $S4$. For the temporal situation, the comprehension model needs to incorporate multiple evaluation states of the same situation. Note that n steps in the past and in the future for temporal situations are limited by the boundaries of the horizon H. In the outer areas of the horizon, only $\leq n$ steps are taken into regard. Hence, the method does not incorporate historical information outside the horizon. For instance, we are not interested in the traffic jam the day before.

B.3. Simulation for Data Collection

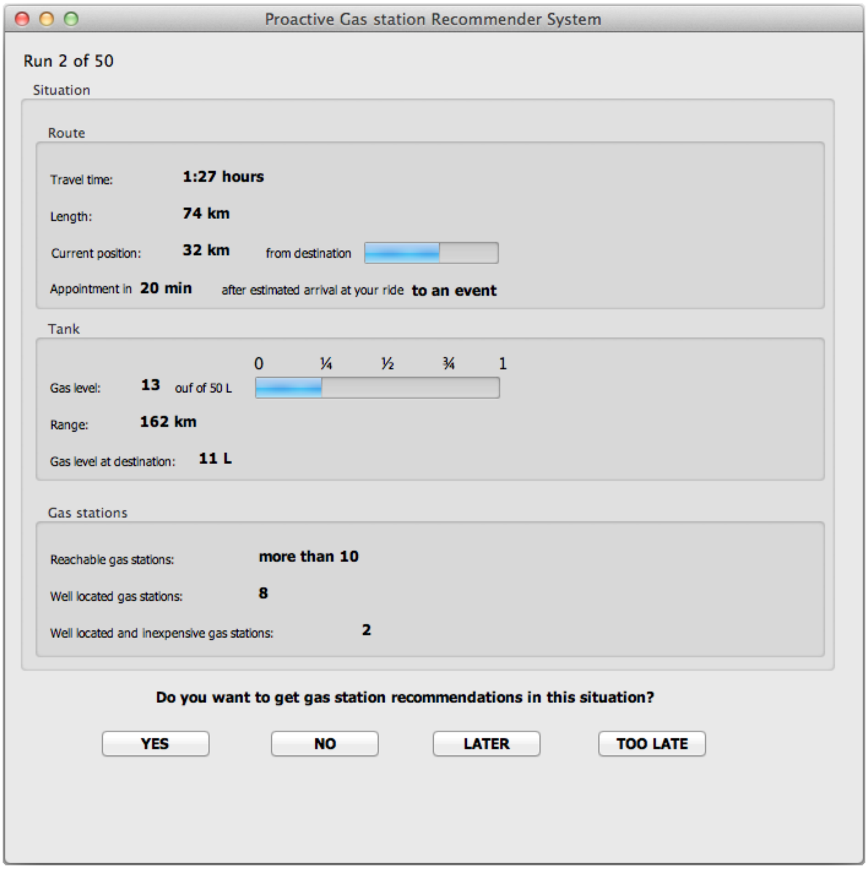


Figure B.2.: Data collection user study interface with simulated situations

Figure B.2 shows the screen in the user study to collect data. The situations are grouped in situations related to the route of the user, situations depending on the gas level and situations resulting from available gas stations. The group "Route" comprises the length and travel time of the route, the distance from the destination and the type of route. Additionally, the users may have an appointment at their destination. The distance from the destination, i.e., the current position on the route, is equally distributed in the simulation. The appointment time is normally distributed with a mean value of 0 and a variance of 20 minutes. The absolute value is taken from negative appointment times to have always values higher 0. The idea is to simulate urgency. The route type and its distribution is chosen based on a mobility study by De la Fuente Layos et al. [de 07]. The type of route influences the length of the route. In average, drivers perform 3 rides a day with a length of 30 to 40 km. In Germany, people spend 17% of this time for work, 6% for education, 11% for shopping, 13% for business trips, 40% for activities in leisure time and 17% for other trips. The study also shows that the longest trips are made in leisure time and the shorter ones for the rest. We defined seven common route types with a Gaussian probability distribution around a mean length. The simulated data for the group "Route" is shown in Table B.1.

Route Type	**Description**	**Mean Length**	**Appointment**	**Probability**
Home	Trip home	short	no	0.20
Vacation	Vacation trip	long	no	0.14
Event	Trip to an event	medium	yes	0.11
Friends	Visiting friends	short	yes	0.11
Leisure	Trip in leisure time	medium	no	0.11
Business	Business trip	short	yes	0.10
Shopping	Going for shopping	very short	no	0.10
Work	Trip to work	short	no	0.13

Table B.1.: Route types in the simulated situations

We split up the leisure trips in leisure time, to friends, events and in other leisure activities. The type "home" is added because most of the trips involve coming home again. The main part of other trips is mapped to vacation trips. The trips for business reasons, to an event and to friends involve some kind of appointment with different urgency. The corresponding length to the types of route is simulated around a mean value with a minimum and maximum value.

In the group "Tank", the gas level is the most important context value. It is simulated with a minimum of 1 liter and a maximum of 50 liter around a mean value of 1 and a variance of 12. Values lower 1 are flipped around 1. Hence, most of the values are in the lower area of 25% of the gas tank where the drivers usually refill their cars. The remaining range and the gas level at the destination can then be calculated with the rest of the tank and the length of the route respectively.

The last group "Gas stations" reflects average properties of available gas stations. The reachability of gas stations in general depends on the current gas level. As the coverage of gas stations is high throughout the area of Germany, the mean value for the Gaussian distribution is calculated by $mean_{reachability} = 80\frac{gaslevel_{current}}{tankvolume}$ with a minimum of 0. Thus, mostly more than 10 gas stations are available. Well located gas stations comprise gas stations that can be reached with a low detour from the current route. The number of reachable gas stations and the length of the remaining route influence the number of well located gas stations. We calculate a random number in the distribution around a mean value of $mean_{location} = 0.8 \times num_{reachablegasstations}$ and flip all values higher than $mean_{location}$. The result is adjusted with $\alpha = 0.4 + distancefromdestination/200$. Finally, we simulate whether some well located gas stations are inexpensive at the same time with a probability of 0.2.

B.4. Perception of Situations

Perceived situations are represented by histograms that are stacked according to the assessment of the user for "Yes", "No" and "Too late". Figure B.3 shows the histogram of subject selections for states of the high-level situation "Proactive Recommendation" PR over all data instances. We exclude the state "Later" because it has a bad performance (see Section 5.2). In Figure B.4, the histograms are depicted for low-level situations.

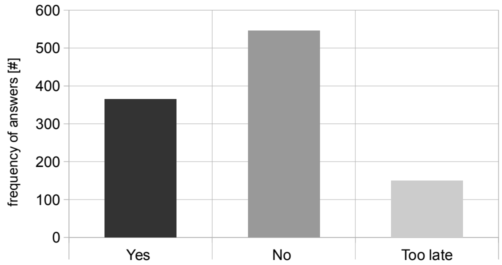
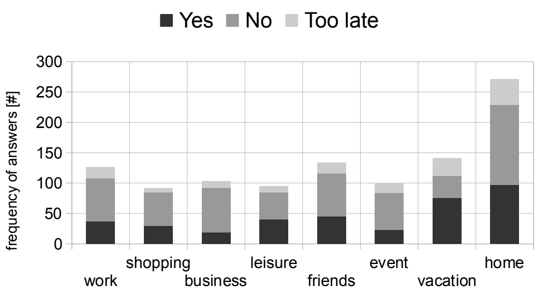
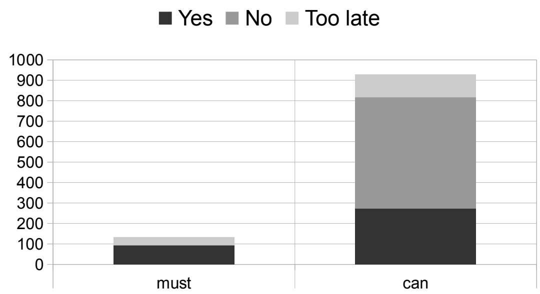


Figure B.3.: Histogram of the high-level situation "Proactive Recommendation" PR

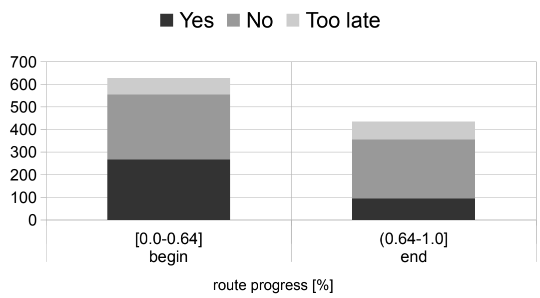


(a) Route type

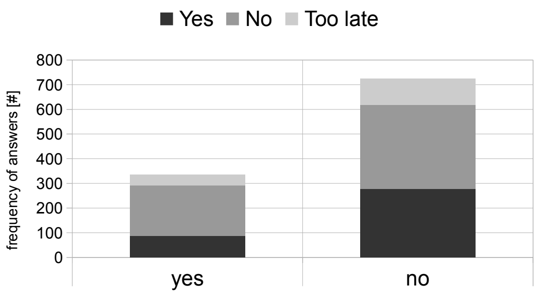


(b) Modality

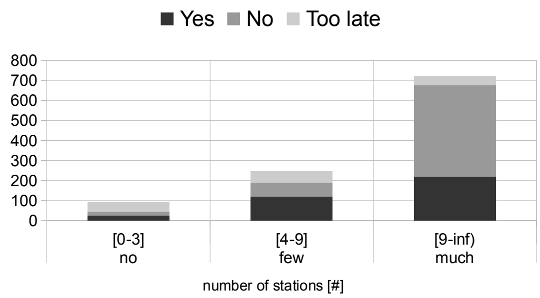
(c) Route length



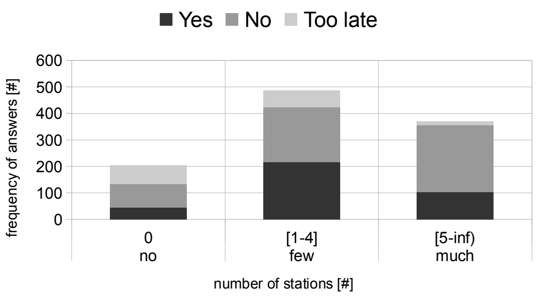
(d) Position on route



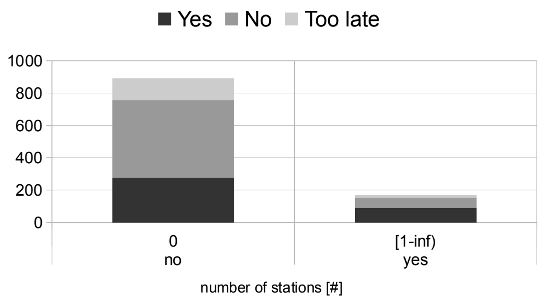
(e) Appointment



(f) Reachability of gas stations



(g) Well located gas stations



(h) Well located and inexpensive gas stations

Figure B.4.: Histograms of low-level situations

B.5. Discretization

Fayyad and Irani [FI93] propose an entropy-based discretization method. It combines a recursive entropy calculation with a minimum description length (MDL) stopping criterion. In each recursion a cutting point is selected that minimizes the entropy of the remaining subintervals. The entropy is calculated for the class C over a set of instances in a subinterval S (Equation B.1).

$$H_C(S) = -\sum_{i=1}^{C} p(C_i, S) \log(p(C_i, S)) \tag{B.1}$$

The lower the entropy, the purer is a subinterval. A cutting point T is evaluated by calculating the weighted average of the two resulting subintervals S_1 andd S_2 (Equation B.2).

$$E(A, T; S) = \frac{|S_1|}{|S|} H_C(S_1) + \frac{|S_2|}{|S|} H_C(S_2) \tag{B.2}$$

The cutting point T for an attribute A lies between the boundaries of the interval S. A series of cutting points are evaluated in each recursion and the one with the a minimum value of $E(A, T; S)$ is selected. The MDL stopping criterion for the recursion corresponds to the aspect that the "simplest" theory out of a set of theories should be selected. In this case, there are two theories: Making a cut or not making a cut. Because we work with entropy, the MDL criterion only accepts a cutting point T if the cost of encoding before the cut is bigger than after the cut. Fayyad and Irani capture this by deriving a gain (Equation B.3).

$$Gain(A, T; S) = H_C(S) - E(A, T; S) \tag{B.3}$$

The gain is compared to a threshold in Equation B.4 where N is the number of values in S. A threshold T is only accepted if the inequality is fulfilled.

$$Gain(A, T; S) > \frac{\log_2(N-1)}{N} + \frac{\triangle(A, T; S)}{N} \tag{B.4}$$

B.6. Feature Selection for Secondary Decision Dimensions

Proactive	**Situation**	**Value**	**Gas**	**CH**	**S**	**C**
Yes	routeLength	(118.5-inf)	(6.6-13.4]	202.0%	0.045	0.774
Too late	routeLength	(118.5-inf)	(2.5-6.6]	172.0%	0.022	0.371
Yes	posOnRoute	(0.64-inf)	(2.5-6.6]	62.9%	0.044	0.353
Yes	posOnRoute	(0.64-inf)	(6.6-13.4]	62.7%	0.029	0.240
Too late	posOnRoute	(0.64-inf)	(2.5-6.6]	160.4%	0.043	0.346
Yes	gasAtDest	(-inf-1.8]	(6.6-13.4]	237.9%	0.029	0.912
Yes	gasAtDest	(4.1-11.2]	(2.5-6.6]	67.0%	0.030	0.376
NO	gasAtDest	(4.1-11.2]	(2.5-6.6]	211.8%	0.038	0.471
NO	gasAtDest	(11.2-inf)	(6.6-13.4]	165.9%	0.024	0.962
Yes	routeType	vacation	(6.6-13.4]	185.6%	0.030	0.711
NO	routeType	business	(6.6-13.4]	146.4%	0.026	0.848
Yes	appointment	yes	(6.6-13.4]	66.4%	0.027	0.254
NO	appointment	yes	(2.5-6.6]	148.5%	0.030	0.330
Yes	wellLocated	(-inf-0.5]	(2.5-6.6]	51.7%	0.024	0.291
NO	wellLocated	(-inf-0.5]	(2.5-6.6]	162.2%	0.029	0.360
Too late	wellLocated	(-inf-0.5]	(2.5-6.6]	161.7%	0.028	0.349
Yes	wellInexpensive	(0.5-inf)	(6.6-13.4]	164.8%	0.034	0.632
Yes	modality	no	(6.6-13.4]	232.0%	0.023	0.889

Table B.2.: Assessment of secondary decision dimensions relative to the gas level (GL)

Proactive	**Situation**	**Value**	**Gas level**	**CH**	**S**	**C**
Yes	posOnRoute	(0.4-inf)	(1.8-4.1]	64.6%	0.025	0.397
Too late	posOnRoute	(0.4-inf)	(-inf-1.8]	156.4%	0.028	0.600
Too late	posOnRoute	(0.4-inf)	(1.8-4.1]	189.8%	0.025	0.397
Yes	gaslevel	(-inf-2.5]	(-inf-1.8]	70.0%	0.047	0.420
Yes	gaslevel	(6.6-13.4]	(-inf-1.8]	152.0%	0.029	0.912
Too late	gaslevel	(-inf-2.5]	(-inf-1.8]	148.9%	0.064	0.571
Too late	gaslevel	(2.5-6.6]	(-inf-1.8]	70.3%	0.023	0.270
Too late	wellLocated	(-inf-0.5]	(-inf-1.805]	186.2%	0.042	0.714
Too late	reachability	(-inf-3.5]	(-inf-1.8]	195.5%	0.028	0.750
Yes	wellInexpensive	(0.5-inf)	(4.1-11.2]	184.7%	0.033	0.593

Table B.3.: Assessment of secondary decision dimensions relative to the gas level at the destination (GLAD)

C. Context Awareness

C.1. Utility Function Study Questionnaire

Nutzerstudie zum Thema „Umweg beim Fahren“

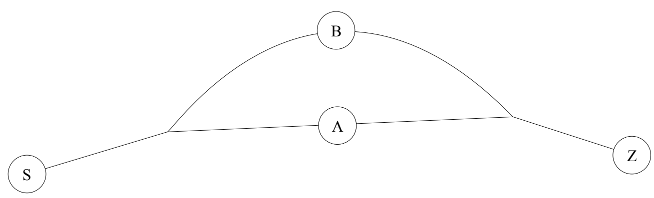
1. Nutzerstudie zu Umweg

In den folgenden Aufgaben werden Sie gebeten, sich in verschiedene Situationen hinein zu versetzten und POIs (Point-Of-Interest) zu bewerten. Ein POI ist ein Ort, der für Sie beim Fahren eine Bedeutung haben könnte, zum Beispiel eine Tankstelle oder ein Restaurant.

Bitte benutzen Sie zum Bewerten eine Skala von 0 bis 10.
Dabei bedeutet:

0	Erfüllt nicht Ihre Anforderungen
3	Erfüllt Ihre Anforderungen zum Teil
5	Erfüllt Ihre Anforderungen mittelmäßig
8	Erfüllt weitgehend Ihre Anforderungen
10	Erfüllt voll Ihre Anforderungen

In allen Situationen können Sie sich die folgende Strecke als Anhaltspunkt vorstellen. Sie fahren von S nach Z. Dabei liegt A genau auf ihrer geplanten Route, während B nur durch einen Umweg zu erreichen ist.



In allen Aufgaben hat A eine Qualität von 1. Die Bedeutung von Qualität wird bei den jeweiligen Aufgaben erklärt. Bitte schreiben Sie in die Tabelle wie gut ein POI ihre Anforderungen erfüllen muss, damit Sie ihn bei dem angegebenen Umweg anfahren würden.

1.1 Umweg bei Tankstellen

Die Qualität einer Tankstelle ergibt sich aus dem Kraftstoffpreis und der Marke .
Da **Tankstelle A eine Qualität von 1** hat, ist sie ziemlich teuer und entspricht nicht der Marke, die Sie bevorzugen.
Beachten Sie beim Bewerten die oben erklärte Skala von 0 bis 10.

Stellen Sie sich vor, dass Sie v**on der Arbeit nach Hause fahren** und tanken müssen.
Die Strecke beträgt **20 km / 20 Minuten**. Wie gut muss B sein, damit Sie dort tanken würden?

Umweg über B in km:	Umweg über B in Minuten:	Qualität von B:
0,5 km	1 min	
1 km	2 min	
2 km – 3 km	3 min – 4 min	
4 km - 5 km	5 min - 6 min	
6 km - 10 km	7 min – 11 min	
11 km - 15 km	12 min - 15 min	
> 15 km	> 15 min	

Stellen Sie sich vor, dass Sie **auf dem Rückweg von Verwandten sind** und tanken müssen.
Die Strecke beträgt **120 km / 90 Minuten**. Wie gut muss B sein, damit Sie dort tanken würden?

Umweg über B in km:	Umweg über B in Minuten:	Qualität von B:
0,5 km	1 min	
1 km	2 min	
2 km – 3 km	3 min – 4 min	
4 km - 5 km	5 min - 6 min	
6 km - 10 km	7 min – 11 min	
11 km - 15 km	12 min - 15 min	
> 15 km	> 15 min	

Stellen Sie sich vor, dass Sie **auf dem Rückweg aus dem Urlaub sind** und tanken müssen.
Die Strecke beträgt **400 km / 200 Minuten**. Wie gut muss B sein, damit Sie dort tanken würden?

Umweg über B in km:	Umweg über B in Minuten:	Qualität von B:
0,5 km	1 min	
1 km	2 min	
2 km – 3 km	3 min – 4 min	
4 km - 5 km	5 min - 6 min	
6 km - 10 km	7 min – 11 min	
11 km - 15 km	12 min - 15 min	
> 15 km	> 15 min	

Welche Aspekte und Situationen beeinflussen ihre Wahl einer Tankstelle im Allgemeinen (z.B. Marke, Benzinpreis, Wetter, usw.)

1.2 Umweg bei Restaurants

Die Qualität eines Restaurants ergibt sich aus der Art der Küche, der Preislage, der Freundlichkeit des Personals, dem Ambiente etc. . Da das **Restaurant A eine Qualität von 1** hat, entspricht es nur wenig Ihren Wünschen. Beachten Sie beim Bewerten die oben erklärte Skala von 0 bis 10.

Stellen Sie sich vor, dass Sie **von der Arbeit nach Hause fahren** und haben Hunger.
Die Strecke beträgt **20 km / 20 Minuten**. Wie gut muss B sein, damit Sie dort essen würden?

Umweg über B in km:	Umweg über B in Minuten:	Qualität von B:
0,5 km	1 min	
1 km	2 min	
2 km – 3 km	3 min – 4 min	
4 km - 5 km	5 min - 6 min	
6 km - 10 km	7 min – 11 min	
11 km - 15 km	12 min - 15 min	
> 15 km	> 15 min	

Stellen Sie sich vor, dass Sie **auf dem Rückweg von Verwandten sind** und haben Hunger.
Die Strecke beträgt **120 km / 90** Minuten. Wie gut muss B sein, damit Sie dort essen würden?

Umweg über B in km:	Umweg über B in Minuten:	Qualität von B:
0,5 km	1 min	
1 km	2 min	
2 km – 3 km	3 min – 4 min	
4 km - 5 km	5 min - 6 min	
6 km - 10 km	7 min – 11 min	
11 km - 15 km	12 min - 15 min	
> 15 km	> 15 min	

Stellen Sie sich vor, dass Sie **auf dem Rückweg aus dem Urlaub sind** und haben Hunger.
Die Strecke beträgt **400 km / 200 Minuten**. Wie gut muss B sein, damit Sie dort essen würden?

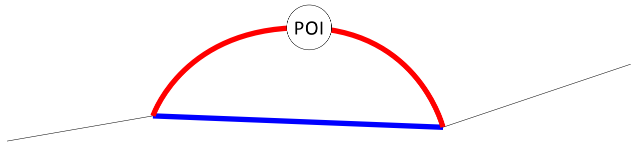
Umweg über B in km:	Umweg über B in Minuten:	Qualität von B:
0,5 km	1 min	
1 km	2 min	
2 km – 3 km	3 min – 4 min	
4 km - 5 km	5 min - 6 min	
6 km - 10 km	7 min – 11 min	
11 km - 15 km	12 min - 15 min	
> 15 km	> 15 min	

Welche Aspekte und Situationen beeinflussen ihre Wahl eines Restaurants im Allgemeinen (z.B. Service, Typ, Wetter, usw.)

