TECHNISCHE UNIVERSITÄT MÜNCHEN
Lehrstuhl für Informatik VII

Latent-Class Statistical Relational Learning with Uncertain Formal Ontologies

Achim B. Rettinger

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. Francisco Javier Esparza Estaun
Prüfer der Dissertation:	1. Univ.-Prof. Michael Beetz, Ph.D. 2. Univ.-Prof. Dr. Stefan Kramer

Die Dissertation wurde am 31.03.2010 bei der Technischen Universität München eingereicht und durch die Fakultät für Informatik am 19.11.2010 angenommen.

Abstract

Ontologies are a well researched formalism in computer science to represent knowledge in a machine processable manner. Recently, with the growing semantic web and social networks, ontologies with large amounts of data are becoming available. While such knowledge representations are intended for data integration and deduction of implicit knowledge, they are in general not designed to induce uncertain knowledge.

This thesis focuses on the statistical analysis of known as well as deducible facts in ontologies and the induction of uncertain knowledge from both. Hereby, the uncertainty of induced knowledge originates not only from a lack of information but also from potentially untrustworthy sources providing the facts.

We outline common ontology languages, their (formal) semantics and expressivity from a perspective of existing real world data sources. Then, existing learning methods are discussed that could be used for automated statistical analysis of facts given in such ontologies and the induction of facts that are not explicitly given in the ontology. Hereby, two fundamental approaches to artificial intelligence need to be combined: The deductive reasoning by logical consequence and the inductive learning by statistical generalization.

The main contribution of this work is a machine learning approach that enforces hard constraints during the learning phase. In short, this is achieved by checking description logic satisfiability of statements inductively inferred by an infinite latent-class multi-relational Bayesian learning method based on Dirichlet process mixture models. This guarantees the compliance of induced facts with deductive arguments and can lead to an improved cluster analysis and predictive performance. To demonstrate the effectiveness of our approach we provide experiments using real world social network data in form of an OWL DL ontology.

In addition, the learning from untrustworthy facts in formal knowledge representations is discussed. We show how to model and learn context-sensitive trust using our learning method. The efficiency and performance of our approach is evaluated empirically e. g. , with user-ratings gathered from online auctions.

In summary, this thesis contributes to the combination of inductive and deductive reasoning approaches and more specifically to latent-class statistical learning with description logic ontologies containing information from potentially untrustworthy sources.

Acknowledgments

I owe the successful completion of my thesis to many people. Despite inconstant external circumstances they constantly supported and inspired me. First and foremost, this applies to Volker Tresp. Without his intense joint work this would not have been possible. Equally important were the collaboration and fruitful discussions with Matthias Nickles which resulted in key scientific contributions. In addition, I would like to express my deep gratitude to Michael Beetz and Stefan Kramer who offered the flexibility, cooperativeness and expertise crucial to bringing it to a successful conclusion. Another key role play Wilfried Brauer and Gerhard Weiss: Thank you for opening up this great opportunity in the first place. I also appreciate the friendship, contribution and support from many colleagues. Apologizing for the incompleteness of this list in advance I thank Zhao Xu, Yi Huang, Markus Bundschus, Angelika Först, Joschka Bödecker, Stefan Kugele and many more.

My special thanks go to all institutions and their affiliated people who provided a great working environment. This includes the Technische Universität München, Siemens AG (CT, Munich), University of Bath (UK) and Osaka University (Japan). I also want to thank the Siemens AG, Deutsche Forschungsgemeinschaft (DFG), Deutscher Akademischer Austausch Dienst (DAAD) and Japan Society for the Promotion of Science (JSPS) which all provided essential financial and organizational support.

However, the most essential share in this achievement have my wife, family and friends. They are the foundation of my life and are the ones who make a research career possible: Thank you for your love and support.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation and Overview

Many recent developments in the area of information technologies show a slow but continuous shift from plain unstructured data sources like text, images videos and sound, towards knowledge that can be processed by computers in a more intelligent way. While there already have been longterm research efforts in areas of computer science and mathematics like *Artificial Intelligence* (AI), formal logic, statistics and databases to make computers more powerful data processing systems, only recently commercial prototypes and large data sources have become available that are targeted at deploying those technologies in the real world.