2. Relevanz bestimmter Angaben für die Entscheidung

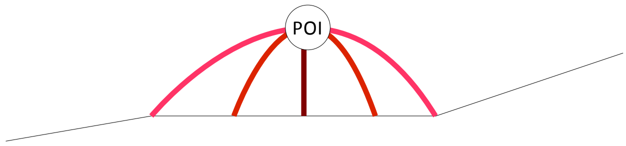
Bitte bewerten Sie die Relevanz folgender Angaben für Ihre Entscheidung für einen POI.

Ist für Sie die Länge bzw. die Zeit des gesamtem Umwegs bei der Entscheidung für einen POI wichtig? (Rot - Blau)



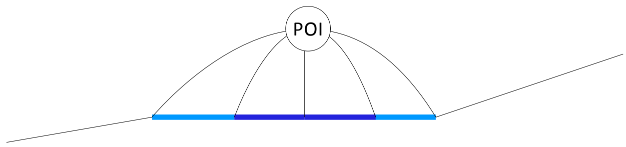
Sehr wichtig ○ ○ ○ ○ ○ ○ ○ ○ ○ ○ Gar nicht wichtig ○ weiß nicht

Ist für Sie die Länge bzw. die Zeit der Abweichung von der Route bei der Entscheidung für einen POI wichtig? (Hellrot -> große Abweichung, Rot -> mittlere Abweichung, Dunkelrot -> kleine Abweichung)



Sehr wichtig ○ ○ ○ ○ ○ ○ ○ ○ ○ ○ Gar nicht wichtig ○ weiß nicht

Ist für Sie die Länge bzw. die Zeit des Stückes der ursprünglichen Route, das Sie beim Anfahren eines POIs auslassen würden, wichtig? (Hellblau -> großes Stück, Blau -> kleines Stück)



Sehr wichtig ○ ○ ○ ○ ○ ○ ○ ○ ○ ○ Gar nicht wichtig ○ weiß nicht

Ist für Sie ein Verlassen eines höheren Straßentyps, z.B. einer Autobahn, auf einen niedrigeren Straßentyp, z.B. einer Dorfstraße, bei Ihrer Entscheidung für einen POI wichtig?

Sehr wichtig ○ ○ ○ ○ ○ ○ ○ ○ ○ ○ Gar nicht wichtig ○ weiß nicht

Ist für sie die Angabe der Länge oder der Zeit, die sie mehr für einen POI fahren müssen, bei der Entscheidung für einen POI wichtiger?

Kilometer ○ ○ ○ ○ ○ ○ ○ ○ ○ ○ Minuten

Anmerkungen:

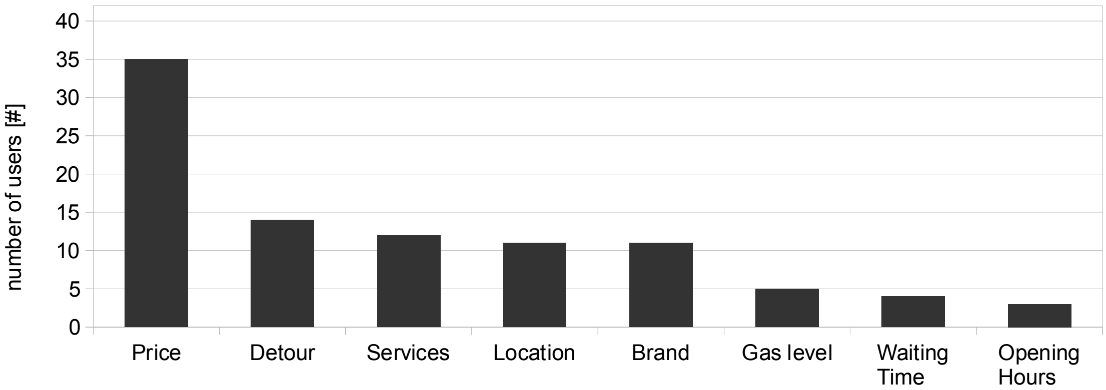
3. Demografische Daten

Geschlecht	○ Männlich	○ Weiblich			
Alter	○ 18 - 30	○ 30 - 40	○ 40 - 50	○ 50 - 60	○ >60
Status	○ Schüler/ Student	○ Doktorand	○ Berufstätiger	○ Rentner	○ Sonstiges
Auto Nutzung	○ Nie	○ 1x pro Monat	○ 1x pro Woche	○ Mehrmals pro Woche	○ Täglich

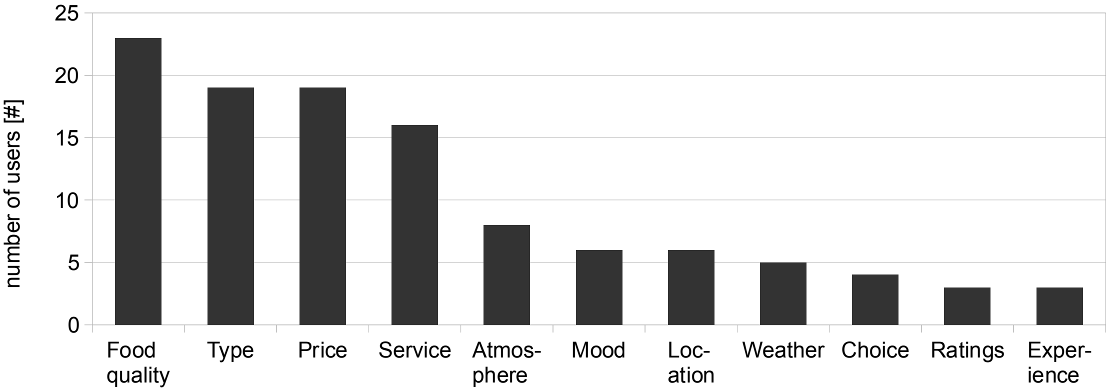
Vielen Dank für Ihre Teilnahme.

Senden

C.2. Criteria for Item Selection



(a) Gas stations



(b) Restaurants

Figure C.1.: Criteria for item selection

C.3. Evaluation results

The tables show the results for every POI set and the average over all POI sets. Each run involves a different combination of MCDM methods on level 1 and level 2.

Level 1		Level 2		Results for POI sets					
MCDM	**Norm**	**MCDM**	**Norm**	**1**	**2**	**3**	**4**	**AVG**	**VAR**
AHP	1	AHP	1	0.62	0.69	0.74	0.58	0.66	0.005
TOPSIS	0	TOPSIS	0	0.82	0.55	0.63	0.58	0.65	0.014
TOPSIS	1	TOPSIS	0	0.82	0.55	0.63	0.58	0.65	0.014
WSM	0	TOPSIS	0	0.77	0.63	0.60	0.54	0.64	0.010
AHP	1	TOPSIS	1	0.64	0.60	0.71	0.56	0.63	0.004
TOPSIS	1	AHP	1	0.59	0.61	0.73	0.55	0.62	0.006
TOPSIS	0	AHP	1	0.59	0.61	0.73	0.55	0.62	0.006
WSM	0	AHP	1	0.60	0.59	0.72	0.50	0.60	0.008
AHP	1	WPM	1	0.62	0.55	0.68	0.48	0.58	0.008
AHP	1	WSM	1	0.57	0.59	0.74	0.38	0.57	0.022
WSM	0	WSM	0	0.64	0.60	0.57	0.44	0.56	0.007
TOPSIS	0	WSM	0	0.61	0.60	0.58	0.39	0.54	0.011
TOPSIS	1	WSM	0	0.61	0.60	0.58	0.39	0.54	0.011
AHP	1	WSM	0	0.36	0.57	0.71	0.47	0.53	0.023
WSM	0	WPM	0	0.69	0.53	0.58	0.26	0.51	0.033
TOPSIS	0	TOPSIS	1	0.57	0.44	0.51	0.43	0.49	0.004
TOPSIS	1	TOPSIS	1	0.57	0.44	0.51	0.43	0.49	0.004
TOPSIS	0	WPM	1	0.55	0.53	0.53	0.33	0.49	0.011
TOPSIS	1	WPM	1	0.55	0.53	0.53	0.33	0.49	0.011
WSM	0	TOPSIS	1	0.55	0.48	0.52	0.39	0.48	0.005
TOPSIS	0	WPM	0	0.70	0.37	0.57	0.30	0.48	0.033
TOPSIS	1	WPM	0	0.70	0.37	0.57	0.30	0.48	0.033
TOPSIS	0	WSM	1	0.54	0.47	0.52	0.29	0.46	0.013
TOPSIS	1	WSM	1	0.54	0.47	0.52	0.29	0.46	0.013
WSM	0	WPM	1	0.54	0.42	0.52	0.25	0.43	0.018
WSM	0	WSM	1	0.53	0.38	0.51	0.26	0.42	0.015
AHP	1	TOPSIS	0	0.11	0.26	0.20	0.60	0.29	0.045
RANDOM				0.27	0.25	0.32	0.15	0.25	0.005
AHP	1	WPM	0	0.03	0.03	0.43	0.33	0.20	0.043

Table C.1.: Precision of MCDM methods on level 1 and 2

Level 1		Level 2		MRE for POI sets				
MCDM	**Norm**	**MCDM**	**Norm**	**1**	**2**	**3**	**4**	**AVG**
AHP	1	WPM	0	0.14	0.13	0.08	0.19	0.13
AHP	1	WPM	1	0.14	0.13	0.08	0.19	0.13
WSM	0	TOPSIS	0	0.14	0.17	0.08	0.18	0.14
WSM	0	TOPSIS	1	0.14	0.17	0.08	0.18	0.14
TOPSIS	0	AHP	1	0.22	0.14	0.07	0.17	0.15
TOPSIS	0	TOPSIS	0	0.14	0.14	0.15	0.18	0.15
TOPSIS	0	TOPSIS	1	0.14	0.14	0.15	0.18	0.15
TOPSIS	1	TOPSIS	0	0.14	0.14	0.15	0.18	0.15
TOPSIS	1	TOPSIS	1	0.14	0.14	0.15	0.18	0.15
TOPSIS	1	AHP	1	0.24	0.14	0.07	0.17	0.15
AHP	1	TOPSIS	0	0.27	0.09	0.12	0.15	0.16
AHP	1	TOPSIS	1	0.27	0.09	0.12	0.15	0.16
WSM	0	AHP	1	0.23	0.14	0.08	0.18	0.16
TOPSIS	0	WSM	0	0.14	0.19	0.07	0.24	0.16
TOPSIS	0	WSM	1	0.14	0.19	0.07	0.24	0.16
TOPSIS	1	WSM	0	0.14	0.19	0.07	0.24	0.16
TOPSIS	1	WSM	1	0.14	0.19	0.07	0.24	0.16
WSM	0	WSM	0	0.14	0.20	0.08	0.25	0.17
WSM	0	WSM	1	0.14	0.20	0.08	0.25	0.17
WSM	0	WPM	0	0.14	0.20	0.08	0.25	0.17
WSM	0	WPM	1	0.14	0.20	0.08	0.25	0.17
TOPSIS	0	WPM	0	0.14	0.23	0.07	0.25	0.17
TOPSIS	0	WPM	1	0.14	0.23	0.07	0.25	0.17
TOPSIS	1	WPM	0	0.14	0.23	0.07	0.25	0.17
TOPSIS	1	WPM	1	0.14	0.23	0.07	0.25	0.17
AHP	1	WSM	0	0.22	0.16	0.08	0.23	0.17
AHP	1	WSM	1	0.22	0.16	0.08	0.23	0.17
AHP	1	AHP	1	0.28	0.17	0.07	0.17	0.17

Table C.2.: Mean Recommendation Error (MRE) of the predicted item with the highest global score

Level 1		**Level 2**		**Probability for POI sets**				
MCDM	**Norm**	**MCDM**	**Norm**	**1**	**2**	**3**	**4**	**AVG**
WSM	0	WSM	0	0.03	0.06	0.03	0.17	0.07
WSM	0	WSM	1	0.03	0.06	0.03	0.17	0.07
AHP	1	WPM	0	0.03	0.06	0.03	0.11	0.06
AHP	1	WPM	1	0.03	0.06	0.03	0.11	0.06
WSM	0	TOPSIS	0	0.03	0.11	0.03	0.09	0.06
WSM	0	TOPSIS	1	0.03	0.11	0.03	0.09	0.06
WSM	0	WPM	0	0.03	0.06	0.03	0.17	0.07
WSM	0	WPM	1	0.03	0.06	0.03	0.17	0.07
TOPSIS	0	WSM	0	0.03	0.06	0.03	0.17	0.07
TOPSIS	0	WSM	1	0.03	0.06	0.03	0.17	0.07
TOPSIS	1	WSM	0	0.03	0.06	0.03	0.17	0.07
TOPSIS	1	WSM	1	0.03	0.06	0.03	0.17	0.07
TOPSIS	0	WPM	0	0.03	0.09	0.03	0.17	0.08
TOPSIS	0	WPM	1	0.03	0.09	0.03	0.17	0.08
TOPSIS	0	TOPSIS	0	0.03	0.09	0.11	0.09	0.08
TOPSIS	0	TOPSIS	1	0.03	0.09	0.11	0.09	0.08
TOPSIS	1	WPM	0	0.03	0.09	0.03	0.17	0.08
TOPSIS	1	WPM	1	0.03	0.09	0.03	0.17	0.08
TOPSIS	1	TOPSIS	0	0.03	0.09	0.11	0.09	0.08
TOPSIS	1	TOPSIS	1	0.03	0.09	0.11	0.09	0.08
WSM	0	AHP	1	0.11	0.09	0.03	0.14	0.09
TOPSIS	0	AHP	1	0.11	0.09	0.03	0.14	0.09
TOPSIS	1	AHP	1	0.11	0.09	0.03	0.14	0.09
AHP	1	TOPSIS	0	0.14	0.06	0.09	0.11	0.10
AHP	1	TOPSIS	1	0.14	0.06	0.09	0.11	0.10
AHP	1	WSM	0	0.11	0.09	0.03	0.17	0.10
AHP	1	WSM	1	0.11	0.09	0.03	0.17	0.10
AHP	1	AHP	1	0.14	0.11	0.03	0.14	0.11

Table C.3.: Probability of recommending an irrelevant item considering the global score

Level 1		Skyline			Results					
MCDM	**Norm**	**Fuz**	**k**	**w**	**1**	**2**	**3**	**4**	**AVG**	**VAR**
AHP	1	10	3	-	0.457	0.561	0.721	0.433	0.543	0.017
AHP	1	0	2	0.9	0.457	0.519	0.629	0.510	0.529	0.005
TOPSIS	1	0	2	0.7	0.30	0.64	0.73	0.43	0.53	0.038
TOPSIS	1	4	3	-	0.571	0.561	0.720	0.207	0.515	0.047
AHP	1	7	3	-	0.298	0.561	0.721	0.433	0.503	0.033
WSM	0	0	2	0.7	0.14	0.64	0.73	0.43	0.49	0.068
TOPSIS	1	2	3	-	0.457	0.559	0.720	0.206	0.486	0.046
AHP	1	2	3	-	0.200	0.561	0.721	0.434	0.479	0.048
WSM	0	20	3	-	0.30	0.49	0.72	0.38	0.47	0.034
AHP	1	0	2	0.7	0.01	0.64	0.73	0.49	0.47	0.104
AHP	1	20	3	-	0.14	0.56	0.72	0.43	0.46	0.060
WSM	0	0	2	0.5	0.14	0.52	0.73	0.40	0.45	0.060
WSM	0	7	3	-	0.30	0.56	0.72	0.21	0.45	0.056
AHP	1	0	2	0.5	0.171	0.524	0.729	0.343	0.442	0.057
AHP	1	4	3	-	0.03	0.56	0.72	0.43	0.44	0.088
AHP	1	0	3	-	0.000	0.559	0.720	0.434	0.428	0.095
WSM	0	10	3	-	0.54	0.21	0.72	0.20	0.42	0.066
TOPSIS	1	50	3	-	0.227	0.491	0.720	0.236	0.419	0.055
WSM	0	0	2	0.3	0.298	0.238	0.729	0.400	0.416	0.048
TOPSIS	1	0	2	0.5	0.01	0.52	0.73	0.40	0.41	0.092
TOPSIS	1	7	3	-	0.143	0.561	0.721	0.207	0.408	0.077
WSM	0	4	3	-	0.143	0.561	0.721	0.206	0.408	0.077
TOPSIS	1	0	3	-	0.143	0.559	0.720	0.206	0.407	0.077
WSM	0	0	2	0.9	0.007	0.519	0.629	0.464	0.405	0.075
TOPSIS	1	0	2	0.9	0.007	0.519	0.629	0.464	0.405	0.075
TOPSIS	1	0	2	0.3	0.298	0.238	0.729	0.286	0.388	0.052
WSM	0	2	3	-	0.029	0.559	0.721	0.206	0.379	0.101
WSM	0	0	3	-	0.029	0.559	0.720	0.206	0.378	0.100
TOPSIS	1	10	3	-	0.007	0.561	0.722	0.204	0.374	0.107
WSM	0	0	2	0.1	0.143	0.210	0.729	0.400	0.370	0.069
AHP	1	0	2	0.3	0.01	0.24	0.73	0.49	0.36	0.097
AHP	1	0	2	0.1	0.000	0.210	0.729	0.486	0.356	0.101
TOPSIS	1	20	3	-	0.000	0.494	0.720	0.205	0.355	0.100
WSM	0	50	3	-	0.029	0.286	0.714	0.386	0.354	0.080
AHP	1	50	3	-	0.007	0.207	0.726	0.464	0.351	0.097
TOPSIS	1	0	2	0.1	0.007	0.210	0.729	0.257	0.301	0.093

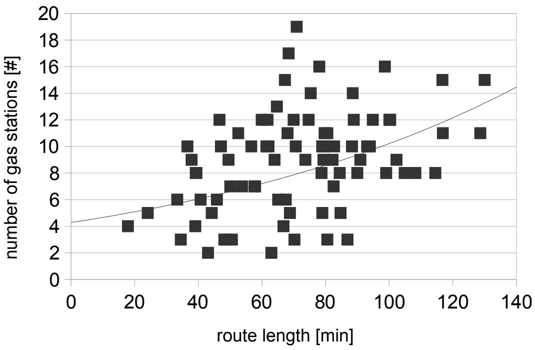
Table C.4.: Precision of Skyline sets

Level 1		Level 2		Results for POI sets					
MCDM	**Norm**	**MCDM**	**Norm**	**1**	**2**	**3**	**4**	**AVG**	**VAR**
TOPSIS	0	TOPSIS	1	0.89	0.97	1.00	0.94	0.95	0.002
TOPSIS	1	TOPSIS	1	0.89	0.97	1.00	0.94	0.95	0.002
WSM	0	WSM	1	0.86	0.97	0.94	1.00	0.94	0.004
TOPSIS	0	WSM	1	0.86	0.97	0.94	1.00	0.94	0.004
TOPSIS	1	WSM	1	0.86	0.97	0.94	1.00	0.94	0.004
WSM	0	TOPSIS	1	0.86	0.97	1.00	0.83	0.91	0.007
AHP	1	WSM	1	0.89	0.80	0.63	1.00	0.83	0.024
AHP	1	TOPSIS	1	0.89	0.80	0.71	0.91	0.83	0.008
WSM	0	AHP	1	0.89	0.74	0.60	0.91	0.79	0.021
AHP	1	WPM	1	0.77	0.94	0.66	0.77	0.79	0.014
TOPSIS	0	AHP	1	0.83	0.69	0.63	0.91	0.76	0.017
TOPSIS	1	AHP	1	0.83	0.69	0.63	0.91	0.76	0.017
TOPSIS	0	WSM	0	0.74	0.69	0.63	0.94	0.75	0.019
TOPSIS	1	WSM	0	0.74	0.69	0.63	0.94	0.75	0.019
WSM	0	WPM	1	0.71	0.89	0.77	0.54	0.73	0.020
AHP	1	AHP	1	0.66	0.63	0.63	0.91	0.71	0.019
TOPSIS	0	WPM	1	0.69	0.54	0.77	0.49	0.62	0.017
TOPSIS	1	WPM	1	0.69	0.54	0.77	0.49	0.62	0.017
WSM	0	TOPSIS	0	0.71	0.51	0.57	0.60	0.60	0.007
TOPSIS	0	TOPSIS	0	0.71	0.43	0.49	0.74	0.59	0.025
TOPSIS	1	TOPSIS	0	0.71	0.43	0.49	0.74	0.59	0.025
WSM	0	WSM	0	0.71	0.63	0.63	0.37	0.59	0.022
AHP	1	WSM	0	0.20	0.31	0.49	0.86	0.46	0.082
WSM	0	WPM	0	0.63	0.34	0.57	0.23	0.44	0.036
TOPSIS	0	WPM	0	0.63	0.17	0.57	0.29	0.41	0.049
TOPSIS	1	WPM	0	0.63	0.17	0.57	0.29	0.41	0.049
AHP	1	TOPSIS	0	0.09	0.17	0.20	0.54	0.25	0.040
AHP	1	WPM	0	0.03	0.00	0.31	0.31	0.16	0.030

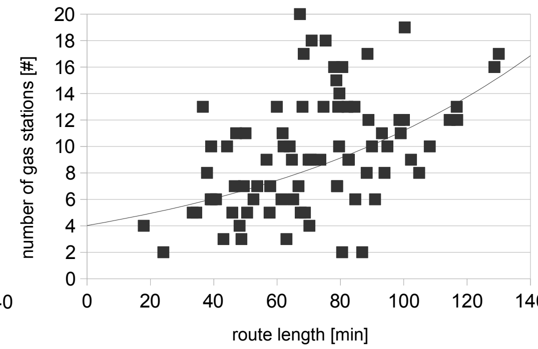
Table C.5.: Precision of a set that was selected with the relevance threshold 0.65 and the global score

Level 1		**Skyline**			**Results**					
MCDM	**Norm**	**Fuz**	**k**	**w**	**1**	**2**	**3**	**4**	**AVG**	**VAR**
WSM	0	0	2	0.9	0.829	0.971	0.686	0.943	0.857	0.017
AHP	1	0	2	0.9	0.829	0.971	0.686	0.943	0.857	0.017
TOPSIS	1	0	2	0.9	0.829	0.971	0.686	0.943	0.857	0.017
WSM	0	0	2	0.7	0.69	0.60	0.63	0.40	0.58	0.015
AHP	1	0	2	0.7	0.69	0.60	0.63	0.40	0.58	0.015
TOPSIS	1	0	2	0.7	0.69	0.60	0.63	0.40	0.58	0.015
AHP	1	0	3	-1	0.429	0.514	0.571	0.514	0.507	0.003
AHP	1	2	3	-1	0.429	0.429	0.571	0.514	0.486	0.005
AHP	1	4	3	-1	0.43	0.43	0.57	0.51	0.49	0.005
WSM	0	0	3	-1	0.429	0.514	0.571	0.371	0.471	0.008
TOPSIS	1	0	3	-1	0.429	0.514	0.571	0.371	0.471	0.008
AHP	1	7	3	-1	0.343	0.429	0.571	0.514	0.464	0.010
AHP	1	10	3	-1	0.343	0.429	0.571	0.514	0.464	0.010
AHP	1	20	3	-1	0.34	0.43	0.57	0.51	0.46	0.010
WSM	0	2	3	-1	0.343	0.514	0.571	0.371	0.450	0.012
TOPSIS	1	2	3	-1	0.343	0.514	0.571	0.371	0.450	0.012
AHP	1	50	3	-1	0.514	0.286	0.571	0.371	0.436	0.017
WSM	0	4	3	-1	0.343	0.429	0.571	0.371	0.429	0.010
WSM	0	7	3	-1	0.34	0.43	0.57	0.37	0.43	0.010
TOPSIS	1	4	3	-1	0.343	0.429	0.571	0.371	0.429	0.010
TOPSIS	1	7	3	-1	0.343	0.429	0.571	0.371	0.429	0.010
TOPSIS	1	10	3	-1	0.343	0.429	0.571	0.371	0.429	0.010
WSM	0	20	3	-1	0.43	0.29	0.57	0.26	0.39	0.021
TOPSIS	1	50	3	-1	0.429	0.286	0.571	0.229	0.379	0.024
WSM	0	10	3	-1	0.23	0.29	0.57	0.37	0.36	0.023
TOPSIS	1	20	3	-1	0.229	0.286	0.571	0.371	0.364	0.023
WSM	0	0	2	0.5	0.14	0.31	0.51	0.26	0.31	0.024
AHP	1	0	2	0.5	0.143	0.314	0.514	0.257	0.307	0.024
TOPSIS	1	0	2	0.5	0.14	0.31	0.51	0.26	0.31	0.024
AHP	1	0	2	0.1	0.143	0.171	0.514	0.371	0.300	0.031
AHP	1	0	2	0.3	0.14	0.17	0.51	0.37	0.30	0.031
WSM	0	50	3	-1	0.114	0.143	0.486	0.257	0.250	0.029
WSM	0	0	2	0.1	0.057	0.171	0.514	0.257	0.250	0.038
WSM	0	0	2	0.3	0.057	0.171	0.514	0.257	0.250	0.038
TOPSIS	1	0	2	0.1	0.057	0.171	0.514	0.229	0.243	0.038
TOPSIS	1	0	2	0.3	0.057	0.171	0.514	0.229	0.243	0.038

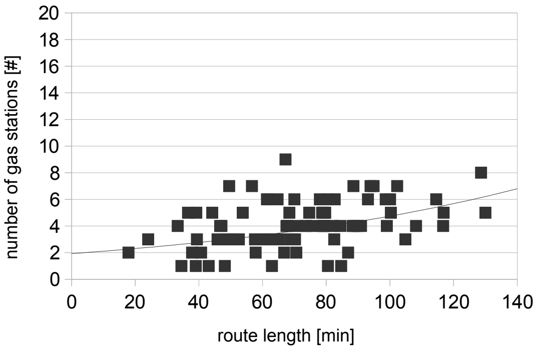
Table C.6.: Precision of a set that was selected by a Skyline



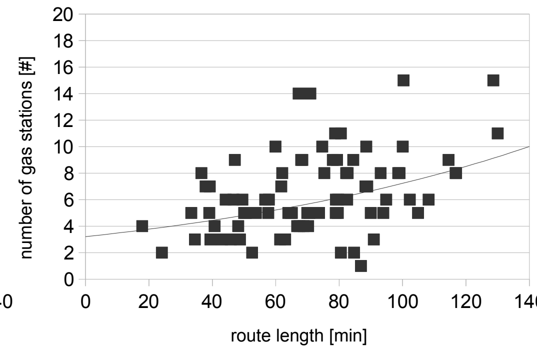
(a) Regular Skyline WSM



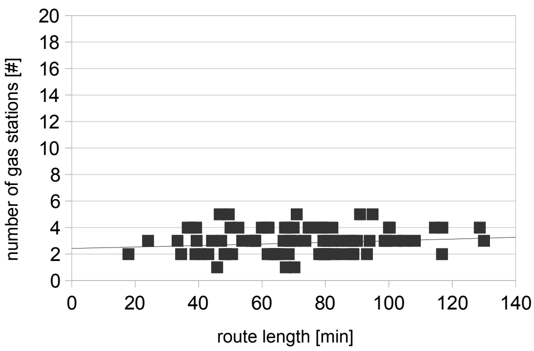
(b) Regular Sykline AHP



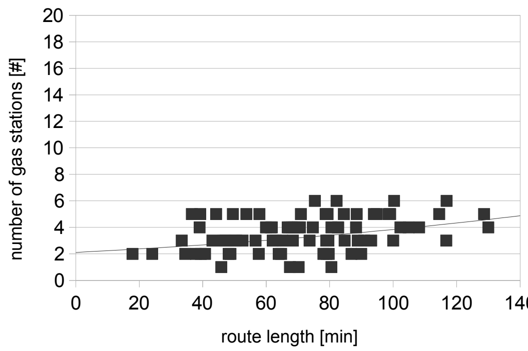
(c) Skyline fuzziness=10% WSM



(d) Skyline fuzziness=10% AHP



(e) Skyline $\omega = 0.7$ WSM



(f) Skyline $\omega = 0.7$ AHP

Figure C.2.: Comparison between the numbers of items relative to the length of the route with WSM and AHP on level 1 and different types of Skyline on level 2

D. Explanations

D.1. Results of the Preliminary Study

Arguments

The cells in the assessment tables refer to the assessments of the subjects. The distribution corresponds to the percentage of the subjects who gave the respective assessment on the Likert scale. The average value (AVG), the variance (VAR) and the median (MED) are listed to estimate the overall usefulness and how the subjects are in agreement. In a correlation table, Pearson's correlation between the assessments for the arguments and the explicit input of user preferences at the end of the survey are calculated.

This gas station was recommended ...	**1**	**2**	**3**	**4**	**5**	**AVG**	**VAR**	**MED**
... because it is cheap.	0.00%	0.00%	7.41%	30.86%	61.73 %	4.54	0.40	5
... because you get a free drink for every re-filling.	23.75%	33.75%	21.25%	15.00%	6.25%	2.46	1.42	2
... because it is the only brand station.	27.50%	35.00%	26.25%	10.00%	1.25%	2.23	1.01	2
... because only a few stations are reachable.	3.85%	8.97%	8.97%	37.18%	41.03%	4.03	1.22	4
... because it directly on the route.	1.28%	2.56%	3.85%	32.05%	60.26%	4.47	0.64	5
... because its the cheapest station.	1.25%	1.25%	6.25%	17.50%	73.75%	4.61	0.59	5
... because it is open at arrival.	1.23%	4.94%	4.94%	14.81%	74.07%	4.56	0.80	5
... because your gas level is nearly empty now.	1.25%	5.00%	8.75%	20.00%	65.00%	4.43	0.88	5
... because it is a brand station.	35.00%	36.25%	23.75%	5.00%	0.00%	1.99	0.80	2
... because there you do not have to reckon with waiting time.	7.50%	17.50%	30.00%	36.25%	8.75%	3.21	1.16	3

Table D.1.: Assessment Table: Useful and useless arguments

Argument - Preference	price	brand	detour	gas level
... because it is cheap.	0.523	-0.202	0.123	-0.204
... because you get a free drink for every refilling.	0.232	0.020	0.092	0.018
... because it is the only grand station.	-0.427	0.364	0.087	0.073
... because only a few stations are reachable.	-0.159	0.032	0.162	0.016
... because it directly on the route.	-0.014	0.030	0.347	-0.067
... because its the cheapest station.	0.461	-0.289	-0.151	-0.142
... because it is open at arrival.	0.124	0.076	0.117	-0.076
... because your gas level is nearly empty now.	-0.064	-0.097	0.109	0.312
... because it is a brand station.	-0.333	0.507	-0.047	0.075
... because there you do not have to reckon with waiting time.	-0.153	0.125	0.028	0.022

Table D.2.: Correlation table: User preferences and arguments

This gas station was recommended ...	1	2	3	4	5	AVG	VAR	MED
... because it is cheap. But you have to take a big detour.	9.88%	23.46%	37.04%	28.40%	1.23%	2.88	0.96	3
... because the gas level is very low now. But the gas price is very high.	3.70%	22.22%	29.63%	34.57%	9.88%	3.25	1.06	3
... because it directly on the route. But it is not a brand station.	2.47%	3.70%	6.17%	44.44%	43.21%	4.22	0.82	4
... because it is a brand station. But you have to take a big detour.	41.98%	44.44%	7.41%	6.17%	0.00%	1.78	0.70	2

Table D.3.: Assessment table: Useful and useless counterarguments

Argument - Preference	price	brand	detour	gas level
... because it is cheap. But you have to take a big detour.	0.479	-0.115	-0.058	-0.100
... because the gas level is very low now. But the gas price is very high.	-0.134	-0.162	0.254	0.213
... because it directly on the route. But it is not a brand station.	0.112	-0.056	0.345	-0.033
... because it is a brand station. But you have to take a big detour.	-0.036	0.309	0.161	0.063

Table D.4.: Correlation table: User preferences and counterarguments

Argument	1	2	3	4	5	AVG	VAR	MED
... because their is a cheaper gas stations in 40 km along the route.	2.47%	4.94%	17.28%	35.80%	39.51%	4.05	1.00	4
... because it closed at arrival.	0.00%	1.23%	1.23%	9.88%	87.65%	4.84	0.24	5
... because you have to take into account long waiting times at this time.	3.70%	14.81%	34.57%	27.16%	19.75%	3.44	1.18	3
... because it does not sell Super 98.	24.69%	14.81%	34.57%	13.58%	12.35%	2.74	1.72	3
... because you cannot reach it with your current gas level.	0.00%	2.47%	3.70%	7.41%	86.42%	4.78	0.40	5
Do you like explanations for gas stations not recommended provided on demand?	6.41%	7.69%	32.05%	35.90%	17.95%	3.51	1.16	4

Table D.5.: Assessment table: Usefulness of arguments for not recommended items

Explanation	1	2	3	4	5	AVG	VAR	MED
Since you are running out of gas, the system carries out a search for gas stations.	%	2.47%	12.35%	34.57%	45.68 %	4.14	1.12	4
The system checks 15.000 gas stations.	58.02 %	23.46%	11.11%	6.17%	1.23%	1.68	1.00	1
Downloading current gas prices.	22.22%	16.05%	23.46%	24.69%	13.58%	2.91	1.85	3
The system found some gas stations which are presented later.	29.63%	24.69%	16.05%	22.22%	7.41%	2.53	1.75	2
No recommendation available currently.	28.40%	13.58%	18.52%	30.86%	8.64%	2.77	1.96	3
Gas prices are up to date now.	22.22%	12.35%	25.93%	24.69%	14.81%	2.98	1.87	3
Since only a few gas stations are reachable anymore, the system carries out a search for gas stations.	8.64%	8.64%	18.52%	33.33%	30.86%	3.68	1.62	4

Table D.6.: Assessment table: Different kind of status information

Structure and Additional Information

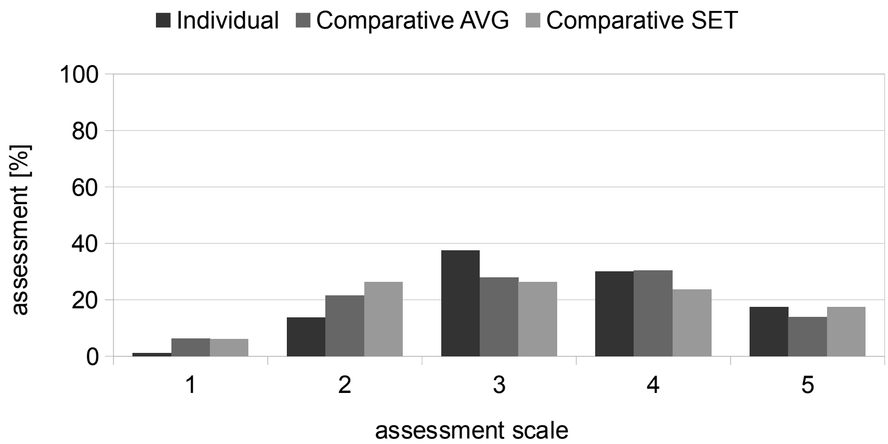


Figure D.1.: Structures of explanation sets: Individually explained items, comparative explained to an average (AVG) and comparative explained to the explanation set

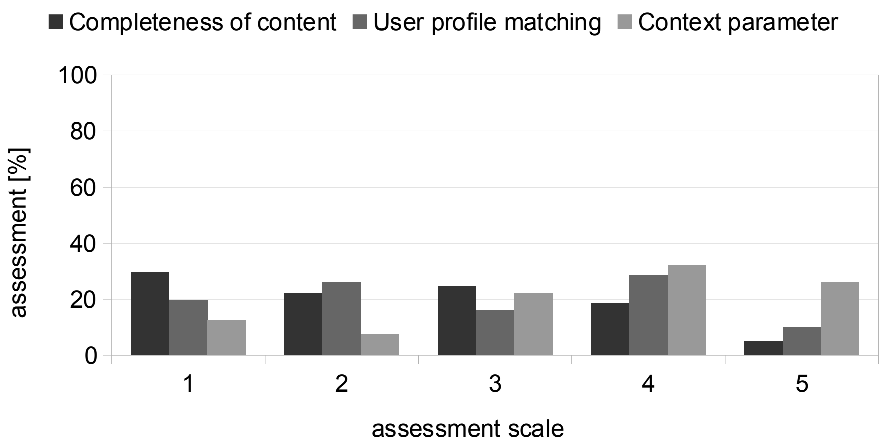


Figure D.2.: Reliability of different information sources: Completeness of available data, matching between user profile and recommendation, quality of context

Figure D.3.: Further categories of POIs that could be interesting for recommendations

Figure D.4.: Number of items for different traffic situations: From demanding to relaxed situations

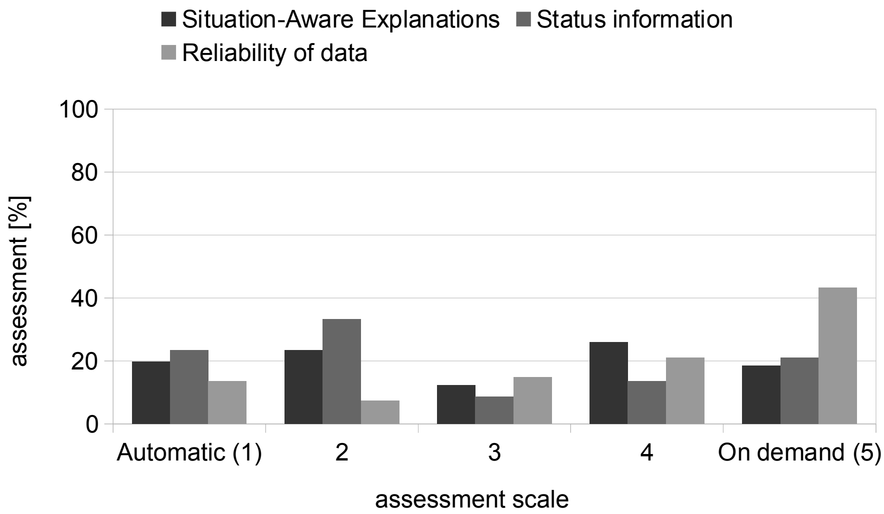


Figure D.5.: Level of proactivity for different kinds of additional explanations: From automatic to on demand delivery

Statements of the Participants

We also asked open questions. Overall, the statements in the open text section show a positive attitude towards the subject. In summary, 13 comments were made.

Concerning the questionnaire, one subject mentioned:

> "Especially on the first questions it was difficult to separate between displayed information and generated recommendations."

This is an issue, we also found in analyzing the data. People tended to rate the argument depending on the gas station they assumed in the recommendation.

One subject warns of information overflow:

> "Do not provide too much information. Information overload is annoying. Sometimes less is more."

We keep the warning in mind, since explanations are additional information that may help if relevant but can cause information overload if decisions are made more complicated as necessary. The reason we use recommender systems to deliver POIs in a car is exactly to reduce information overload.

Another subject likes the idea of making the car more intelligent:

> "[The gas station recommender] lets the car in eye of the drivers act a bit more intelligent and tailored to their needs."

Especially the use case of recommending gas stations seems to be plausible and easy to understand for driver. Further types of recommendations can be deployed later.

Finally, one subject made a comment about the reliability of data:

> "The customers anticipate that the data behind the service is available in acceptable quality."

The statement refers to the deployment of such a kind of service just in case the underlying data is of acceptable quality. Otherwise the service may become useless in the opinion of customers. Especially providers of premium quality products ensure a minimum of quality behind their used data.

D.2. Results for Item Explanations

Evaluation User Interface

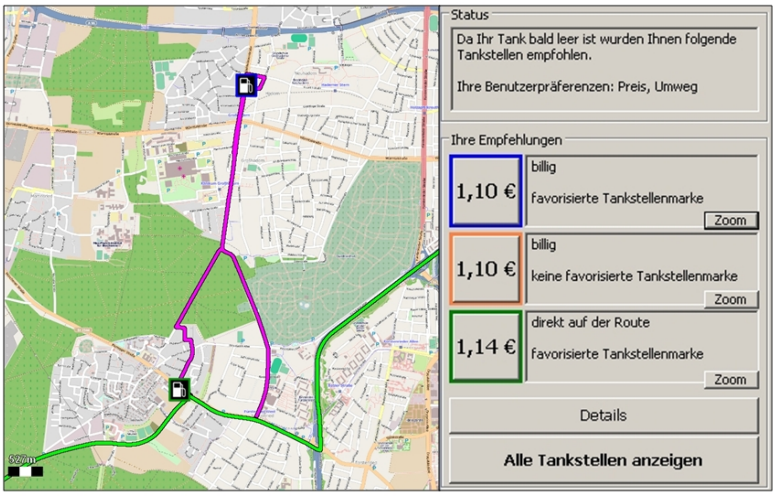


Figure D.6.: Level 1: Proactive View

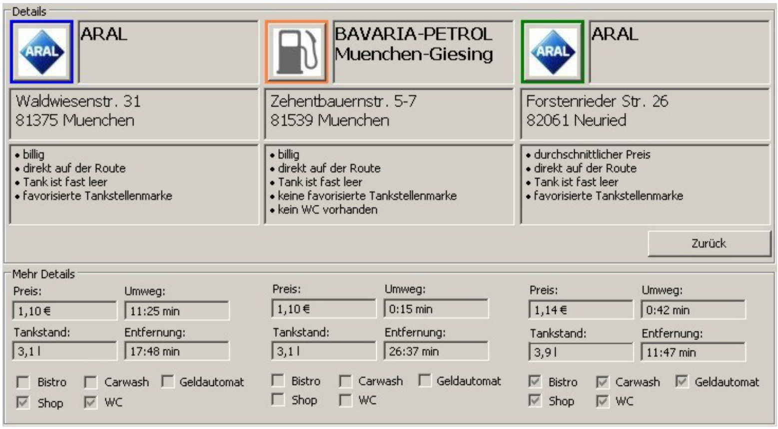
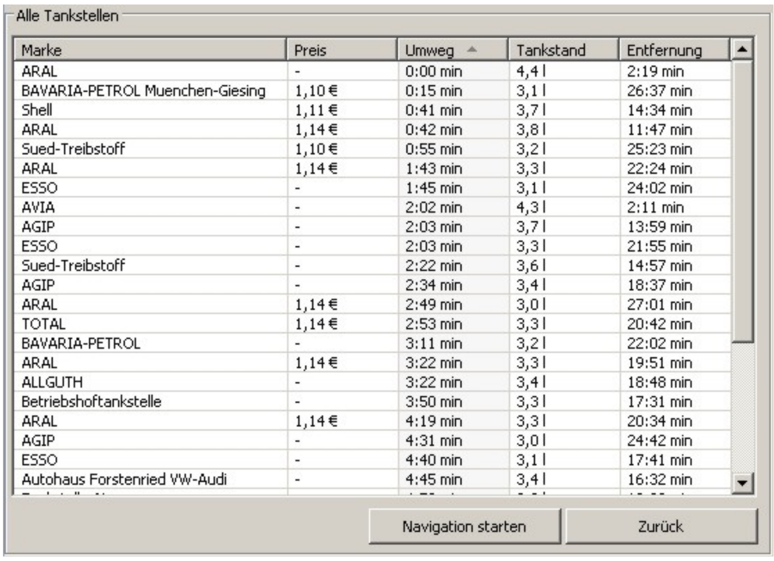


Figure D.7.: Level 2: Detail View



Alle Tankstellen

Marke	Preis	Umweg	Tankstand	Entfernung
ARAL	-	0:00 min	4,4 l	2:19 min
BAVARIA-PETROL Muenchen-Giesing	1,10 €	0:15 min	3,1 l	26:37 min
Shell	1,11 €	0:41 min	3,7 l	14:34 min
ARAL	1,14 €	0:42 min	3,8 l	11:47 min
Sued-Treibstoff	1,10 €	0:55 min	3,2 l	25:23 min
ARAL	1,14 €	1:43 min	3,3 l	22:24 min
ESSO	-	1:45 min	3,1 l	24:02 min
AVIA	-	2:02 min	4,3 l	2:11 min
AGIP	-	2:03 min	3,7 l	13:59 min
ESSO	-	2:03 min	3,3 l	21:55 min
Sued-Treibstoff	-	2:22 min	3,6 l	14:57 min
AGIP	-	2:34 min	3,4 l	18:37 min
ARAL	1,14 €	2:49 min	3,0 l	27:01 min
TOTAL	1,14 €	2:53 min	3,3 l	20:42 min
BAVARIA-PETROL	-	3:11 min	3,2 l	22:02 min
ARAL	1,14 €	3:22 min	3,3 l	19:51 min
ALLGUTH	-	3:22 min	3,4 l	18:48 min
Betriebshoftankstelle	-	3:50 min	3,3 l	17:31 min
ARAL	1,14 €	4:19 min	3,3 l	20:34 min
AGIP	-	4:31 min	3,0 l	24:42 min
ESSO	-	4:40 min	3,1 l	17:41 min
Autohaus Forstenried VW-Audi	-	4:45 min	3,4 l	16:32 min

Navigation starten

Zurück

Figure D.8.: Level 3: List View

User Preferences

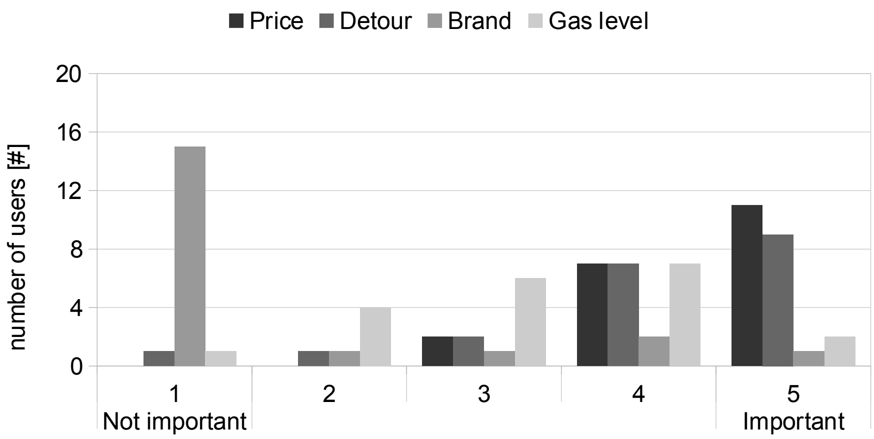
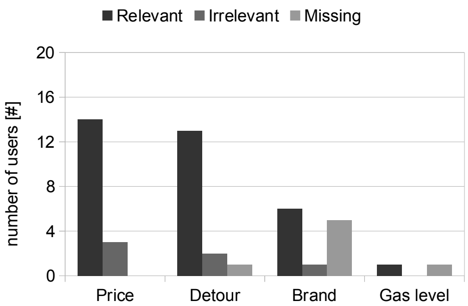


Figure D.9.: User preferences for gas station features

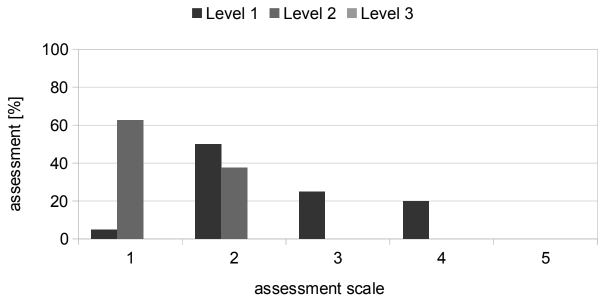
Scenarios

Scenario A: An Additional Need

In Scenario A, the subjects drive from home to work. They know that they need money for lunch and hence an ATM to withdraw some money. We wanted to see how this need that the system does not know and is not related to the actual task of refilling influences the assessment.