At the core of those systems is a *semantic* or for computers more 'meaningful' data representation that associates instances to entity classes and their interdependencies to relations. The illustrating example we focus on in this thesis are social networks of people which are represented in a semantic and formal (e. g. , graph based) manner. Such social networks have become ubiquitous on the internet since it has become a platform for intense communication and social interactions, often referred to as the *web 2.0* or the *social web*.

See Fig. 1.1 for the following simple introductory example: There are concrete instances, like *tim*, *tom* or *tina* (depicted as circles), that are known to be members of the abstract entity class *Person* (rectangle). A second class of entities describes *Age-groups* of *Persons* (see relation *is_in*). Furthermore, it is known, that in general people might know each other (labeled line *knows*) and for some instances it is also given that this is actually the case (unlabeled lines, see e. g. , the line between *tim* and *16-30*, indicating that *tim* is known to be in *Age-group 16-30*).

Such symbolic knowledge representations are typically called *ontologies* in computer science. The common usage of *semantic* refers to the property of such systems that computers can assign an abstract symbol like *Person* to one concrete instance like *tim*. In this sense the computer has a meaningful interpretation of the data e. g. , it 'understands' that *tim* is a person and that *Persons* can *know* each other and can be assigned to certain *Age-groups*.

Traditionally, ontologies in AI are primarily applied to problems, like data integration and logical deduction of implicit knowledge. For instance, if ad-

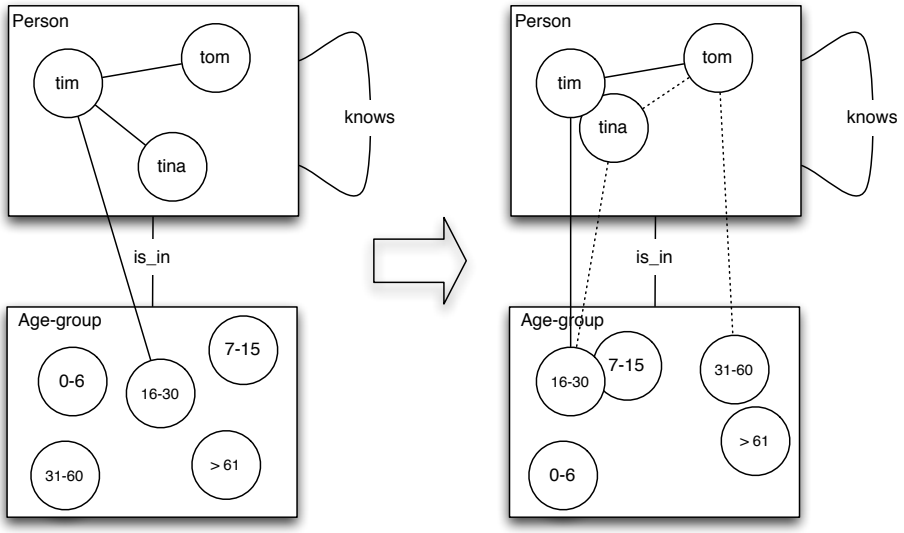

Figure 1.1: Simple example of a social network (left) and the additional knowledge to be gained by statistical inference (right). Lines indicate known facts, dashed lines indicate uncertain links between individuals and circles that appear closer to each other indicate more similar individuals.

ditional meta information is added, like rules or constraints stating that all *Students* are *Persons* and cannot be in *Age-group 0-6*, the computer can deduce that a given instance of a *Student* is also a *Person* and thus *is_in* a specific *Age-group* except in *0-6*. Hereby, they provide a big potential for more concise data representations and more intelligent information processing.

However, so far, ontologies are in general not designed to handle or even induce uncertain knowledge. Usually, only a small number of possible statements is known to be true or can be inferred to be true by logical reasoning. Here, statistical machine learning can be used to estimated the truth values of additional statements by exploring regularities in the semantic data.

This thesis focuses on this task by investigating the statistical analysis of known facts in ontologies and the induction of uncertain knowledge from those known facts resulting in what we then call *uncertain ontologies*. Hereby one of our contributions is that the extracted information should not violate constraints defined in such ontologies. In addition, we consider that the uncertainty of induced knowledge originates not only from a lack of information but also from potentially untrustworthy sources providing the original facts.

The additional knowledge gained for *uncertain ontologies* can for instance be used to predict properties and relations of instances (see dashed lines in Fig. 1.1) and delivers valuable insight into the data, like clusters of similar instances (depicted as nearby circles). Referring to the example of social networks one could e. g. , recommend potentially new friends to a specific person, predict their age or trustworthiness or group similar persons. For instance, it might be possible to assess the probability that *tina is_in Age-group 16-30* and *knows tom* (cmp. Fig. 1.1).