(a) Argument assessment

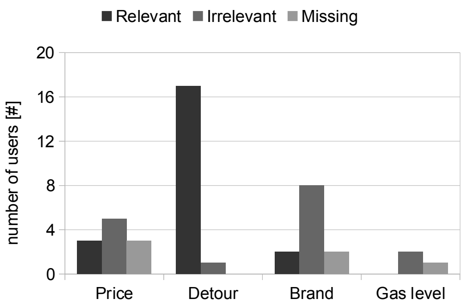


(b) Satisfaction

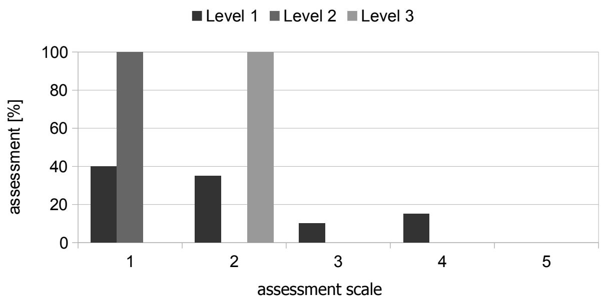
Figure D.10.: Results for Scenario A

Scenario B: An Urgent Appointment

Scenario B comprises additionally to the ride to work an urgent appointment at work. The system adjusts user preferences and recommends only POIs with low or no detour. We wanted to reassure our findings in the preliminary study that indicates that proactive adaptation of the user profile is regarded as useful.



(a) Argument assessment

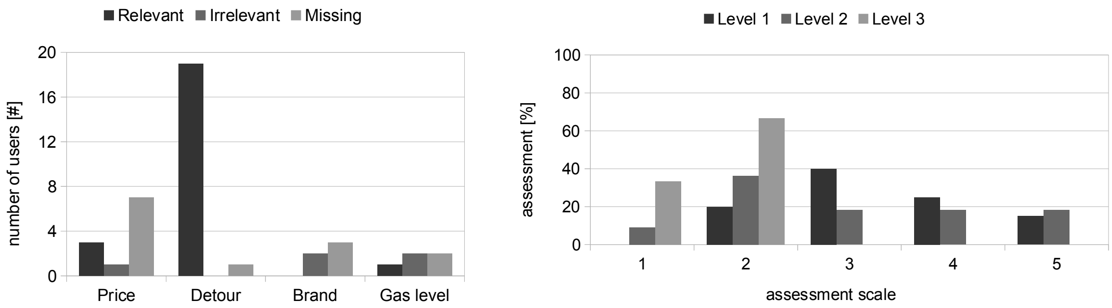


(b) Satisfaction

Figure D.11.: Results for Scenario B

Scenario C: A Regular Trip

This scenario represents a regular trip to friends. No further conditions are used here.

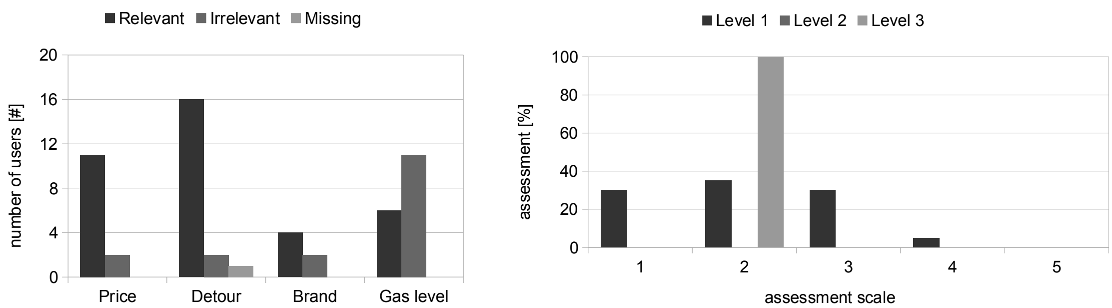


(a) Argument assessment (b) Satisfaction

Figure D.12.: Results for Scenario C

Scenario D: A Regular Trip (Without Explanations)

Scenario D also represents a regular trip. In this case, we replaced the explanations by a list of facts with all four decision dimensions for every item. We wanted to find out which information is regarded as useless.

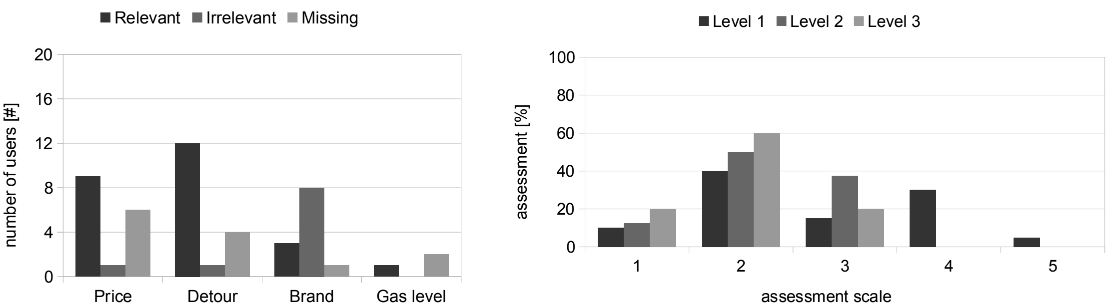


(a) Argument assessment (b) Satisfaction

Figure D.13.: Results for Scenario D

Scenario E: A Regular Trip with Inverted User Profile

Again, in this scenario a regular trip was used. In contrast to the other scenarios, we inverted the user profile, i.e., important dimensions become not important and vice versa. In the preliminary study, we have seen how important the user profile is for generating explanations. Here, we want to reassure what happens if a completely different profile is used.

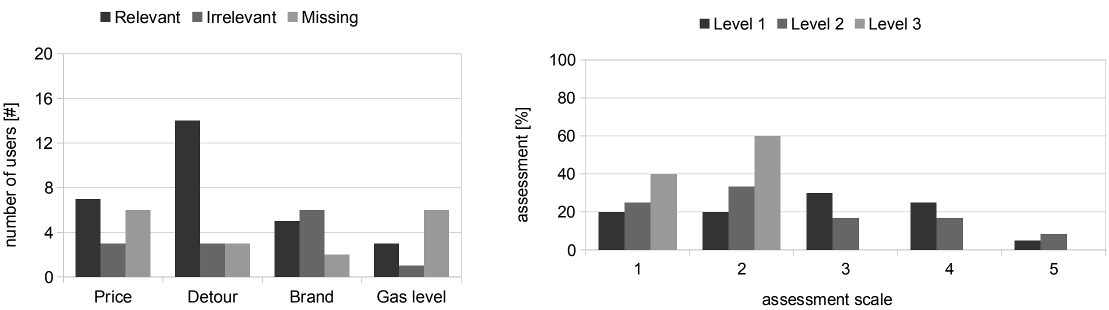


(a) Argument assessment (b) Satisfaction

Figure D.14.: Results for Scenario E

Scenario F: A Vacation Trip

Finally, the last scenario comprises a longer trip to a vacation destination. We wanted to find out whether there are additional decision criteria for non-regular trips.



(a) Argument assessment (b) Satisfaction

Figure D.15.: Results for Scenario F

Further Results

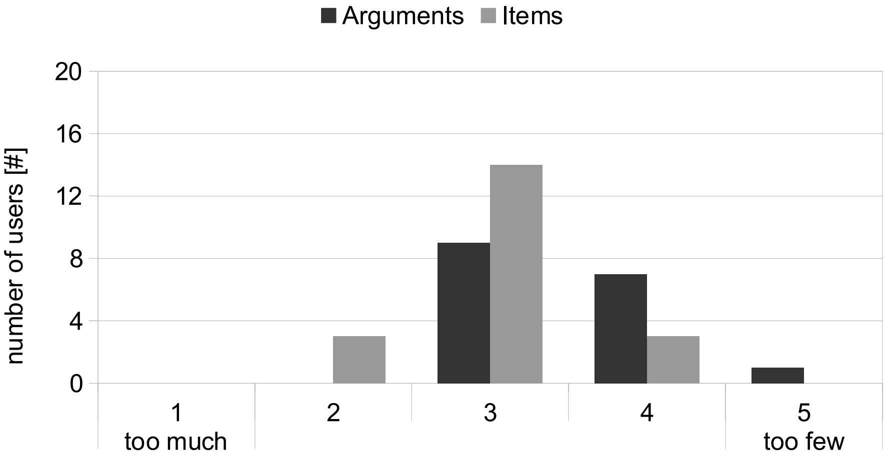


Figure D.16.: Perception of the number of arguments and items in the recommendation: From too much to too few

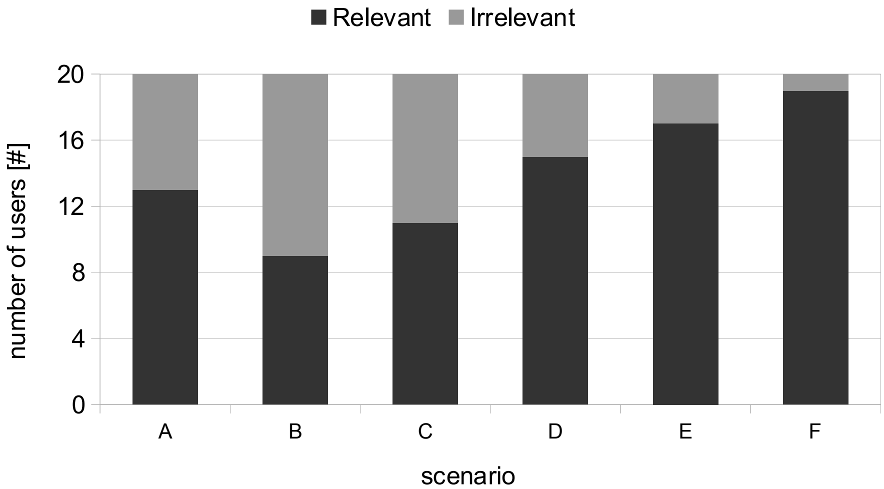


Figure D.17.: Relevance of the map view in each scenario

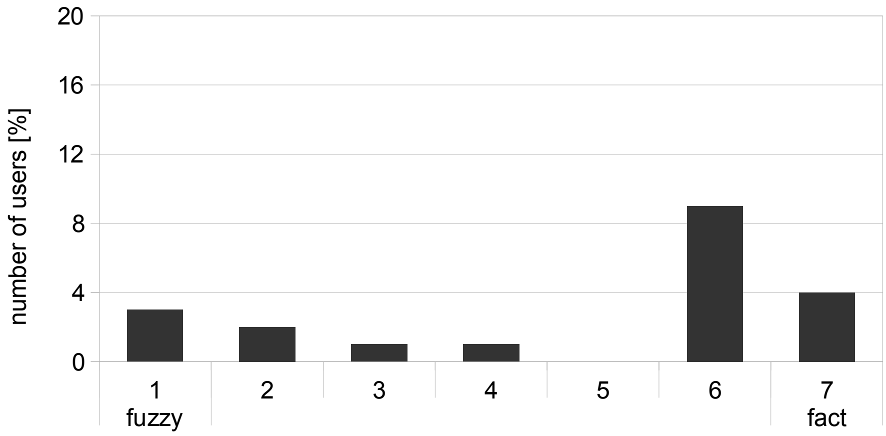


Figure D.18.: User assessments concerning fact or fuzzy explanations

D.3. Situation Explanations

Prefiltering of Arguments

Procedure 6 selectArgument

Input: $O :=$ Set of observations of lower-level situations
Output: $E :=$ explanation set containing observations as arguments

$O^* = sortByImportance(O)$
for all $o \in O^*$ **do**
  $i = impact(o)$
  if $i > \alpha$ **then**
    $E \leftarrow o$
    if $(D = P \wedge positive(o)) \vee (D = N \wedge !positive(o))$ **then**
      $atLeastOneSupport = true$
    end if
    if $(i > \beta) \wedge (atLeastOneSupport = true)$ **then**
      $break$
    end if
  end if
  if $number(o \in E) = n_{max}$ **then**
    $break$
  end if
end for
if $atLeastOneSupport = false$ **then**
  for all $o \in O^*$ **do**
    if $(D = P \wedge positive(o)) \vee (D = N \wedge !positive(o))$ **then**
      $E \leftarrow o$
      $atLeastOneSupport = true$
      $break$
    end if
  end for
end if
if $atLeastOneSupport = true$ **then**
  return E
else
  return $Error$
end if

Argument Ranking with AHP

In the second hierarchy level of the AHP the weights of each criterion are determined by a manual pairwise comparison by experts. The pairwise comparison matrix for the criteria is:

	IM	IP	OB	PK
IM	1.00	1.00	3.00	5.00
IP	1.00	1.00	3.00	5.00
OB	0.33	0.33	1.00	3.00
PK	0.20	0.20	0.33	1.00

Importance and impact are equally important and both are slightly more important than the obviousness of information. Furthermore, prior knowledge is assessed as much less important than importance and impact and less important than obviousness. With the standard method for Eigen vector calculation, we receive a normalized Eigen vector of $(0.39, 0.39, 0.15, 0.6)$. It corresponds to the weights that we apply for the criteria.

For pairwise comparison of the alternatives, we need a function $f(x)$ depending on the distribution of the values over the range of criterion c. Example functions can be seen in Figure D.19.

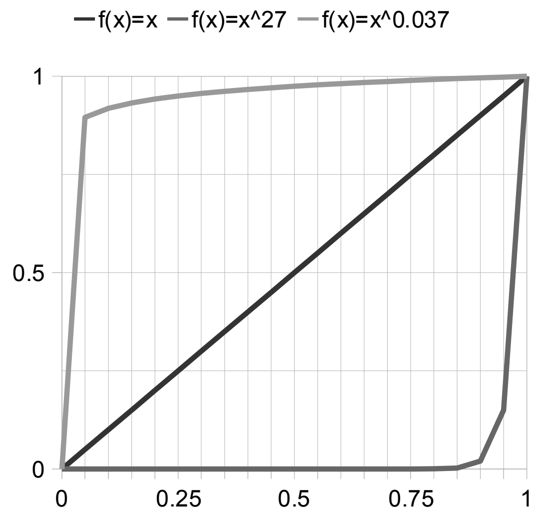


Figure D.19.: Pairwise comparison functions for equally usage of the range $f(x) = x$, mostly lower values usage $f(x) = x^{\frac{1}{27}}$ and higher values usage $f(x) = x^{27}$

If the range is equally used, the identity function $f(x) = x$ can be used. Otherwise, we increase the difference for lower values with a square root function $f(x) = \sqrt[n]{x}$ for the usage of lower values and vice versa with $f(x) = x^n$ for the usage of higher values. The value n is selected by means of the usually highest measured value in the range

x_m, e.g., the value with a confidence interval ci of 95%. For the lower case, it holds $\sqrt[n]{x} = ci \Leftrightarrow n = log_{x_m} ci$ because differences for $f(x) \in [0, ci]$ are increased and for $f(x) \in [ci, 1]$ are decreased. This is valid for lower values $x_m \in [0.0, ci]$. The opposite aspect applies to the usage of higher values. For instance, with $n = 27$, the highest expected value for lower values should be $x_m = 0.25$.

E. User Acceptance

Expectations

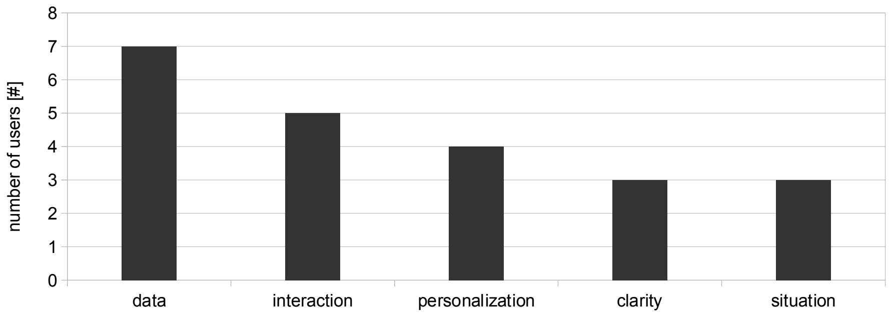


Figure E.1.: Expectations of users of an automatic recommendation system

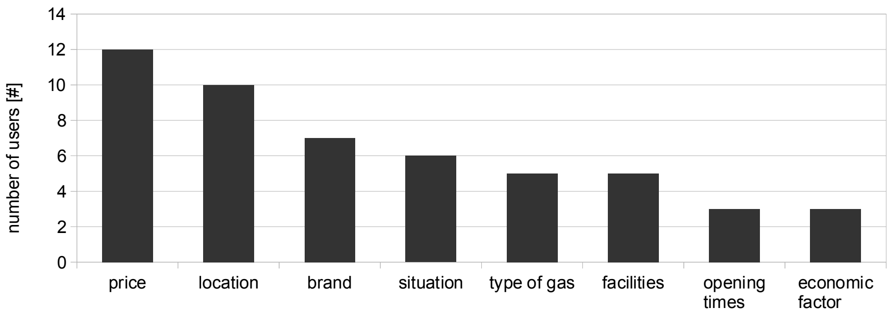


Figure E.2.: Expectations of users of an automatic gas station recommender

Scenarios

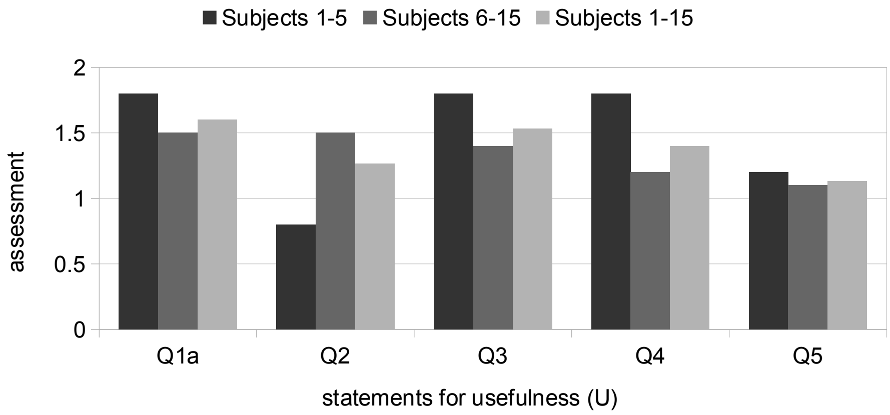


Figure E.3.: Assessments for Scenario 2

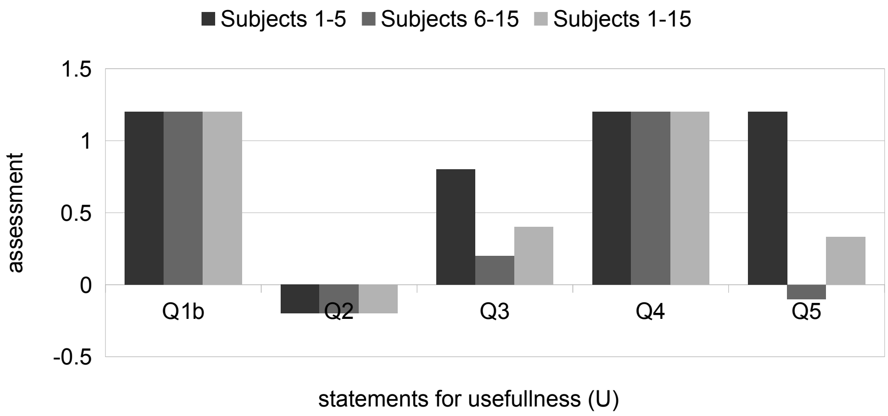


Figure E.4.: Assessments for Scenario 3

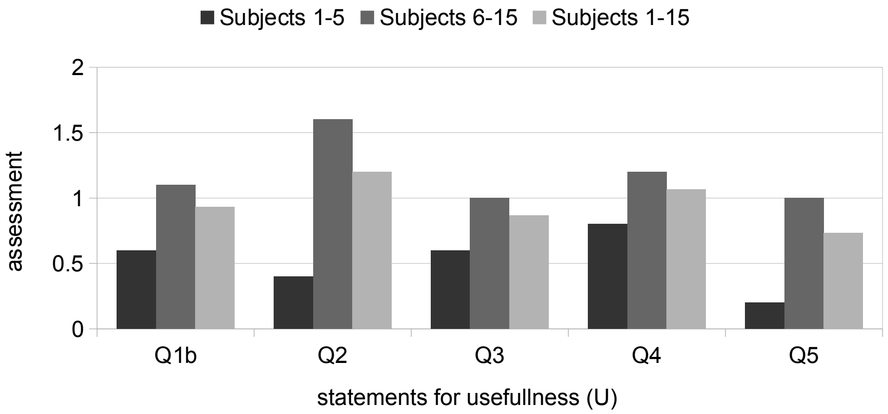


Figure E.5.: Assessments for Scenario 4

Amount of Recommended Items

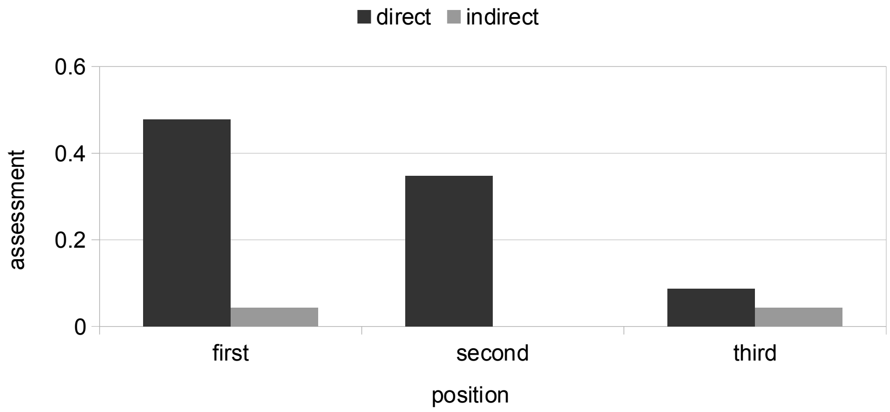


Figure E.6.: Position of items in the ranking in recommendation sets with more than 1 item

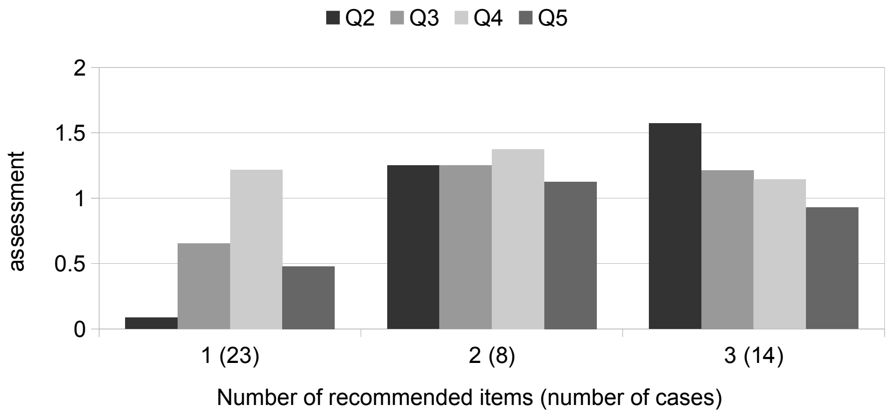


Figure E.7.: Assessments relative to the amount of recommended items

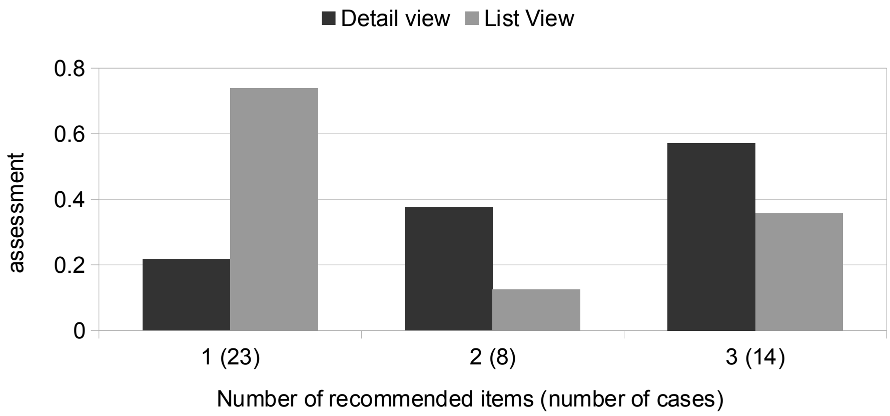


Figure E.8.: Interaction behavior (detail and list view invocation) relative to the amount of recommended items

Wording

(a) Both fuzzy

(b) Price fuzzy, detour crisp

(c) Price crisp, detour fuzzy

(d) Both crisp

Figure E.9.: Different variants of wordings (fuzzy, crisp and mixed)

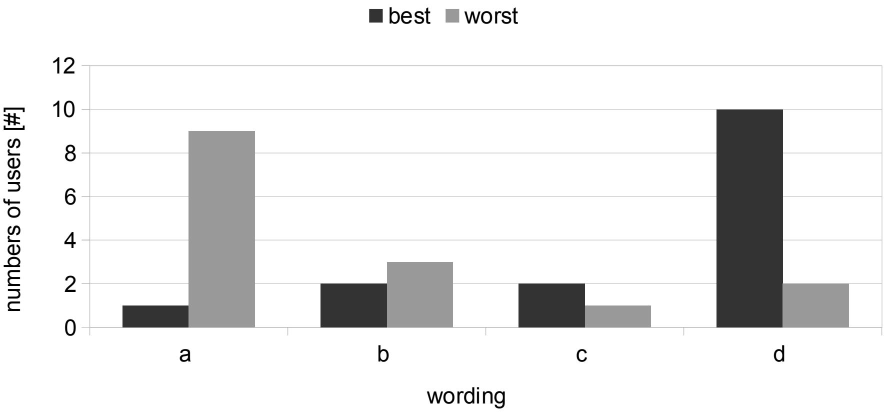


Figure E.10.: Assessments for different wording variants

List of Abbreviations

A	Attitude
ACC	Adaptive cruise control
ADAS	Advanced driver assistance systems
ADDIF	Additive difference
AHP	Analytic hierarchy process
AI	Artificial intelligence
AUC	Area under the ROC curve
BDI	Belief desire intention
BI	Behavioral intention
C2X	Car2X
CARS	Context-aware recommender systems
CF	Collaborative filtering
CHI	Chi square
CID	Central information display
COG	Center-of-gravity
CPT	Conditional probability table
CSP	Constraint satisfaction problem
DAG	Directed acyclic graph
EBA	Elimination-by-aspect
EH	Electronic horizon

EM	Expectation maximization
EOU	Ease of use
EQW	Equal weight
ESP	Electronic stability program
FRQ	Frequency of good and bad features
GNSS	Global navigation satellite system
GPS	Geographical positioning system
GR	Gain ratio
HMI	Human machine interface
IP	Interface-proactivity
IR	Information retrieval
IRNN	In-route nearest neighbor
IVIS	In-vehicle information systems
IVRS	In-vehicle recommender system
JITIR	Just-in-time information retrieval
LBS	Location-based services
LEX	Lexicographic
MAE	Mean absolute error
MANET	Mobile ad-hoc networks
MAUT	Multi-attribute utility theory
MCD	Majority of confirming dimensions
MCDM	Multi-criteria decision making
MI	Mutual information
ML	Maximum likelihood

MPE	Most probable explanation
MRE	Mean recommendation error
MSE	Mean square error
OS	Optimizing-satisficing
P-IVRS	Proactive in-vehicle recommender systems
POI	Point-of-interest
POMDP	Partially observable Markov decision process
PRS	Proactive recommender systems
RBF	Radial basis networks
RFF	ReliefF
ROC	Receiver operating characteristic
RS	Recommender systems
SAT	Satisficing
SVM	Support vector machine
SWRL	Semantic web rule language
TAM	Technology acceptance model
TMC	Traffic message channel
TOPSIS	Technique for order preference by similarity to ideal solution
U	Usefulness
WADD	Weighted additive
WPM	Weighted product model
WSM	Weighted sum model

Bibliography

[AAH+97] Gregory D Abowd, Christopher G Atkeson, Jason Hong, Sue Long, Rob Kooper, and Mike Pinkerton. Cyberguide: A mobile context-aware tour guide. *Wireless Networks*, 3(5):421–433, 1997.

[AAM01] Stefano Aguzzoli, Paolo Avesani, and Paolo Massa. Compositional Recommender Systems Using Case-Based Reasoning Approach. In *Workshop on Recommender Systems*, New Orleans, LA, USA, 2001.

[AB09] Bettina Abendroth and Ralph Bruder. Die Leistungsfähigkeit des Menschen für die Fahrzeugführung. In *Handbuch Fahrerassistenzsysteme: Grundlagen, Komponenten und Systeme für aktive Sicherheit und Komfort*, chapter 1, pages 4–14. Vieweg+Teubner Verlag, 2009.

[Abe01] Bettina Abendroth. *Gestaltungspotentiale für ein PKW-Abstandsregelsystem unter Berücksichtigung verschiedener Fahrertypen.* Ergonomia, Stuttgart, 1 edition, 2001.

[Abl09] Markus Ablassmeier. *Multimodales , kontextadaptives Informationsmanagement im Automobil.* Phd thesis, Technische Universität München, 2009.

[ACK94] Abhaya Asthana, Mark Crauatts, and Paul Krzyzanowski. An Indoor Wireless System for Personalized Shopping Assistance. In *Workshop on Mobile Computing Systems and Applications*, pages 69–74, Santa Cruz, CA, USA, 1994.

[ADA12] ADAC. Mehr Transparenz auf dem Kraftstoffmarkt. Technical report, 2012.

[AF94] J Allen and G Ferguson. Actions and Events in Interval Temporal Logic. *Journal of Logic and Computation*, 4(5):531–579, 1994.

[AGKO04] Lauri Aalto, Nicklas Göthlin, Jani Korhonen, and Timo Ojala. Bluetooth and WAP push based location-aware mobile advertising system. In *Conference on Mobile Systems, Applications, and Services*, pages 49–58, New York, NY, USA, 2004.

[AKH06] Hynuchul Ahn, Kyoung-jae Kim, and Ingoo Han. Mobile Advertisement Recommender System using Collaborative Filtering: MAR-CF. In *The Korea Society of Management Information Systems*, pages 709–715, 2006.

[AKS+10] Florian Alt, Dagmar Kern, Fabian Schulte, Bastian Pfleging, Alireza Sahami Shirazi, and Albrecht Schmidt. Enabling Micro-Entertainment in Vehicles Based on Context Information. In *Conference on AutomotiveUI*, pages 117–124, Pittsburgh, KS, USA, 2010.

[ALWM10] Hidir Aras, Han Li, Zicheng Wang, and Rainer Malaka. Proactive Recommendation of User-generated Mobile Video Messages. In *Workshop on Social Mobile Web*, Lisbon, Portugal, 2010.

[AM87] Joseph W. Alba and Howard Marmorstein. The Effects of Frequency Knowledge on Consumer Decision Making. *Journal of Consumer Research*, 14(1):14–25, 1987.

[AM07] Sarabjot Singh Anand and Bamshad Mobasher. Contextual Recommendation. In *Web to Social Web: Discovering and Deploying User and Content Profiles*, pages 142–160. Springer, 2007.

[AMK11] Gediminas Adomavicius, Nikos Manouselis, and Youngok Kwon. Multi-Criteria Recommender Systems. In *Recommender Systems Handbook: A Complete Guide for Research Scientists and Practitioners*, chapter 24, pages 769–798. Springer, 2011.

[AMRT12] Gediminas Adomavicius, Bamschad Mobasher, Francesco Ricci, and Alexander Tuzhilin. Context-aware recommender systems. *Artificial Intelligence Magazine*, 32(3):67–80, 2012.

[ANH07] Christos Anagnostopoulos, Yiorgos Ntarladimas, and Stathes Hadjiefthymiades. Situational computing: An innovative architecture with imprecise reasoning. *Journal of Systems and Software*, 80(12):1993–2014, 2007.

[ANY+12] Kengo Akaho, Takashi Nakagawa, Yoshihisa Yamaguchi, Katsuya Kawai, Hirokazu Kato, and Shogo Nishida. Route guidance by a car navigation system based on augmented reality. *Electrical Engineering in Japan*, 180(2):43–54, 2012.

[AP94] Agnar Aamodt and Eric Plaza. Case-Based Reasoning: Foundational Issues, Methodological Variations, and System Approaches. *Artificial Intelligence Communications*, 7(1):39–59, 1994.

[AP09] Leila Amgoud and Henri Prade. Using arguments for making and explaining decisions. *Artificial Intelligence*, 173(3-4):413–436, 2009.

[App11] AppoftheDay. Apples App Store Statistics For 2010, 2011.

[APRR07] Markus Ablassmeier, Tony Poitschke, Stefan Reifinger, and Gerhard Rigoll. Context-Aware Information Agents for the Automotive Domain Using Bayesian Networks. In *Symposium on Human Computer Interface*, pages 561–570, Beijing, China, 2007.

[APTE05] Miguel B Araujo, Richard G Pearson, Wilfried Thuiller, and Markus Erhard. Validation of species-climate impact models under climate change. *Global Change Biology*, 11(9):1504–1513, 2005.

[ARN08] Olga Averjanova, Francesco Ricci, and Quang Nhat Nguyen. Map-Based Interaction with a Conversational Mobile Recommender System. In *Conference on Mobile Ubiquitous Computing, Systems, Services and Technologies*, pages 212–218, Valencia, Spain, 2008.

[ART06] Rakesh Agrawal, Ralf Rantzau, and Evimaria Terzi. Context-sensitive ranking. In *Conference on Management of data*, pages 383–394, Chicago, IL, USA, 2006.

[AS02] Daniel Ashbrook and Thad Starner. Learning significant locations and predicting user movement with GPS. In *Symposium on Wearable Computers*, pages 101–108, White Plains, NY, USA, 2002.

[ASST05] Gediminas Adomavicius, Ramesh Sankaranarayanan, Shahana Sen, and Alexander Tuzhilin. Incorporating contextual information in recommender systems using a multidimensional approach. *ACM Transactions on Information Systems*, 23(1):103–145, 2005.

[AT05a] Gediminas Adomavicius and Alexander Tuzhilin. Incorporating Context into Recommender Systems Using Multidimensional Rating Estimation Methods. *ACM Transactions on Information Systems*, 23(1):103 – 145, 2005.

[AT05b] Gediminas Adomavicius and Alexander Tuzhilin. Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions. *IEEE Transactions on Knowledge and Data Engineering*, 17(6):734–749, 2005.

[AT08] Gediminas Adomavicius and Alexander Tuzhilin. Context-aware recommender systems. In *Conference on Recommender Systems*, pages 335–336, New York, NY, USA, 2008.

[AT10] Gediminas Adomavicius and Alexander Tuzhilin. Context-Aware Recommender Systems. In *Recommender Systems Handbook: A Complete Guide for Research Scientists and Practitioners*, chapter 7, pages 217–250. Springer, 2010.

[BA09] Linas Baltrunas and Xavier Amatriain. Towards Time-Dependant Recommendation based on Implicit Feedback. In *Workshop on Context-Aware Recommender Systems*, pages 423–424, New York, NY, USA, 2009.

[Bar81] Jon Barwise. Scenes and other situations. *The Journal of Philosophy*, 78(7):369–97, 1981.

[Bar86] Robert S. Baron. Distraction-Conflict Theory: Progress and Problems. *Advances in Experimental Social Psychology*, 19:1–40, 1986.

[BB03] Mauro Brunato and Roberto Battiti. PILGRIM: A location broker and mobility-aware recommendation system. In *Conference on Pervasive Computing and Communications*, pages 265–272, Fort Worth, TX, USA, 2003.

[BB05] Mary Bazire and Patrick Brézillon. Understanding Context Before Using It. In *Conference on Modeling and Using Context*, pages 29 – 40, Paris, France, 2005.

[BBC97] Peter J Brown, John D Bovey, and Xian Chen. Context-aware applications: from the laboratory to the marketplace. *IEEE Personal Communications*, 4(5):58–64, 1997.

[BBE+02] Daniel Billsus, Clifford A Brunk, Craig Evans, Brian Gladish, and Michael Pazzani. Adaptive interfaces for ubiquitous web access. *Communications of the ACM*, 45(5), 2002.

[BBH+07] Katharina Bachfischer, Thorsten Bohnenberger, Martin Hofmann, Christoph Wäller, and Yongmei Wu. Kontext-adaptive Fahrerinformationssysteme am Beispiel eines Navigationssystems. *Künstliche Intelligenz*, 21(3):57–63, 2007.

[BBH+10] Claudio Bettini, Oliver Brdiczka, Karen Henricksen, Jadwiga Indulska, Daniela Nicklas, Anand Ranganathan, and Daniele Riboni. A survey of context modelling and reasoning techniques. *Pervasive and Mobile Computing*, 6(2):161–180, 2010.

[BC92] Nicholas J Belkin and W Bruce Croft. Information filtering and information retrieval: Two sides of the same coin? *Communications of the ACM*, 29(10):1–10, 1992.

[BCB+05] Barry Brown, Matthew Chalmers, Marek Bell, Malcolm Hall, Paul Rudman, and Ian MacColl. Sharing the square: Collaborative Leisure in the City Streets. In *Conference on Computer Supported Cooperative Work*, pages 427 – 447, Paris, France, 2005.

[BDR07] Matthias Baldauf, Schahram Dustdar, and Florian Rosenberg. A survey on context-aware systems. *International Journal of Ad Hoc and Ubiquitous Computing*, 2(4):263–277, 2007.

[BDSK05] Rebecca Bulander, Michael Decker, Gunther Schiefer, and Bernhard Kölmel. Comparison of Different Approaches for Mobile Advertising. In *Workshop on Mobile Commerce and Services*, pages 174–182, Munich, Germany, 2005.

[BDW09] Amel Bouzeghoub, Kien Ngoc Do, and Leandro Krug Wives. Situation-Aware Adaptive Recommendation to Assist Mobile Users in a Campus Environment. In *Conference on Advanced Information Networking and Applications*, pages 503–509, Bradford, UK, 2009.

[Bei00] Michael Beigl. MemoClip: A location-based remembrance agent. *Personal and Ubiquitous Computing*, 4(4):230–233, 2000.

[Bel00] Nicolas J Belkin. Helping people find what they don't know. *Communications of the ACM*, 43(8):58–61, 2000.

[BFH+07] Thomas Beer, Matthias Fuchs, Wolfram Höpken, Jörg Rasinger, and Hannes Werthner. CAIPS: A Context-Aware Information Push Service in Tourism. In *Conference on Information and Communication Technologies in Tourism*, pages 129 – 140, Ljubljana, Slovenia, 2007.

[BH99] Jay Budzik and Kristian J Hammond. Watson: Anticipating and Contextualizing Information Needs. In *Annual Meeting of the American Society for Information Science*, pages 727–740, Washington, VA, USA, 1999.

[BHBK00] Jay Budzik, Kristian J Hammond, Larry Birnbaum, and Marko Krema. Beyond Similarity. In *Workshop on AI for Web Search*, Austin, TX, USA, 2000.

[BHK98] John S Breese, David Heckerman, and Carl Kadie. Empirical Analysis of Predictive Algorithms for Collaborative Filtering. In *Conference on Uncertainty in artificial intelligence*, pages 43–52, Madison, WI, USA, 1998.

[BHMA05] Daniel Billsus, David M. Hilbert, and Dan Maynes-Aminzade. Improving proactive information systems. In *Conference on Intelligent User Interfaces*, pages 159–166, New York, NY, USA, 2005.

[BHZR06] Thomas Beer, Wolfram Höpken, Markus Zanker, and Jörg Rasinger. An Intelligent Information Push Service for the Tourist Domain. In *Workshop on Recommender Systems*, pages 125–127, Riva del Garda, Italy, 2006.

[Bil95] Charles E Billings. Situation awareness measurement and analysis: A commentary. In *Conference on Experimental Analysis and Measurement of Situation Awareness*, Daytona Beach, FL, USA, 1995.

[BJC+07] Dick C A Bulterman, A J Jansen, Pablo Cesar, Hendrik Knoche, and William Seager. Enabling Pro-Active User-Centered Recommender Systems: An Initial Evaluation. In *Symposium on Multimedia Workshops*, pages 195–200, Taichung, Taiwan, 2007.

[BJJ05] Thorsten Bohnenberger, Oliver Jacobs, and Anthony Jameson. Decision-Theoretic Planning Meets User Requirements : Enhancements and Studies of an Intelligent Shopping Guide. In *Conference on Pervasive Computing*, pages 279–296, Munich, Germany, 2005.

[BKS01] Stephan Börzsönyi, Donald Kossmann, and Konrad Stocker. The Skyline Operator. In *Conference on Data Engineering*, pages 421 – 430, Heidelberg, Germany, 2001.

[BKWL11] Roland Bader, Andreas Karitnig, Wolfgang Wörndl, and Gerhard Leitner. Explanations in Proactive Recommender Systems in Automotive Scenarios. In *Workshop on Decision Making and Recommendation Acceptance Issues in Recommender Systems*, Girona, Spain, 2011.

[BL07] James Bennett and Stan Lanning. The Netflix Prize. In *KDD Cup and Workshop*, San Jose, CA, USA, 2007.