1.1.1 Chapter Overview

The thesis is structured as follows:

1. This introductory chapter (Ch. 1) is intended to provide an intuitive, high-level and informal introduction to the subject of this thesis for non-experts. First, we point out recent trends that show promising developments in the area of semantic computing (Sec. 1.2). Next, popular formats of semantic data representations and existing data collections are presented (Sec. 1.2.1) and some applications are outlined that are based on semantic data representations (Sec. 1.2.2). Then, the task covered in this thesis, namely Machine Learning (ML), is mentioned in relation to semantic data (Sec. 1.3). Finally, an informal definition of the research challenge covered in this thesis as well as the main contributions are briefly presented (Sec. 1.4).

2. Ch. 2 focuses on the source of semantic data that can be used for ML. After a short introduction (Sec. 2.1), we gradually replace the vague term 'semantic' by a formal definition. This is done by introducing different levels of knowledge representations, following a data-driven perspective of existing large scale real world data sources. We start from semi-formal representations and continue towards expressive formal ontologies (Sec. 2.2). Then, the focus will be on concrete ontology language on the Semantic Web (SW) (Sec. 2.3). A formal definition of the OWL DL ontology language syntax and semantic as well as basic inference problems are presented (Sec. 2.3.1 and 2.3.2). This chapter concludes with probabilistic extensions to semantic web languages (Sec. 2.3.4).

3. After introducing formal knowledge representations in the previous chapter, Ch. 3 focusses on the second theoretical background needed for the proposed approach, namely *Statistical (Multi-)Relational Learning* (SRL). As in the chapter before the least complex approaches will be introduced first (Sec. 3.1), adding more complex data representation in each section (Sec. 3.1 - 3.4). The chapter concludes with SRL algorithms that are based on expressive logical representations (Sec. 3.4).

4. Ch. 4 presents the main contribution of this thesis, the Infinite Hidden Semantic Model (IHSM). After discussing related work (Sec. 4.1) we define the framework for the formal knowledge representation in form of an description logic ontology (Sec. 4.2) and the Bayesian learning model in form of a Dirichlet process mixture model (Sec. 4.3). Next, we will show how we can learn from facts given in expressive formal ontologies while preserving the consistency of such knowledge bases (Sec. 4.4). To demonstrate the feasibility of our approach we provide an empirical analysis using a real world social network besides other data sets (Sec. 4.5). First, we show how the performance with inconsistent predictions compares to other inconsistent approaches. Next, we evaluate the differences to that when keeping the knowledge base in form of a description logic ontology consistent during inference.

5. Ch. 5 covers the second major contribution of this thesis, the Infinite Hidden Semantic Trust Model (IHSTM). It addresses the problem of handling

ontologies containing facts from untrustworthy sources (Sec. 5.1). We propose a way how such trust situations can be modeled using IHSM and how trust based on past observations and context information can be learned and integrated into the ontology(Sec. 5.2). The performance, practicability and efficiency of our approach is evaluated empirically (Sec. 5.4) on synthetic data, user-ratings gathered from eBay and on negotiation data (Sec. 5.3).

6. The last chapter (Ch. 6) concludes by summarizing the contributions of this thesis (Sec. 6.1) and mentioning future work (Sec. 6.2).

1.1.2 Published Scientific Contributions

Much of the material in this thesis has already been published, mostly in conference proceedings. The main contribution of Ch. 4 - the Infinite Hidden Semantic Model (IHSM) - was presented at ECML 2009 [Rettinger et al., 2009]. Own preliminary work related to IHSM is [Rettinger and Nickles, 2009, Xu et al., 2009]. The main contribution of Ch. 5 - the Infinite Hidden Semantic Trust Model (IHSTM) - was presented at AAMAS 2008 [Rettinger et al., 2008]. Own preliminary work related to trust learning is [Rettinger et al., 2007, Först et al., 2009].