[BLPR11] Linas Baltrunas, Bernd Ludwig, Stefan Peer, and Francesco Ricci. Context-Aware Places of Interest Recommendations and Explanations. In *Workshop on Decision Making and Recommendation Acceptance Issues in Recommender Systems*, pages 531–540, Girona, Spain, 2011.

[BMW11] BMW. Innovationstage Connected Drive, 2011.

[BNWP11] Roland Bader, Eugen Neufeld, Wolfgang Wörndl, and Vivian Prinz. Context-aware POI recommendations in an automotive scenario using multi-criteria decision making methods. In *Workshop on Context-awareness in Retrieval and Recommendation*, pages 23–30, Palo Alto, CA, 2011.

[BPH+05] Jennifer Boger, Pascal Poupart, Jesse Hoey, Craig Boutilier, Geoff Fernie, and Alex Mihailidis. A Decision-Theoretic Approach to Task Assistance for Persons with Dementia. In *Conference on Artificial intelligence*, pages 1293–1299, Edinburgh, Scotland, UK, 2005.

[BPR+08] Victoria Bellotti, Bob Price, Paul Rasmussen, Michael Roberts, Diane J Schiano, Alan Walendowski, Bo Begole, Ed H Chi, Nicolas Ducheneaut, Ji Fang, Ellen Isaacs, Tracy King, Mark W Newman, and Kurt Partridge. Activity-based serendipitous recommendations with the Magitti mobile leisure guide. In *Conference on Human factors in computing systems*, pages 1157–1166, New York, NY, USA, 2008.

[BR09] Linas Baltrunas and Francesco Ricci. Context-Dependent Items Generation in Collaborative Filtering. In *Workshop on Context-aware Recommender Systems*, New York, NY, USA, 2009.

[Bre09] Jörg Breuer. Bewertungsverfahren von Fahrerassistenzsystemen. In *Handbuch Fahrerassistenzsysteme: Grundlagen, Komponenten und Systeme für aktive Sicherheit und Komfort*, chapter 6, pages 55–68. Vieweg+Teubner Verlag, 2009.

[BRHF07] Thomas Beer, Jörg Rasinger, Wolfram Höpken, and Matthias Fuchs. Exploiting E-C-A Rules for Defining and Processing Context-Aware Push Messages. In *Conference on Advances in Rule Interchange and Applications*, pages 199–206, Santorini Island, Greece, 2007.

[Bro01] John Seely Brown. Where Have All the Computers Gone? *MIT Technology Review*, 104(1):86–87, 2001.

[BS84] Bruce G Buchanan and Edward H. Shortliffe. *Rule Based Expert Systems: The Mycin Experiments of the Stanford Heuristic Programming Project.* Addison-Wesley, 1984.

[BS97] Marko Balabanović and Yoav Shoham. Fab: content-based, collaborative recommendation. *Communications of the ACM*, 40(3):66–72, 1997.

[BSW11] Roland Bader, Oliver Siegmund, and Wolfgang Wörndl. A Study on User Acceptance of Proactive In-Vehicle Recommender Systems. In *Conference on AutomotiveUI*, pages 47–54, Salzburg, Austria, 2011.

[BSZ09] Fabian Bohnert, Daniel F Schmidt, and Ingrid Zukerman. Spatial Processes for Recommender Systems. In *Conference on Artificial Intelligence*, pages 2022–2027, Pasadena, CA, USA, 2009.

[Bub03] Heiner Bubb. Fahrerassistenz primär ein Beitrag zum Komfort oder für die Sicherheit? In *Der Fahrer im 21. Jahrhundert : (Anforderungen, Anwendungen, Aspekte fur Mensch-Maschine-Systeme)*, pages 25–44, Braunschweig, Germany, 2003. TUM.

[Bur00] Robin D Burke. Knowledge-based recommender systems. *Encyclopedia of Library and Information Systems*, 69(32), 2000.

[Bur02] Robin D Burke. Hybrid Recommender Systems: Survey and Experiments. *User Modeling and User-Adapted Interaction*, 12(4):331–370, 2002.

[BWP10] Roland Bader, Wolfgang Wörndl, and Vivian Prinz. Situation Awareness for Proactive In-Car Recommendations of Points-Of-Interest (POI). In *Workshop on Context Aware Intelligent Assistance*, Karlsruhe, Germany, 2010.

[CA11] Mark Chan and Paul Atchley. Potential benefits of a concurrent verbal task when feeling fatigued due to monotonous driving conditions. In *Symposium on Human Factors in Driver Assessment, Training and Vehicle Design*, pages 151–157, Lake Tahoe, CA, USA, 2011.

[CBW01] Dean P Chiang, Aaron M Brooks, and David H Weir. An Experimental Study of Destination Entry with an Example Automobile Navigation System. *Intelligent Vehicle Initiative (IVI): Technology and Navigation Systems*, pages 113–123, 2001.

[CCG+06] Federica Cena, Luca Console, Cristina Gena, Anna Goy, Guido Levi, Sonia Modeo, and Ilaria Torre. Integrating heterogeneous adaptation techniques to build a flexible and usable mobile tourist guide. *AI Communications*, 19(4):369 – 384, 2006.

[CCLM09] Alessandro Ciaramella, Mario G C A Cimino, Beatrice Lazzerini, and Francesco Marcelloni. Situation-Aware Mobile Service Recommendation with Fuzzy Logic and Semantic Web. In *Conference on Intelligent Systems Design and Applications*, pages 1037–1042, Pisa, Italy, 2009.

[CCM04] William W Cohen, Vitor R Carvalho, and Tom M Mitchell. Learning to Classify Email into Speech Acts. In *Conference on Empirical Methods in Natural Language Processing*, pages 309–316, Barcelona, Spain, 2004.

[CCMS10] Alessandro Ciaramella, Mario G C A Cimino, Francesco Marcelloni, and Umberto Straccia. Combining Fuzzy Logic and Semantic Web to Enable Situation-Awareness in Service Recommendation. In *Conference on Database and expert systems applications*, pages 31–45, Bilbao, Spain, 2010.

[CDM+00] Keith Cheverst, Nigel Davies, Keith Mitchell, Adrian Friday, and Christos Efstratiou. Developing a context-aware electronic tourist guide. In *Conference on Human factors in computing systems*, pages 17–24, The Hague, Netherlands, 2000.

[CER+08] Henriette Cramer, Vanessa Evers, Satyan Ramlal, Maarten Someren, Lloyd Rutledge, Natalia Stash, Lora Aroyo, and Bob Wielinga. The effects of transparency on trust in and acceptance of a content-based art recommender. *User Modeling and User-Adapted Interaction*, 18(5):455–496, 2008.

[CFJ03] Harry Chen, Tim Finin, and Anupam Joshi. An Intelligent Broker for Context-Aware Systems. In *Conference on Ubiquitous Computing*, pages 12–15, Seattle, WA, USA, 2003.

[CGM+99] Mark Claypool, Anuja Gokhale, Tim Miranda, Pavel Murnikov, Dmitry Netes, and Matthew Sartin. Combining Content-Based and Collaborative Filters in an Online Newspaper. In *Workshop on Recommender Systems*, pages 40–48, Berkeley, CA, USA, 1999.

[CH01] In-Gook Chun and In-Sik Hong. The implementation of knowledge-based recommender system for electronic commerce using Java expert system library. In *Symposium on Industrial Electronics*, pages 1766–1770, Pusan, Korea, 2001. IEEE.

[CJT+06] Chee-yong Chan, H V Jagadish, Kian-lee Tan, Anthony K H Tung, and Zhenjie Zhang. Finding k-Dominant Skylines in High Dimensional Space.

In *Conference on Management of data*, pages 503–514, Chicago, IL, USA, 2006.

[CK00] Guanling Chen and David Kotz. A Survey of Context-Aware Mobile Computing Research. Technical report, Dartmouth College, 2000.

[CK02] Guanling Chen and David Kotz. Solar: An Open Platform for Context-Aware Mobile Applications. In *Conference on Pervasive Computing*, pages 41–47, Zurich, Switzerland, 2002.

[CL08] Joon Yeon Choeh and Hong Joo Lee. Mobile push personalization and user experience. *AI Communications*, 21(2-3):185–193, 2008.

[CLA+03] Dan Cosley, Shyong K Lam, Istvan Albert, Joseph A Konstan, and John Riedl. Is seeing believing? In *Conference on Human factors in computing systems*, pages 585–592, Fort Lauderdale, FL, USA, 2003.

[CM06] Giuseppe Carenini and Johanna D Moore. Generating and evaluating evaluative arguments. *Artificial Intelligence*, 170(11):925–952, 2006.

[Coh80] Sheldon Cohen. Aftereffects of stress on human performance and social behavior: A review of research and theory. *Psychological Bulletin*, 88(1):82–108, 1980.

[Cov89] Stephan R Covey. *The 7 Habits of Highly Effective People.* Simon & Schuster, 1989.

[COYO06] Kenta Cho, Yuzo Okamoto, Tomohiro Yamasaki, and Masayuki Okamoto. Implementation of Telematics Services with Context-Aware Agent Framework. In *Symposium on Ubiquitous Computing Systems*, pages 98–110, Seoul, Korea, 2006.

[CP07] Li Chen and Pearl Pu. Hybrid Critiquing-based Recommender Systems. In *Conference on Intelligent user interfaces*, pages 22–31, Honolulu, HI, USA, 2007.

[CSC+08] Doreen Cheng, Henry Song, Hyuk Cho, Sangoh Jeong, Swaroop Kalasapur, and Alan Messer. Mobile Situation-Aware Task Recommendation Application. In *Conference on Next Generation Mobile Applications, Services, and Technologies*, pages 228–233, Cardiff, Wales, UK, 2008.

[DA99] Anind K Dey and Gregory D Abowd. Towards a Better Understanding of Context and Context-Awareness. In *1st international symposium on Handheld and Ubiquitous Computing*, pages 304 – 307, Karlsruhe, Germany, 1999.

[DA00] Anind K Dey and Gregory D Abowd. CyberMinder: a context-aware system for supporting reminders. In *Symposium on Handheld and Ubiquitous Computing*, pages 172–186, Bristol, UK, 2000.

[DAS01] Anind K Dey, Gregory D Abowd, and Daniel Salber. A Conceptual Framework and a Toolkit for Supporting the Rapid Prototyping of Context-Aware Applications. *Human-Computer Interaction*, 16(2):97–166, 2001.

[Dav89] Fred D Davis. Perceived Usefulness, Perceived Ease of Use, and User Acceptance of Information Technology. *MIS Quarterly*, 13(3):319–340, 1989.

[Daw79] Robyn M Dawes. The robust beauty of improper linear models in decision making. *American Psychologist*, 34(7):571–582, 1979.

[dCGM01] Luis M de Campos, Jose A Gamez, and Serafin Moral. Simplifying explanations in Bayesian belief networks. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 9(4):461–48, 2001.

[dCNPG09] Berardina de Carolis, Nicole Novielli, Vito Leonardo Plantamura, and Enrica Gentile. Generating comparative descriptions of places of interest in the tourism domain. In *Conference on Recommender Systems*, pages 277–280, New York, NY, USA, 2009.

[DCSH04] Susan Dumais, Edward Cutrell, Raman Sarin, and Eric Horvitz. Implicit queries (IQ) for contextualized search. In *Conference on Research and Development in Information Retrieval*, page 594, Sheffield, UK, 2004.

[de 07] Luis Antonio de la Fuente Layos. Mobilität im Personenverkehr in Europa Statistik kurz gefasst. Technical report, eurostat, 2007.

[DEO+05] Sandor Dornbush, Jesse English, Tim Oates, Zary Segall, and Anumpam Joshi. XPod: A Human Activity Aware Learning Mobile Music Player. In *Conference on Mobile Technology, Applications and Systems*, pages 1–6, Guangzhou, China, 2005.

[DHT06] Simon Duff, James Harland, and John Thangarajah. On Proactivity and Maintenance Goals. In *Conference on Autonomous Agents and Multiagent Systems*, pages 1033–1040, Hakodate, Japan, 2006.

[Die92] Joachim Diederich. Explanation and artificial neural networks. *International Journal of Man-Machine Studies*, 37(3):335–355, 1992.

[Dij59] Edsger W Dijkstra. A note on two problems in connection with graphs. In *Numerische Mathematik 1*, pages 269–271. 1959.

[DKH+98] Brent J Dahlen, Joseph A Konstan, Jonathan L Herlocker, Nathaniel Good, Al Borchers, and John T Riedl. Jump-Starting MovieLens: User Benefits of Starting a Collaborative Filtering System with "Dead Date". Technical report, University of Minnesota, 1998.

[DMH+95] T Dingus, D McGehee, M Hulse, S Jahns, and N Manakkal. Travtrek evaluation task C3 Camera Car study. Technical report, Office of Safety and Traffic Operations, 1995.

[DMM+04] Mark Dunlop, Alison Morrison, Stephen Mccallum, Piotr Ptaskinski, and Fraser Stewart. Focussed Palmtop Information Access Combining Starfield Displays with Profile-Based Recommendations. In *Workshop on Mobile and Ubiquitous Information Access*, pages 79–89, Glasgow, UK, 2004.

[Don82] Edmund Donges. Aspekte der Aktiven Sicherheit bei der Führung von Personenkraftwagen. *Autmobil-Industrie*, 27(2):183–190, 1982.

[Don99] Edmund Donges. A Conceptual Framework for Active Safety in Road Traffic. *Vehicle System Dynamics*, 32(2-3):113–128, 1999.

[Dou04] Paul Dourish. What we talk about when we talk about context. *Personal and Ubiquitous Computing*, 8(1):19–30, 2004.

[DPH+09] Nicolas Ducheneaut, Kurt Partridge, Qingfeng Huang, Bob Price, Mike Roberts, Ed H Chi, Victoria Bellotti, and Bo Begole. Collaborative Filtering Is Not Enough? Experiments with a Mixed-Model Recommender for Leisure Activities. In *Conference on User Modeling, Adaptation and Personalization*, pages 295–306, Trento, Italy, 2009.

[DPW09] Daniel Delling, Thomas Pajor, and Dorothea Wagner. Accelerating Multimodal Route Planning by Access-Nodes. In *Symposium on Algorithms*, pages 587–598, Copenhagen, Denmark, 2009.

[DR85] Edward L Deci and Richard M Ryan. *Intrinsic motivation and self-determination in human behavior.* Plenum Press, 1985.

[DV04] Fred D Davis and Viswanath Venkatesh. Toward Preprototype User Acceptance Testing of New Information Systems: Implications for Software Project Management. *Transactions on Engineering Management*, 51(1):31–46, 2004.

[EBJ09] Ronald Ecker, Verena Broy, and Nina Joshi. Toggle strategies for the POI selection via the iDrive controller. In *Conference on Automotive User Interfaces and Interactive Vehicular Applications*, pages 71–74, Essen, Germany, 2009.

[End00] Mica R Endsley. Theoretical underpinnings of situation awareness: A critical review. In *Situation Awareness Analysis and Measurement*, pages 3–27. CRC Press, 2000.

[ESW+05] Jaime Esteban, Andrew Starr, Robert Willetts, Paul Hannah, and Peter Bryanston-Cross. A Review of data fusion models and architectures: towards engineering guidelines. *Neural Computing and Applications*, 14(4):273–281, 2005.

[F8̈7] H J Förster. Menschliches Verhalten, eine vergessene Ingenieurwissenschaft? *Autmobil-Industrie*, 32(6):671– 678, 1987.

[F0̈5] Janos Fülöp. Introduction to Decision Making Methods. 2005.

[Fah08] Yetim Fahri. A Framework for Organizing Justifications for Strategic Use in Adaptive Interaction Contexts. In *Conference on Information Systems*, pages 815–825, Galway, Ireland, 2008.

[Fas95] Wolfgang Fastenmeier. Die Verkehrssituation als Analyseeinheit im Verkehrssystem. In *Autofahrer und Verkehrssituation : neue Wege zur Bewertung von Sicherheit und Zuverlässigkeit moderner Straßenverkehrssysteme*, chapter 3, pages 27 – 28. TÜV Rheinland, 1995.

[FB08] Alexander Felfernig and Robin Burke. Constraint-based Recommender Systems. In *Conference on Electronic Commerce*, Innsbruck, Austria, 2008.

[FGL+08] Alexander Felfernig, Bartosz Gula, Gerhard Leitner, Marco Maier, Rudolf Melcher, and Erich Teppan. Persuasion in Knowledge-Based Recommendation. In *Conference on Persuasive Technology*, pages 71–82, Oulu, Finland, 2008.

[FGM05] M Julia Flores, José A Gámez, and Serafín Moral. Abductive Inference in Bayesian Networks : Finding a Partition of the Explanation Space. In *Conference on Symbolic and Quantitative Approaches to Reasoning with Uncertainty*, pages 63–75, Barcelona, Spain, 2005.

[FI93] Usama M Fayyad and Keki B Irani. Multi-interval discretisation of continuous-valued attributes. In *Conference on Artificial Intelligence*, pages 1022–1027, Chambéry, France, 1993.

[Fis74] Peter C Fishburn. Exceptional Paper–Lexicographic Orders, Utilities and Decision Rules: A Survey. *Management Science*, 20(11):1442–1471, 1974.

[FLW+06] Yusuke Fukazawa, Marko Luther, Matthias Wagner, Atsuki Tomioka, Takefumi Naganuma, Kunihiro Fujii, Shoji Kurakake, Network Labs, N T T Docomo, and Docomo Euro-labs. Situation-aware Task-based Service Recommendation, 2006.

[FMS+10] Alexander Felfernig, Monika Mandl, Stefan Schippel, Monika Schubert, and Erich Teppan. Adaptive Utility-Based Recommendation. In *Conference on Industrial Engineering and Other Applications of Applied Intelligent Systems*, pages 641–650, Cordoba, Spain, 2010.

[FNJT07] Alan Fern, Sriraam Natarajan, Kshitij Judah, and Prasad Tadepalli. A Decision-Theoretic Model of Assistance. In *Conference on Artificial Intelligence*, pages 1879–1884, Hyderabad, India, 2007.

[For11] Ford. Ford Explorer New Lane Keeping Technology Can Help Detect Drowsy Drivers, Suggest Coffee Breaks, 2011.

[Fra46] Viktor Emil Frankl. *Man's Search For Meaning.* Koesel, 1946.

[FYW+04] Xiaocong Fan, John Yen, Rui Wang, Shuang Sun, and Richard A. Volz. Context-centric proactive information delivery. In *Conference on Intelligent Agent Technology*, pages 31–37, Beijing, China, 2004.

[GB99] Shirley Gregor and Izak Benbasat. Explanations from Intelligent Systems: Theoretical Foundations and Implications for Practice. *MIS Quarterly*, 23(4):497, 1999.

[Gig07] Gerd Gigerenzer. *Gut Feelings: The Intelligence of the Unconscious.* Viking Adult, 2007.

[Gla05] Malcolm Gladwell. *Blink: the power of thinking without thinking.* Little, Brown and Co., 2005.

[GLN+10] Robert Geisberger, Dennis Luxen, Sabine Neubauer, Peter Sanders, and Lars Völker. Fast Detour Computation for Ride Sharing. In *Workshop on Algorithmic Approaches for Transportation Modelling, Optimization, and Systems*, Liverpool, UK, 2010.

[GLPS95] Paul Green, William Levison, Gretchen Paelke, and Colleen Serafin. Preliminary Human Factors Design Guidelines for Driver Information Systems. Technical report, University of Michigan, 1995.

[GNL+09] Ruinan Gong, Ke Ning, Qing Li, David OSullivan, Yuliu Chen, and Stefan Decker. Context modeling and measuring for proactive resource recommendation in business collaboration. *Computers & Industrial Engineering*, 57(1):27–36, 2009.

[GNOT92] David Goldberg, David Nichols, Brian M. Oki, and Douglas Terry. Using collaborative filtering to weave an information tapestry. *Communications of the ACM*, 35(12):61–70, 1992.

[GP11] Sandro Rodriguez Garzon and Mark Poguntke. The Personal Adaptive In-Car HMI : Integration of External Applications for Personalized Use. In *Workshop on Augmenting User Models with Real World Experiences to Enhance Personalization and Adaptation*, pages 35–46, Girona, Spain, 2011.

[Gre99] Paul Green. The 15-Second Rule for Driver Information Systems. In *Conference on Intelligent Transportation Systems*, pages 1–9, Washington, DC, USA, 1999.

[GSB99] Michael A Goodrich, Wynn C Stirling, and Erwin R Boer. Satisficing Revisited. *Minds and Machines*, 10(1):79–109, 1999.

[Gut98] Robert H Guttman. *Merchant Differentiation through Integrative Negotiation in Agent-mediated Electronic Commerce*. Master thesis, MIT, 1998.

[GVCF07] Laurent-Walter Goix, Massimo Valla, Laura Cerami, and Paolo Falcarin. Situation Inference for Mobile Users: A Rule Based Approach. In *Conference on Mobile Data Management*, pages 299–303, Mannheim, Germany, 2007.

[GVWB11] Daniel Gallego-Vico, Wolfgang Wörndl, and Roland Bader. A Study on Proactive Delivery of Restaurant Recommendations for Android Smartphones. In *Workshop on Personalization in Mobile Applications*, Chicago, IL, 2011.

[HB06] Bowen Hui and Craig Boutilier. Who's asking for help?: a Bayesian approach to intelligent assistance. In *Conference on Intelligent user interfaces*, pages 186–193, Sydney, Australia, 2006.

[HBH+98] Eric Horvitz, Jack Breese, David Heckerman, David Hovel, and Koos Rommelse. The Lumiere Project : Bayesian User Modeling for Inferring the Goals and Needs of Software Users. In *Conference on Uncertainty in Artificial Intelligence*, pages 256–265, Madison, WI, USA, 1998.

[HCMB05] Monika Henzinger, Bay-Wei Chang, Brian Milch, and Sergey Brin. Query-Free News Search. *World Wide Web*, 8(2):101–126, 2005.

[Hel04] Wolfram Hell. Unfallursache Einschlafen - Auftreten und Pravention von Mudigkeitsunfallen im Straßenverkehr. In *Deutscher Verkehrsgerichtstag*, Goslar, Germany, 2004.

[HG97] Peter E Hart and Jamey Graham. Query-free information retrieval. *IEEE Expert: Intelligent Systems and Their Applications*, 12(5):32–37, 1997.