Novel work contained in this thesis that is not published elsewhere mainly comprises the hierarchies of knowledge representations in Sec. 2.2 and the hierarchies of ML algorithms in Ch. 3. Own work that is loosely related to those topics was published in [Rettinger et al., 2006, Bundschus et al., 2009, Tresp et al., 2008, Huang et al., 2009]

1.2 Semantic Computing

The term *semantic* is being used heavily in information technologies in recent years. It is associated with new trends and technologies like tagging, the semantic web, web 3.0 or web services and sometimes even considered a *semantic revolution* [Syeda-Mahmood et al., 2006]. As common with such buzzword, like web 2.0 before, there is no consistent definition of the meaning of 'semantic' underlying the different usages.

Observing the informal meaning of *semantic* from a distance all such systems have the idea of an additional abstract data representation in common. This semantic meta-layer is added on top of the raw input signal that usually consists of multimedia data like text and speech in natural language, images, videos and so on. It is supposed to provide the meaning of the content of the data to computers and allow for more intelligent information processing systems.

Semantic technologies are intended to solve some key problems that current information systems like the world wide web are faced with: A computer, for example, cannot 'know' that a webpage of a specific person is actually a description of a human being which has certain properties like a date of birth and a name. Thus, it cannot compute a single clear answer to a question like "How old would Ghandi be today?". In theory, semantic technologies might be able to answer such queries, making current information systems much more useful to other computers as well as to human users.

1.2.1 Semantic Data

In principle, all software systems use semantic data representations. An operating system, for instance, using the desktop metaphor like Microsoft Windows[1], 'knows' that there are folders which can have names and subfolders and store information by topic. However, most of the information available in electronic form is not accessible by computers.

The driving force behind many semantic technologies is the vision of a Semantic Web (SW). The idea is to enrich the information on the internet with a semantic, machine readable description. In many cases the information on the web already is available in a structured form. For instance, entering text into fields, ticking boxes or choosing an entry from a drop down list already enables the computer to make meaningful associations: Entering "tom" into a name-textfield of a user profile can make the computer know that there is a *Person* with the *Name* "tom". This way, e. g. , eBay[2] products are categorized into a given hierarchy and associated with predefined properties. Wikipedia[3] also categorizes entries and provides basic property boxes for each category.

Besides this implicit structuring of web pages various formalisms have been proposed how websites can be semantically annotated and enhanced with metadata. Newer versions of plain HTML like HTML5[4] introduce a number of more semantic replacements for common uses of generic blocks like `<div>` and inline `<span>` elements with `<article>`, `<nav>` or `<footer>`.

A more flexible approach has been achieved by *tagging*. Here, unstructured resources are associated with single phrases, called tags, which are supposed to associate meaning.

Another approach to semantic markup are microformats[5]. Microformats seek to re-use existing XHTML and HTML tags to convey metadata. Current microformats allow the encoding and extraction of events, contact information, social relationships and so on. New versions of web browsers are expected to include native support for microformats. Microformats have been developed for particular types of information, like hAtom for marking up Atom feeds from within standard HTML or hCalendar for events.

A step towards the technologies proposed for the SW is RDFa[6] or embedded RDF. RDFa is a W3C Recommendation that adds a set of attribute level extensions to XHTML for embedding rich metadata within Web documents.

Another field where semantic modeling plays a key role is the area of databases and software engineering. Software specification methodologies such as UML and entity relationship models allow to model expert knowledge about the application domain in a semi-formal but semantic way. Consequently, the largest existing amount of data represented in structured form is stored in databases. For instance a relational database consists of tables grouping common characteristics found in data sets, like persons and their properties. Due to this structuring of the data a semantic is implied although it is intended to ease human access rather than one by computers.

[1] http://www.microsoft.com/Windows/
[2] http://www.ebay.com/
[3] http://www.wikipedia.org/
[4] http://dev.w3.org/html5/spec/Overview.html
[5] http://microformats.org/
[6] http://www.w3.org/TR/xhtml-rdfa-primer/

The first large collaborative data set on the SW is called *linked data*[7] and originates from such curated databases (cmp. Sec. 2.2.2) that are linked together. This data set might become the first success story of large scale semantic technologies. It has already resulted in a huge amount of data that has become available in a semantic data format. However, if it can be used for any convincing applications is not clear yet.

Another already existing semantic data store which is more centralized is twine[8]. Twine is a web service for information storage, authoring and discovery. It combines features of forums, wikis, online databases and newsgroups and employs intelligent software to automatically mine and store data relationships expressed with RDF statements. Another approach to a collaborative semantic knowledge base offers Freebase[9]. It is an online collection of structured data harvested from many sources. Another approach to the collection of semantic data by a commercial company is for instance Powerset[10].

The example of linked data shows the need to share a common meaning of the concepts and relationships that are present in the data. This distributed approach demands the use of ontologies which are expressed in formal languages with a well-defined semantic. The most simple semantic data representations that are in wide use are glossaries or controlled vocabularies which do not use a shared formal semantic.

The simplest representations that are in use - e. g. , in linked data - and provide formal semantics are so called lightweight ontologies (see Sec. 2.2). They make a distinction between classes, instances and properties and contain at least a hierarchy of concepts (called a taxonomy). In this sense, the use of small, lightweight ontologies that are linked already has become reality. However, the expressivity for automated processing of lightweight ontologies is still very low and the question of usefulness is still open.

1.2.2 Applications of Semantic Computing

With semantic data becoming more available in recent years the interest in applications that are using semantic technologies is growing . Academic institutions have been working on semantic tools for several decades. Only recently, big companies have shown more initiative in this field which indicates that the belief in mass and commercial application of such technologies is growing. In the following we list a few semantic applications that have ben published by companies in the recent years.

Maybe the most prominent and most impressive application of semantic technologies is provided by WolframAlpha[11]. WolframAlpha describes itself as a computational knowledge engine which is developed by Wolfram Research. It is an online service that answers factual queries directly by computing the answer from structured data, rather than providing a list of documents or web pages that might contain the answer as a search engine would. It is in use as an extension to traditional search engines like Bing[12] by delivering computa-

[7] http://linkeddata.org/
[8] http://www.twine.com/
[9] http://www.freebase.com/
[10] http://www.powerset.com/
[11] http://www.wolframalpha.com/
[12] http://www.bing.com/

tional answers. Thus, it already shows capabilities which might be in use on the semantic web in the future. However, WolframAlpha is limited to rather technical domains and does not use open and distributed data sources. In this respect it shows many parallels with Cyc[13], a project aimed at developing a common-sense inference engine since the 1980s.

Another commercial application of semantic technologies is *Yahoo! SearchMonkey*[14]. It allows web site owners to use structured data to make the *Yahoo! search engine*[15] results more useful and visually appealing to the end user, and drive more relevant traffic to their sites. Semantic data can be placed on the web site using microformats which will be gathered by Yahoo!'s crawlers and made available to developers. The developers in turn can decide how this semantic data is used to enhance search results. Thus, web developers have more control how their site is presented in the search results.

Google[16] also offers applications which use semantic technologies. *Google Squared*[17] is a semantic search product which extracts structured data from the web and presents its results in a table-like format. Each search query returns a table for an entity which is the result of the search, accompanied by its specific set of attributes that are associated with the topic of a search. The data representation generated by Google Squared has simple semantics and might potentially be used for other semantic applications.

Many more semantic applications have been or are being developed. So far, most of them fulfill the purpose of browsers, search engines and indexer for semantic data. As semantic data is meaningful for humans and computers alike those applications can either be human oriented or application oriented. WeFind[18], SemaPlorer[19] or Yago-Naga[20] are just a few more interesting applications that can be mentioned in this respect.

Faceted search[21] is one of the examples of a step towards human oriented search for semantic data. The web search world has offered two paradigms so far. *Navigational search* uses a hierarchy structure (taxonomy) to enable users to browse the information space by iteratively narrowing the scope of their quest in a predetermined order, as exemplified by Yahoo! Directory or DMOZ. *Direct search* allows users to simply write their queries as key words in a text box. This approach has been made enormously popular by Web search engines, such as Google and Yahoo! Search.

Over the last few years, the direct search paradigm has gained dominance and the navigational approach became less and less popular. Recently, a new approach has emerged, combining both paradigms, namely faceted search. Faceted search enables users to navigate a multi-dimensional information space by combining text search with a progressive narrowing of choices in each dimension. It has become the prevailing user interaction mechanism in e-commerce sites and is being extended to deal with semi-structured data, continuous dimensions,

[13] http://www.cyc.com/
[14] http://developer.yahoo.com/searchmonkey/
[15] http://www.yahoo.com/
[16] http://www.google.com/
[17] http://www.google.com/squared
[18] http://www.wefind.de/
[19] http://btc.isweb.uni-koblenz.de/
[20] http://www.mpi-inf.mpg.de/yago-naga/
[21] http://www.gfacet.org/

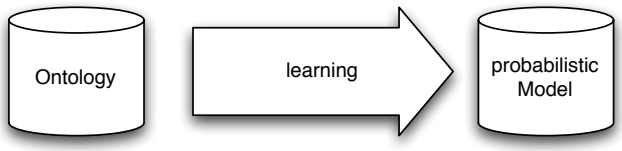

Figure 1.2: Learning with ontologies: basic setup

and folksonomies. For instance, Neofonie[22] offers a faceted search interface for Wikipedia[23]. For that DBpedia[24], a semantic representation extracted from Wikipedia and part of the linked data cloud, is used as the knowledge base.