[HG09] Jens Hoffmann and Jens Gayko. Fahrerwarnelemente. In *Handbuch Fahrerassistenzsysteme: Grundlagen, Komponenten und Systeme für aktive Sicherheit und Komfort*, chapter 24, pages 343–354. Vieweg+Teubner Verlag, 2009.

[HJH99] Eric Horvitz, Andy Jacobs, and David Hovel. Attention-Sensitive Alerting. In *Conference on Uncertainty and Artificial Intelligence*, pages 305–313, Stockholm, Sweden, 1999.

[HK01] Jonathan L Herlocker and Joseph A Konstan. Content-independent task-focused recommendation. *IEEE Internet Computing*, 5(6):40–47, 2001.

[HKM+04] Karen Zita Haigh, Liana M Kiff, Janet Myers, Valerie Guralnik, Christopher W Geib, John Phelps, and Tom Wagner. The Independent LifeStyle Assistant (I.L.S.A.): AI Lessons Learned. In *Conference on Innovative Applications of Artificial Intelligence*, pages 852–857, San Jose, CA, USA, 2004.

[HKR00] Jonathan L Herlocker, Joseph A Konstan, and John Riedl. Explaining collaborative filtering recommendations. In *Conference on Computer Supported Cooperative Work*, pages 241–250, Philadelphia, PA, USA, 2000.

[HKS07] Eric Horvitz, Paul Koch, and Muru Subramani. Mobile Opportunistic Planning: Methods and Models. In *Conference on User Modeling*, pages 228–237, Corfu, Greece, 2007.

[HKT03] Kouichi Hanamura, Kenta Kawabata, and Tomohiro Takagi. Intention-aware information-delivery system for mobile device. In *Conference on Fuzzy Systems*, pages 1110–1115, St. Louis, MO, USA, 2003.

[HKTR04] Jonathan L Herlocker, Joseph A Konstan, Loren G Terveen, and John T Riedl. Evaluating collaborative filtering recommender systems. *ACM Transactions on Information Systems*, 22(1):5–53, 2004.

[HMBR05] Harald Holz, Heiko Maus, Ansgar Bernardi, and Oleg Rostanin. A Lightweight Approach for Proactive , Task-Specific Information Delivery. In *Conference on Knowledge Management*, Graz, Austria, 2005.

[HMPR04] Alan R Hevner, Salvatore T March, Jinsoo Park, and Sudha Ram. Design Science in Information Systems Research. *MIS Quarterly*, 28(1):75–105, 2004.

[HNV06] Tzvetan Horozov, Nitya Narasimhan, and Venu Vasudevan. Using Location for Personalized POI Recommendations in Mobile Environments. In *Symposium on Applications and the Internet*, pages 124–129, Phoenix, AZ, USA, 2006. IEEE.

[Hor93] Eric Horvitz. Lumiere Project: Bayesian Reasoning for Automated Assistance. Technical report, Microsoft Research, Redmond, WA, USA, 1993.

[Hor99] Eric Horvitz. Principles of mixed-initiative user interfaces. In *Conference on Human factors in computing systems*, pages 159–166, Pittsburgh, PA, USA, 1999.

[HS59] Franz Halberg and A N Stephens. Susceptibility to ouabain and physiologic circadian periodicity. In *Minn. Acad. Sci.*, pages 139–143, 1959.

[HSAR07] Stefan Hoch, Manfred Schweigert, Frank Althoff, and Gerhard Rigoll. The BMW SURF Project: A Contribution to the Research on Cognitive Vehicles. In *Symposium on Intelligent Vehicles*, pages 692–697, Istanbul, Turkey, 2007.

[HSB09] Annika Hinze, Kai Sachs, and Alejandro Buchmann. Event-based applications and enabling technologies. In *Conference on Distributed Event-Based Systems*, Nashville, TN, USA, 2009.

[HSK09] Jong-yi Hong, Eui-ho Suh, and Sung-Jin Kim. Context-aware systems: A literature review and classification. *Expert Systems with Applications*, 36(4):8509–8522, 2009.

[HSTA+07] David R Hardoon, John Shawe-Taylor, Antti Ajanki, Kai Puolamäki, and Samuel Kaski. Information Retrieval by Inferring Implicit Queries from Eye Movements. In *Conference on Artificial Intelligence and Statistics*, Minneapolis, MN, USA, 2007.

[HTW00] D Haigney, R Taylor, and S Westerman. Concurrent mobile (cellular) phone use and driving performance: task demand characteristics and compensatory processes. *Transportation Research Part F: Traffic Psychology and Behaviour*, 3(3):113–121, 2000.

[Hu09] Beibei Hu. Context-based Information Delivery for Mobile Users in Emergency Situations. In *Doctoral Consortium on User Modeling, Adaptation and Personalization*, pages 61–66, Trento, Italy, 2009.

[HV03] Annika Hinze and Agnès Voisard. Location- and Time-Based Information Delivery in Tourism. In *Symposium on Advances in Spatial and Temporal Databases*, pages 489–507, Santorini Island, Greece, 2003.

[HW90] John R Hauser and Birger Wernerfelt. An evaluation cost model of consideration sets. *Journal of Consumer Research*, 16(4):393–408, 1990.

[HY81] Ching-Lai Hwang and Kwangsun Yoon. *Multiple attribute decision making: methods and applications : a state-of-the-art survey.* Springer, 1981.

[Iba07] Wayne Iba. When Is Assistance Really Helpful? In *Symposium on Interaction Challenges for Intelligent Assistants*, Palo Alto, CA, USA, 2007.

[IL00] Sheena S Iyengar and Mark R Lepper. When choice is demotivating: can one desire too much of a good thing? *Journal of personality and social psychology*, 79(6):995–1006, 2000.

[iL12] iOnRoad Ltd. iOnRoad - Augmented Reality Driving, 2012.

[IP05] Charles L Isbell and Jeff S Pierce. An IP Continuum for Adaptive Interface Design. In *Conference on Human-Computer Interaction*, Las Vegas, NV, USA, 2005.

[Jam11] Anthony Jameson. What Should Recommender Systems People Know About the Psychology of Choice and Decision Making, 2011.

[JB04] Gareth J F Jones and Peter J Brown. Context-Aware Retrieval for Ubiquitous Computing Environments. In *Workshop on Mobile and Ubiquitous Information Access*, pages 227–243, Udine, Italy, 2004.

[Jef61] Harold Jeffreys. *Theory of Probability.* Oxford, 1961.

[JFM97] Thorsten Joachims, Dayne Freitag, and Tom Mitchell. WebWatcher: A Tour Guide for the World Wide Web. In *Conference on Artificial Intelligence*, pages 770–775, Nagoya, Japan, 1997.

[Jon05] Gareth J F Jones. Challenges and Opportunities of Context-Aware Information Access. In *Workshop on Ubiquitous Data Management*, pages 53–62, Tokyo, Japan, 2005. Ieee.

[JP08] Nicolas Jones and Pearl Pu. User Acceptance Issues in Music Recommender Systems. Technical report, EPFL, Lausanne, Switzerland, 2008.

[JZFF11] Dietmar Jannach, Markus Zanker, Alexander Felfernig, and Gerhard Friedrich. *Recommender Systems: An Introduction.* Cambridge University Press, 1 edition, 2011.

[Kaa05] Eija Kaasinen. *User acceptance of mobile services.* Phd thesis, Tampere University of Technology, 2005.

[KABO10] Alexandros Karatzoglou, Xavier Amatriain, Linas Baltrunas, and Nuria Oliver. Multiverse recommendation: n-dimensional tensor factorization for context-aware collaborative filtering. In *Conference on Recommender Systems*, pages 79–86, Barcelona, Spain, 2010.

[KBH+07] Antonio Krüger, Jörg Baus, Dominik Heckmann, Michael Kruppa, and Rainer Wasinger. Adaptive Mobile Guides. In *The Adaptive Web*, volume 4321 of *Lecture Notes in Computer Science*, chapter 17, pages 521–549. Springer, 2007.

[KBKPS09] Thomas Kleine-Besten, Ulrich Kersken, Werner Pöchmüller, and Heiner Schepers. Navigation und Telematik. In *Handbuch Fahrerassistenzsysteme: Grundlagen, Komponenten und Systeme für aktive Sicherheit und Komfort*, chapter 39, pages 599–624. Vieweg+Teubner Verlag, 2009.

[KBS+04] Antonio Krüger, Andreas Butz, Christoph Stahl, Christian Müller, Rainer Wasinger, Karl-Ernst Steinberg, and Andreas Dirschl. The Connected User Interface: Realizing a Personal Situated Navigation Service. In *Conference on Intelligent User Interface*, pages 161–168, Madeira, Portugal, 2004.

[KBV09] Yehuda Koren, Robert Bell, and Chris Volinsky. Matrix Factorization Techniques for Recommender Systems. *Computer*, 42(8):30–37, 2009.

[KH10] Ece Kamar and Eric Horvitz. Investigation of Principles of Context-Sensitive Reminding. Technical report, Microsoft Research, Redmond, WA, USA, 2010.

[KIGI09] Kazuki Kodama, Yuichi Iijima, Xi Guo, and Yoshiharu Ishikawa. Skyline queries based on user locations and preferences for making location-based recommendations. In *Workshop on Location Based Social Networks*, pages 9–16, Seattle, WA, USA, 2009.

[Kir07] Katja Kircher. Driver distraction A review of the literature. Technical report, VTI, Linköping, Sweden, 2007.

[KKY04] Joonhee Kwon, Sungrim Kim, and Yongik Yoon. Just-in-Time Recommendation Using Multi-agents for Context-Awareness in Ubiquitous Computing Environment. In *Conference on Database Systems for Advanced Applications*, pages 656–669, Jeju Island, Korea, 2004.

[Kla09] André Klahold. *Empfehlungssysteme*. Vieweg+Teubner Verlag, 2009.

[Kon94] Igor Kononenko. Estimating Attributes: Analysis and Extensions of RELIEF. In *Conference on machine learning*, pages 171–182, Catania, Italy, 1994.

[Kon04] Joseph A Konstan. Introduction to recommender systems. *ACM Transactions on Information Systems*, 22(1):1–4, 2004.

[Kop05] Matthias Kopf. Was nützt es dem Fahrer, wenn Fahrerinformations- und -assistenzsysteme etwas über ihn wissen? In *Fahrerassistenzsysteme mit maschineller Wahrnehmung*, pages 117–139. Springer, 2005.

[Kos05] Timo Kosch. *Situationsadaptive Kommunikation in Automobilen Ad-hoc Netzen*. Phd thesis, Technische Universität München, Tampere, 2005.

[KR76] Ralph Keeney and Howard Raiffa. *Decisions with Multiple Objectives: Preferences and Value Tradeoffs*. Cambridge University Press, 1976.

[KS87] Kevin Lane Keller and Richard Staelin. Effects of Quality and Quantity of Information on Decision Effectiveness. *Journal of Consumer Research*, 14(2):200–213, 1987.

[KSS97] Henry Kautz, Bart Selman, and Mehul Shah. Referral Web: combining social networks and collaborative filtering. *Communications of the ACM*, 40(3):63–65, 1997.

[KT79] Daniel Kahneman and Amos Tversky. Prospect Theory: An Analysis of Decision under Risk. *Econometrica*, 47(2):263–292, 1979.

[KYS05] Ohbyung Kwon, Keedong Yoo, and Euiho Suh. UbiDSS: a proactive intelligent decision support system as an expert system deploying ubiquitous computing technologies. *Expert Systems with Applications*, 28(1):149–161, 2005.

[LÖ3] George Löwenstein. *Time and Decision: Economic and Psychological Perspectives on Intertemporal Choice*. Russell Sage Foundation, 2003.

[LB07] Danielle H Lee and Peter Brusilovsky. Fighting Information Overflow with Personalized Comprehensive Information Access : A Proactive Job

Recommender. In *Conference on Autonomic and Autonomous Systems*, Athens, Greece, 2007.

[LBR+04] James Llinas, Christopher Bowman, Galina Rogova, Alan Steinberg, Ed Waltz, and Frank White. Revisiting the JDL Data Fusion Model II. In *Conference on Information Fusion*, Quebec, Canada, 2004.

[LC10] Kun Chang Lee and Heeryon Cho. Designing a Ubiquitous Decision Support Engine for Context Prediction : General Bayesian Network Approach. *Science And Technology*, 3(3):25–36, 2010.

[LD03] Carmen Lacave and Francisco J Diez. A review of explanation methods for Bayesian networks. *The Knowledge Engineering Review*, 17(02):107 – 127, 2003.

[LD11] Brian Y Lim and Anind K Dey. Design of an Intelligible Mobile Context-Aware Application. In *Conference on Human Computer Interaction with Mobile Devices and Services*, pages 157–166, Stockholm, Sweden, 2011.

[LDA09] Brian Y Lim, Anind K Dey, and Daniel Avrahami. Why and why not explanations improve the intelligibility of context-aware intelligent systems. In *Conference on Human Factors in Computing Systems*, pages 2119–2128, Boston, MA, USA, 2009.

[Lee10] George Lee. An Automatic Personalized Context-Aware Event Notification System for Mobile Users, 2010.

[Lie97] Henry Lieberman. Autonomous interface agents. In *Conference on Human-Computer Interface*, pages 67–74, Atlanta, GA, USA, 1997.

[LK77] J Richard Landis and Gary G Koch. The Measurement of Observer Agreement for Categorical Data. *Biometrics*, 33(1):159, 1977.

[LN92] Michael G Lamming and William M Newman. Activity-based Information Retrieval : Technology in Support of Personal Memory. In *World Computer Congress on Information Processing*, pages 68–81, Madrid, Spain, 1992.

[LPB99] Mary Frances Luce, John W. Payne, and James R. Bettman. Emotional Trade-Off Difficulty and Choice. *Journal of Marketing Research*, 36(2):143, 1999.

[LS01] David B Leake and Ryan Scherle. Towards context-based search engine selection. In *Conference on Intelligent user interfaces*, pages 109–112, Santa Fe, NM, USA, 2001.

[LS03] Henry Lieberman and Ted Selker. Agents for the User Interface. In *Handbook of Agent Technology*. MIT Press, 2003.

[LSY03] Greg Linden, Brent Smith, and Jeremy York. Amazon.com recommendations: item-to-item collaborative filtering. *IEEE Internet Computing*, 7(1):76–80, 2003.

[LVR+08] Gilly Leshed, Theresa Velden, Oya Rieger, Blazej Kot, and Phoebe Sengers. In-car gps navigation. In *Conference on Human Factors in Computing Systems*, pages 1675–1684, Florence, Italy, 2008.

[LZS07] Sute Lei, Kang Zhang, and Edwin Sha. Applying Situation Awareness to Mobile Proactive Information Delivery. In *Conference on Emerging Directions in Embedded and Ubiquitous Computing*, pages 592–603, Taipei, Taiwan, 2007.

[Mae94] Pattie Maes. Agents that reduce work and information overload. *Communications of the ACM*, 37(7):30–40, 1994.

[Mam77] Ebrahim H Mamdani. Application of Fuzzy Logic to Approximate Reasoning Using Linguistic Synthesis. *IEEE Transactions on Computers*, C-26(12):1182–1191, 1977.

[Mar83] Alain Paul Martin. *Think Proactive: New Insights into Decision Making.* Professional Dev Inst Pr, 1983.

[MBCS00] Paul P Maglio, Rob Barrett, Christopher S Campbell, and Ted Selker. SUITOR: an attentive information system. In *Conference on Intelligent user interfaces*, pages 169–176, New Orleans, LA, USA, 2000.

[MC07] Nikos Manouselis and Constantina Costopoulou. Analysis and Classification of Multi-Criteria Recommender Systems. *World Wide Web*, 10(4):415–441, 2007.

[McF02] Daniel McFarlane. Comparison of Four Primary Methods for Coordinating the Interruption of People in Human-Computer Interaction. *Human-Computer Interaction*, 17(1):63–139, 2002.

[McS05] David McSherry. Explanation in Recommender Systems. *Artificial Intelligence Review*, 24(2):179–197, 2005.

[MdRS05] Ian Millard, David de Roure, and Nigel Shadbolt. Contextually Aware Information Delivery in Pervasive Computing Environments. In *Workshop on Location- and Context-Awareness*, pages 189 – 197, Oberpfaffenhofen, Germany, 2005.

[MF00] Barry McGuinness and Louise Foy. A subjective measure of SA: The Crew Awareness Rating Scale. In *Conference on human performance, situation awareness, and automation*, Savannah, Georgia, 2000.

[MG02] Jonathan P Munson and Vineet K Gupta. Location-based notification as a general-purpose service. In *Workshop on Mobile commerce*, pages 40–44, Atlanta, GA, USA, 2002.

[MGT+87] Thomas W Malone, Kenneth R Grant, Franklyn A Turbak, Stephen A Brobst, and Michael D Cohen. Intelligent information-sharing systems. *Communications of the ACM*, 30(5):390–402, 1987.

[Mil56] George Miller. The Magical Number Seven, Plus or Minus Two. *The Psychological Review*, 63:81–97, 1956.

[MKHG07] Marko Modsching, Ronny Kramer, Klaus Ten Hagen, and Ulrike Gretzel. Effectiveness of Mobile Recommender Systems for Tourist Destinations: A User Evaluation. In *Workshop on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications*, pages 663–668, Dortmund, Germany, 2007.

[Mon03] Miquel Montaner. A Taxonomy of Recommender Agents on the Internet. *Artificial Intelligence Review*, 19(4):285–330, 2003.

[MRB05] Nader Mirzadeh, Francesco Ricci, and Mukesh Bansal. Feature Selection Methods for Conversational Recommender Systems. In *Conference on e-Technology, e-Commerce and e-Service*, pages 772–777, Hong Kong, 2005.

[MRMS04] Kevin McCarthy, James Reilly, Lorraine McGinty, and Barry Smyth. Thinking Positively - Explanatory Feedback for Conversational Recommender Systems. In *Workshop on Explanation in CBR*, pages 115–124, Madrid, Spain, 2004.

[MS00] Natalia Marmasse and Chris Schmandt. Location-Aware Information Delivery with ComMotion. In *Symposium on Handheld and Ubiquitous Computing*, volume 1927 of *Lecture Notes in Computer Science*, pages 157–171, Bristol, UK, 2000.

[MS03] Lorraine McGinty and Barry Smyth. On the Role of Diversity in Conversational Recommender Systems. In *Conference on Case-Based Reasoning Research and Development*, pages 276–290, Trondheim, Norway, 2003. .

[MSC+06] Kevin McCarthy, Maria Salamó, Lorcan Coyle, Lorraine McGinty, Barry Smyth, and Paddy Nixon. Group recommender systems. In *Conference on Intelligent user interfaces*, pages 267–269, Sydney, Australia, 2006.

[MTM05] Miikka Miettinen, Ville H Tuulos, and Petri Myllym. A Testbed for Proactive Information Retrieval. In *Workshop on Context Awareness for Proactive Systems*, Helsinki, Finland, 2005.

[MW05] Naoto Mukai and Toyohide Watanabe. Proactive Route Planning Based on Expected Rewards for Transport Systems. In *Conference on Tools with Artificial Intelligence*, pages 51–57, Hong Kong, 2005.

[MWST09] Alexander Meschtscherjakov, David Wilfinger, Thomas Scherndl, and Manfred Tscheligi. Acceptance of future persuasive in-car interfaces towards a more economic driving behaviour. In *Conference on AutomotiveUI*, pages 81–88, Essen, Germany, 2009.

[MYs07] Karen Myers and Neil Yorke-smith. Proactive Behavior of a Personal Assistive Agent. In *Workshop on Metareasoning in Agent-Based Systems*, pages 31–45, Honolulu, HI, 2007.

[Naa04] Karl Naab. Sensorik- und Signalverarbeitungsarchitekturen fur Fahrerassistenz und Aktive Sicherheit. In *Tagung Aktive Sicherheit durch Fahrerassistenz*, Munich, Germany, 2004.

[Nak04] Robbie T Nakatsu. Explanatory Power of Intelligent Systems : A Research Framework. In *Conference on Decision Support Systems*, pages 568–577, Prato, Italy, 2004.

[NC05] Conor Nugent and Pádraig Cunningham. A Case-Based Explanation System for Black-Box Systems. *Artificial Intelligence Review*, 24(2):163–178, 2005.

[NJTF07] Sriraam Natarajan, Kshitij Judah, Prasad Tadepalli, and Alan Fern. A Decision-Theoretic Model of Assistance - Evaluation , Extensions and Open Problems. In *Conference on Artificial Intelligence*, pages 1879–1884, Hyderabad, India, 2007.

[NPE08] Ulf H Nielsen, Jean-Philippe Pellet, and Andre Elisseeff. Explanation Trees for Causal Bayesian Networks. In *Conference on Uncertainty in Artificial Intelligence*, pages 427–434, Helsinki, Finland, 2008.

[NR98] Karl Naab and Günter Reichart. Grundlagen der Fahrerassistenz und Anforderungen aus Nutzersicht. In *Seminar Fahrerassistenzsysteme, Haus der Technik*, Essen, Germany, 1998.

[Oli80] Richard L Oliver. A Cognitive Model of the Antecedents and Consequences of Satisfaction Decisions. *Journal of Marketing Research*, 17(4):460–469, 1980.

[ONMU06] Kenta Oku, Shinsuke Nakajima, Jun Miyazaki, and Shunsuke Uemura. Context-Aware SVM for Context-Dependent Information Recommendation. In *Conference on Mobile Data Management*, pages 109–109, Nara, Japan, 2006.

[OSJ99] Reinhard Oppermann, Marcus Specht, and Igor Jaceniak. Hippie: A Nomadic Information System. In *Symposium on Handheld and ubiquitous computing*, pages 330–333, Karlsruhe, Germany, 1999.

[PB07] Michael J Pazzani and Daniel Billsus. Content-Based Recommendation Systems. In *The Adaptive Web*, chapter 10, pages 325–341. Springer, 2007.

[PBD+07] Mari Carmen Puerta Melguizo, Toine Bogers, Anita Deshpande, Lou Boves, and Antal van den Bosch. What a Proactive Recommendation System Needs: Relevance, Non-Intrusiveness, and a New Long-Term Memory. In *Conference on Enterprise Information Systems*, pages 86–91, Madeira, Portugal, 2007.

[PBJ93] John W Payne, James R Bettman, and Eric J Johnson. *The adaptive decision maker.* Cambridge University Press, 1993.

[PBKB07] Kyounga Park, Michael G H Bell, Ioannis Kaparias, and Klaus Bogenberger. Learning user preferences of route choice behaviour for adaptive route guidance. *Intelligent Transport Systems*, 1(2):159–166, 2007.

[PBW10] Vivian Prinz, Roland Bader, and Wolfgang Wörndl. Design, implementation and evaluation of a structured P2P algorithm for vehicular communication systems. In *Conference on Wireless and Mobile Computing*, pages 644–652, Niagara Falls, Ontario, Canada, 2010.

[PC06] Pearl Pu and Li Chen. Trust building with explanation interfaces. In *Conference on Intelligent User Interfaces*, pages 93–100, Sydney, Australia, 2006.

[Pea88] Judea Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference.* Morgan Kaufmann Publishers Inc., 1988.

[PGS98] Morgan Price, Gene Golovchinsky, and Bill N Schilit. Linking By Inking: Trailblazing in a Paper-like Hypertext. In *Conference on Hypertext and Hypermedia*, pages 30–39, Pittsburgh, PA, USA, 1998.

[PHJ02] Ivana Podnar, Manfred Hauswirth, and Mehdi Jazayeri. Mobile push: delivering content to mobile users. In *Conference on Distributed Computing Systems Workshops*, pages 563–568, Vienna, Austria, 2002.

[PLM+01] Stefan Poslad, Heimo Laamanen, Rainer Malaka, Achim Nick, Phil Buckle, and Alexander Zipf. CRUMPET: Creation of user-friendly mobile services personalised for tourism. In *Conference on 3G Mobile Communication Technologies*, London, UK, 2001.

[PM05] Bob Price and Paul R Messinger. Optimal Recommendation Sets : Covering Uncertainty over User Preferences. In *Conference on Artificial Intelligence*, pages 541–548, Pittsburgh, PA, USA, 2005.

[PP09] Kurt Partridge and Bob Price. Enhancing Mobile Recommender Systems with Activity Inference. In *Conference on User Modeling, Adaptation, and Personalization*, pages 307–318, Trento, Italy, 2009.

[PTG08] Cosimo Palmisano, Alexander Tuzhilin, and Michele Gorgoglione. Using Context to Improve Predictive Modeling of Customers in Personalization Applications. *IEEE Transactions on Knowledge and Data Engineering*, 20(11):1535–1549, 2008.

[PTG+09] Umberto Panniello, Alexander Tuzhilin, Michele Gorgoglione, Cosimo Palmisano, and Anto Pedone. Experimental comparison of pre- vs. post-filtering approaches in context-aware recommender systems. In *Conference on Recommender Systems*, pages 265–268, New York, NY, USA, 2009.

[PV09] Stefan Pfennigschmidt and Agnès Voisard. Handling Temporal Granularity in Situation Based Services. Technical report, Fraunhofer ISST, Dortmund, Germany, 2009.

[PYC06] Han-saem Park, Ji-oh Yoo, and Sung-bae Cho. A Context-Aware Music Recommendation System. In *Conference on Fuzzy Systems and Knowledge Discovery*, pages 970 – 979, Xi'an, China, 2006.

[QON07] Mohammed A Quddus, Washington Y Ochieng, and Robert B Noland. Current map-matching algorithms for transport applications: State-of-the art and future research directions. *Transportation Research Part C: Emerging Technologies*, 15(5):312–328, 2007.

[Ras83] Jens Rasmussen. Skills , Rules , and Knowledge; Signals , Signs , and Symbols , and Other Distinctions in Human Performance Models. *Systems, Man and Cybernetics*, 13(3):257–266, 1983.

[RC02] Anand Ranganathan and Roy H Campbell. Advertising in a pervasive computing environment. In *Workshop on Mobile Commerce*, pages 10–14, Atlanta, GA, USA, 2002.

[RD83] J Edward Russo and Barbara A Dosher. Strategies for multiattribute binary choice. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 9(4):676–696, 1983.

[REHK05] Christian Ress, Aria Etemed, Thomas Hochkirchen, and Detlef Kuck. Electronic Horizon - Supporting ADAS applications with predictive map data. In *ITS European Congress*, pages 1–8, Hannover, Germany, 2005.

[Rho00] Bradley J Rhodes. *Just-In-Time Information Retrieval.* Phd thesis, MIT Media Lab, 2000.

[Rho03] Bradley J Rhodes. Using physical context for just-in-time information retrieval. *IEEE Transactions on Computers*, 52(8):1011–1014, 2003.

[Ric79] Elaine Rich. User Modeling via Stereotypes. *Cognitive Science*, 3(4):329–354, 1979.

[Ric03] Debbie Richards. Knowledge-Based System Explanation: The Ripple-Down Rules Alternative. *Knowledge and Information Systems*, 5(1):2–25, 2003.

[RIS+94] Paul Resnick, Neophytos Iacovou, Mitesh Suchak, Peter Bergstrom, and John Riedl. GroupLens. In *Conference on Computer supported cooperative work*, pages 175–186, Chapel Hill, NC, USA, 1994.

[RMGG01] Thomas A Ranney, Elizabeth Mazzae, Riley Garrott, and Michael J Goodman. NHTSA driver distraction research: past, present and future. Technical report, SAE International, Warrendale, PA, USA, 2001.

[RN02] Stuart Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach.* Prentice Hall, 2 edition, 2002.

[Rob81] Stephen Robertson. The methodology of information retrieval experiment. In *Information Retrieval Experiment*, chapter 1, pages 9–31. Butterworths, 1981.

[RPM98] Nick Ryan, Jason Pascoe, and David Morse. Enhanced Reality Fieldwork: the Context-aware Archaeological Assistant. *Computer Applications in Archaeology*, 1998.

[RRFS07] Matthias Rockl, Patrick Robertson, Korbinian Frank, and Thomas Strang. An Architecture for Situation-Aware Driver Assistance Systems. In *Conference on Vehicular Technology*, pages 2555–2559, Dublin, Ireland, 2007.

[RT08] K Nageswara Rao and V G Talwar. Application Domain and Functional Classification of Recommender Systems A Survey. *DESIDOC Journal of Library and Information Technology*, 28(3):17–35, 2008.

[RV97] Paul Resnick and Hal R Varian. Recommender systems. *Communications of the ACM*, 40(3):56–58, 1997.

[RW02] Francesco Ricci and Hannes Werthner. Case base querying for travel planning recommendation. *Information Technology and Tourism*, 4(3-4):215–226, 2002.

[Saa80] Thomas L Saaty. *The Analytic Hierarchy Process.* McGraw-Hill International, 1980.

[SBB04] Inessa Seifert, Sven Bertel, and Thomas Barkowsky. Assistance for Spatio-Temporal Planning in Ubiquitous Computing Environments Based on Mental Models. In *Workshop on Artificial Intelligence in Mobile Systems*, pages 9–14, Nottingham, UK, 2004.

[SBG99] Albrecht Schmidt, Michael Beigl, and Hans-w Gellersen. There is more to Context than Location. *Computers and Graphics Journal*, 23(6):893–901, 1999.

[SBNPZ02] Barbara Schmidt-Belz, Achim Nick, Stefan Poslad, and Alexander Zipf. Personalized and Location-based Mobile Tourism Services. In *Workshop on Mobile Tourism Support Systems*, Pisa, Italy, 2002.

[SDB02] C Schmitt, D Dengler, and M Bauer. THE MAUT MACHINE : An Adaptive Recommender System. In *Workshop Adaptivität und Benutzermodellierung in interaktiven Softwaresystemen*, Hannover, Germany, 2002.

[SH04] Tobias Straub and Andreas Heinemann. An anonymous bonus point system for mobile commerce based on word-of-mouth recommendation. In *Symposium on Applied computing*, pages 766–773, Nicosia, Cyprus, 2004.

[SHBB03] John Salerno, Mike Hinman, Doug Boulware, and Paul Bello. Information Fusion for Situational Awareness. In *Conference on Information Fusion*, pages 507–513, Cairns, Queensland, Australia, 2003.

[She87] Thomas B Sheridan. Supervisory Control. In *Handbook of human factors and ergonomics*, chapter 38, pages 1243–1268. Wiley, 1987.

[Shi91] Solomon Eyal Shimony. Explanation, Irrelevance, and Statistical Independence. In *Conference on Artificial Intelligence*, pages 482–487, Anaheim, CA, USA, 1991.

[Shn93] Ben Shneiderman. Direct Manipulation. In *Sparks of innovation in human-computer interaction*, chapter 1, pages 13–62. Ablex Publishing Corporation, 1993.

[Sim55] Herbert A Simon. A Behavioral Model of Rational Choice. *The Quarterly Journal of Economics*, 69(1):99, 1955.

[Sim97] Herbert A Simon. *Models of Bounded Rationality.* MIT Press, 1997.

[SKKR01] Badrul Sarwar, George Karypis, Joseph Konstan, and John T Riedl. Item-based collaborative filtering recommendation algorithms. In *Conference on World Wide Web*, pages 285–295, Hong Kong, 2001.

[SKR01] J Ben Schafer, Joseph A Konstan, and John Riedl. E-Commerce Recommendation Applications. *Data Mining and Knowledge Discovery*, 5(1-2):115–153, 2001.

[SLGH08] Timothy Sohn, Kevin A Li, William G Griswold, and James D Hollan. A diary study of mobile information needs. In *Conference on Human factors in computing systems*, pages 433–442, Florence, Italy, 2008.

[Slo72] Paul Slovic. *From Shakespeare to Simon: speculations and some evidence about man's ability to process information.* Oregon Research Institute, 1972.

[SLP04] Thomas Strang and Claudia Linnhoff-Popien. A Context Modeling Survey. In *Workshop on Advanced Context Modelling, Reasoning and Management*, pages 31–41, Nottingham, UK, 2004.

[SLPF03] Thomas Strang, Claudia Linnhoff-Popien, and Korbinian Frank. CoOL: A Context Ontology Language to enable Contextual Interoperability. In *Conferece on Distributed Applications & Interoperable Systems*, pages 236–247, Paris, France, 2003.

[SM95] Upendra Shardanand and Pattie Maes. Social information filtering. In *Conference on Human factors in computing systems*, pages 210–217, Vancouver, Canada, 1995.

[SMK+06] Takeshi Suzuki, Kaoru Maeda, Katsumi Kanasaki, Andreas Dengel, Harald Holz, and Oleg Rostanin. TaskNavigator: Agile Workflow Management and Proactive Information Delivery. Technical Report 32, DFKI, Saarbrücken, Germany, 2006.

[SMRM04] Barry Smyth, Lorraine McGinty, James Reilly, and Kevin McCarthy. Compound Critiques for Conversational Recommender Systems. In *Conference on Web Intelligence*, pages 145–151, Beijing, China, 2004.

[SNM08] Panagiotis Symeonidis, Alexandros Nanopoulos, and Yannis Manolopoulos. Providing Justifications in Recommender Systems. *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, 38(6):1262–1272, 2008.

[SO04] Antti Salovaara and Antti Oulasvirta. Six modes of proactive resource management. In *Nordic conference on Human-computer interaction*, pages 57–60, Tampere, Finland, 2004.

[Sou91] Richard W Southwick. Explaining reasoning: an overview of explanation in knowledge-based systems. *The Knowledge Engineering Review*, 6(01):1, 1991.

[SP08] Kostas Stefanidis and Evaggelia Pitoura. Fast contextual preference scoring of database tuples. In *Conference on Extending database technology Advances in database technology*, volume pages, pages 344–355, Nantes, France, 2008.

[SPM91] William Swartout, Cecile Paris, and Johanna Moore. Explanations in knowledge systems: design for explainable expert systems. *IEEE Expert: Intelligent Systems and Their Applications*, 6(3):58–64, 1991.

[SS02] Rashmi Sinha and Kirsten Swearingen. The role of transparency in recommender systems. In *Extended Abstracts on Human Factors in Computing Systems*, pages 830–831, Minneapolis, MN, USA, 2002.

[SST93] Eldar Shafir, Itamar Simonson, and Amos Tversky. Reason-based choice. *Cognition*, 49(1-2):11–36, 1993.

[ST94] Bill N Schilit and Marvin M Theimer. Disseminating active map information to mobile hosts. *IEEE Network: The Magazine of Global Internetworking*, 8(5):22–32, 1994.

[STC05] David Shinar, Noam Tractinsky, and Richard Compton. Effects of practice, age, and task demands, on interference from a phone task while driving. *Accident Analysis and Prevention*, 37(2):315–326, 2005.

[Str07] Markus Strassberger. *Kontextbereitstellung in Automobilen Ad-hoc Netzen*. Phd thesis, Ludwig-Maximilians-Universität München, 2007.

[Sue92] Henri J Suermondt. *Explanation in Bayesian Belief Networks*. Phd thesis, Stanford University, 1992.

[Sug85] Michio Sugeno. *Industrial Applications of Fuzzy Control*. Elsevier Science Inc, 1985.

[SUPOK08] Somkiat Sae-Ueng, Sineenard Pinyapong, Akihiro Ogino, and Toshikazu Kato. Personalized Shopping Assistance Service at Ubiquitous Shop Space. In *Conference on Advanced Information Networking and Applications - Workshops*, pages 838–843, Okinawa, Japan, 2008.

[SVV99] Cheri Speier, Joseph S Valacich, and Iris Vessey. The Influence of Task Interruption on Individual Decision Making: An Information Overload Perspective. *Decision Sciences*, 30(2):337–360, 1999.

[SY03] Shashi Shekhar and Jin Soung Yoo. Processing in-route nearest neighbor queries. In *Symposium on Advances in geographic information systems*, pages 9–16, New Orleans, LA, USA, 2003.

[Tas05] Leo Tasca. Driver distraction: towards a working definition. In *Conference on Distracted Driving*, Toronto, Ontario, 2005.

[TBK06] Marcus Tonnis, Verena Broy, and Gudrun Klinker. A Survey of Challenges Related to the Design of 3D User Interfaces for Car Drivers. In *Symposium on 3D User Interfaces*, pages 127–134, Essen, Germany, 2006.

[Ten00] David Tennenhouse. Proactive computing. *Communications of the ACM*, 43(5):43–50, 2000.

[TGC$^+$09] Justin Tang, Nate Gertsch, Hye Jung Choi, Alfred Kobsa, and Shidan Habibi. Ad As You Go: A Study of Ad Placement on Personal Navigation Devices. In *Workshop on Personalization in Mobile and Pervasive Computing*, Trento, Italy, 2009.

[TGSR07] Amit Thawani, Srividya Gopalan, V. Sridhar, and Krithi Ramamritham. Context-aware Timely Information Delivery in Mobile Environments. *The Computer Journal*, 50(4):460–472, 2007.

[THG10] Stéphane Turlier, Clemens Hahn, and Sascha Gebhardt. Browsing online music catalogs in a vehicle. In *Workshop on Mobile cloud media computing*, pages 53–58, Firenze, Italy, 2010.

[TM05] Loren Terveen and David W McDonald. Social matching. *ACM Transactions on Computer-Human Interaction*, 12(3):401–434, 2005.

[TM11] Nava Tintarev and Judith Masthoff. Designing and Evaluating Explanations for Recommender Systems. In *Recommender Systems Handbook: A Complete Guide for Research Scientists and Practitioners*, chapter 15, pages 479 – 510. Springer, 2011.

[Tou03] Stephen Edelston Toulmin. *The uses of argument.* Cambridge University Press, 2003.

[TPG98] Louis Tijerina, Edwin Parmer, and Michael J Goodman. Driver workload assessment of route guidance system destination entry while driving: A test track study. In *ITS World Congress*, Seoul, Korea, 1998.

[TQ00] Brendon Towle and Clark Quinn. Knowledge Based Recommender Systems Using Explicit. In *AAAI Workshop (AAAI Technical Report WS-0004)*, pages 74–77, Menlo Park, CA, USA, 2000.

[Tre11] Phillip Tretten. Placement of information in the car's windshield investigated, 2011.

[Tsa93] Edward Tsang. *Foundations of Constraint Satisfaction.* Academic Press, 1993.

[Tve69] Amos Tversky. Intransitivity of preferences. *Psychological Review*, 76(1):31–48, 1969.

[Tve72] Amos Tversky. Elimination by aspects: A theory of choice. *Psychological Review*, 79(4):281–299, 1972.

[VBB09] Neven Vrcek, Goran Bubas, and Neven Bosilj. User Acceptance of Location-based Services. *izlazi u samo elektroničkom izdanju*, pages 152–157, 2009.

[vdHKK05] Hans van der Heijden, Gabriele Kotsis, and Reinhard Kronsteiner. Mobile recommendation systems for decision making "on the go". In *Conference on Mobile Business*, pages 137–143, Sydney, Australia, 2005.

[vEGH96] Frans H van Eemeren, Rob Grootendorst, and Snoeck Henkemans. *Fundamentals of argumentation theory: a handbook of historical backgrounds and contemporary developments*. Lawrence Erlbaum Associates, 1996.

[VHE05] Trent W Victor, J L Harbluk, and Johan A Engström. Sensitivity of eye-movement measures to in-vehicle task difficulty. *Transportation Research Part F Traffic Psychology and Behaviour*, 8(2):167–190, 2005.

[VK09] Muthuswamy Vijayalakshmi and Arputharaj Kannan. Proactive location-based context aware services using agents. *International Journal of Mobile Communications*, 7(2):232, 2009.

[vSPK04] Mark van Setten, Stanislav Pokraev, and Johan Koolwaaij. Context-aware recommendations in the mobile tourist application COMPASS. In *Conference on Adaptive Hypermedia*, pages 235–244, Eindhoven, Netherlands, 2004.

[VSR08] Jesse Vig, Shilad Sen, and John Riedl. Tagsplanations. In *Conference on Intelligent user interfaces*, pages 47–56, Sanibel Island, FL, USA, 2008.

[WBE08] Wolfgang Wörndl, Michele Brocco, and Robert Eigner. A Context-Aware Gas Station Recommender System for Vehicular Ad-Hoc Networks. In *Conference on Wireless Applications and Computing Conference*, pages 101–108, Amsterdam, Netherlands, 2008.

[WG07] Wolfgang Wörndl and Georg Groh. Utilizing Physical and Social Context to Improve Recommender Systems. In *Conferences on Web Intelligence and Intelligent Agent Technology*, pages 123–128, Silicon Valley, CA, USA, 2007.

[WHBGV11] Wolfgang Wörndl, Johannes Hübner, Roland Bader, and Daniel Gallego-Vico. A model for proactivity in mobile, context-aware recommender systems. In *Conference on Recommender systems*, pages 273–276, Chicago, IL, USA, 2011.

[Wic84] Christopher D Wickens. *Engineering Psychology and Human Performance*. Addison-Wesley, 1984.

[Win01] Terry Winograd. Architectures for Context. *Human-Computer Interaction*, 16(2):401–419, 2001.

[WPT02] Roy Want, Trevor Pering, and David Tennenhouse. Comparing Autonomic & Proactive Computing. *IBM Systems Journal*, 42(1):129–135, 2002.

[WSW07] Wolfgang Wörndl, Christian Schüller, and Rolf Wojtech. A Hybrid Recommender System for Context-aware Recommendations of Mobile Applications. In *Conference on Data Engineering Workshop*, pages 871–878, Istanbul, Turkey, 2007.

[YF02] Yunwen Ye and Gerhard Fischer. Supporting reuse by delivering task-relevant and personalized information. In *Conference on Software engineering*, pages 513–523, Orlando, FL, USA, 2002.

[YLB05] Guosheng Yang, Yingzi Lin, and Prabir Bhattacharya. A Driver Fatigue Recognition Model Using Fusion of Multiple Features. In *Conference on Systems, Man and Cybernetics*, volume 2, pages 1777–1784, Big Island, HI, USA, 2005.

[YMT+09] Ji Hyun Yang, Zhi-hong Mao, Louis Tijerina, Tom Pilutti, Joseph F Coughlin, and Eric Feron. Detection of Driver Fatigue Caused by Sleep Deprivation. *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, 39(4):694–705, 2009.

[YR03] Kristie Young and Michael Regan. Driver distraction: A review of the literature. Technical report, Monash University Accident Research Centre, Melbourne, Australia, 2003.

[YSSMM09] Neil Yorke-Smith, Shahin Saadati, Karen L Myers, and David N Morley. Like an Intuitive and Courteous Butler : A Proactive Personal Agent for Task Management. In *Conference on Autonomous Agents and Multiagent Systems*, pages 337–344, Budapest, Hungary, 2009.

[YTP05] Ghim-Eng Yap, Ah-Hwee Tan, and Hwee-Hwa Pang. Dynamically-optimized context in recommender systems. In *Conference on Mobile data management*, pages 265–272, Ayia Napa, Cyprus, 2005.

[YTP07] Ghim-Eng Yap, Ah-Hwee Tan, and Hwee-Hwa Pang. Discovering and Exploiting Causal Dependencies for Robust Mobile Context-Aware Recommenders. *IEEE Transactions on Knowledge and Data Engineering*, 19(7):977–992, 2007.

[YY10] Kam Fung Yeung and Yanyan Yang. A Proactive Personalized Mobile News Recommendation System. *Developments in E-systems Engineering*, pages 207–212, 2010.

[YZZ+06] Zhiwen Yu, Xingshe Zhou, Daqing Zhang, Chung-Yau Chin, Xiaohang Wang, and Ji Men. Supporting Context-Aware Media Recommendations for Smart Phones. *IEEE Pervasive Computing*, 5(3):68–75, 2006.

[Zad65] Lotfi A Zadeh. Fuzzy sets. *Information and Control*, 8(3):338–353, 1965.

[Zim03] Andreas Zimmermann. Context-Awareness in User Modelling: Requirements Analysis for a Case-Based Reasoning Application. In *Conference on Case-based reasoning: Research and Development*, pages 718–732, Trondheim, Norway, 2003.

[Zip49] George K Zipf. *Human Behavior and the Principle of Least Effort*. Addison-Wesley Press, 1949.

[ZN10] Markus Zanker and Daniel Ninaus. Knowledgeable Explanations for Recommender Systems. In *Conference on Web Intelligence and Intelligent Agent Technology*, pages 657–660, Toronto, Canada, 2010.

[ZZXM09] Yu Zheng, Lizhu Zhang, Xing Xie, and Wei-Ying Ma. Mining interesting locations and travel sequences from GPS trajectories. In *Conference on World wide web*, pages 791–800, Madrid, Spain, 2009.