All things considered, the trend towards applications that deploy semantic technologies or use semantic data is increasing rapidly and has reached a point where it influences large scale systems.

1.3 Machine Learning with Semantic Data

The applications mentioned in the last section are examples for what can be done with semantic data. An interesting technique that is commonly not used in those applications but has great potential is statistical Machine Learning (ML). Computers generating hypothesis from data is, in general, called ML [Hastie et al., 2001].

Semantic technologies are originally designed for logical inference not statistical analysis. If ML is applied to ontologies it is usually called ontology learning [Maedche and Staab, 2004]. The purpose of ontology learning is the extraction of semantic knowledge representations from unstructured data formats like text or images.

However, semantic data also opens up new possibilities and challenges for data mining and machine learning. Unlike ontology learning we are not concerned with the extraction, but with using this semantic knowledge to learn more, not explicitly given information. Consequently, it is about learning *with* or *from* an ontology, not learning *of* an ontology. Learning *of* ontologies can rather be seen as a preprocessing step for learning *with* ontologies.

In semantic data usually only a small number of interesting facts is known to be true or can be inferred to be true by logical reasoning. For instance, in a social network people might specify not all people they know and it is not universally true that 'someone also knows the friends of his friends'. However, such truth values of potential statements can be estimated by exploring regularities in the semantic data by using machine learning. This is the focus of this thesis: How can machines infer the probability of facts not given explicitly in or not deducable from semantic data representations (see Fig. 1.2)?

Semantic representations offer computers easy access to the data for ML tasks. Without the knowledge of the structure of the data a computer would not be capable of directly generalizing from known facts. In semantic data the structure is given to varying degrees, making it easier to apply it to ML

[22] http://www.neofonie.de/
[23] http://dbpedia.neofonie.de/browse/
[24] http://dbpedia.org/

techniques. However, traditional ML algorithms require a very specific input format. The representation of existing semantic data is more versatile and in many cases more expressive than what can be handled with standard ML algorithms.

Thus, two major challenges remain for the application of machine learning to semantic data. First, the specific properties of semantic data, like complex statistical dependencies, expressive semantic constructs, sparse data and the open world assumption[25] make semantic data not well suited for most existing ML algorithms. Second, the expected size of future semantic data sets and the noisiness of information due to conflicting and untrustworthy sources will be a challenge in regard to scalability and performance. However, the strength of statistical techniques compared to other approaches actually is its applicability to problems of the second kind.

This thesis makes contributions to both challenges. On the one hand, we focus on ML from semantic data of different levels of formality and expressivity, particularly on highly-expressive ontologies. On the other hand, we examine uncertain ontologies containing facts from different potentially untrustworthy information providers. In doing so, this thesis advances the applicability of ML to the vast amounts of semantic data currently becoming available. Looking at the bigger picture, it contributes to realizing the long term goal of AI to implement a large scale application of automated reasoning grounded on real world observations.

1.4 Problem Definition and Contributions

After outlining the setting we conclude this introduction by providing a concise - yet informal - specification of the tasks covered in this thesis:

> Given a knowledge representation in formal logic infer uncertain implicit knowledge by statistical inductive inference, while
> i) staying consistent with the explicit knowledge as well as the implicit knowledge that can be deduced by logical inference and
> ii) learn to assess the trustworthiness of statements within the explicitly given knowledge.

The overall contribution of this thesis is the development of a probabilistic framework that can infer new statements by means of a statistical analysis of the formal knowledge base. The framework is able to analyze similarities of entities and relationships between entities specified in the knowledge base and therewith provides a meaningful assessment of the instance stored in the knowledge base.

Concerning task i) this framework can guarantee that the statistical analysis performed does not result in an inconsistent knowledge base, which in addition can lead to an improved predictive performance. Concerning task ii) the framework shows an improved learning speed and improved predictive performance in assessing the trustworthiness of statements from uncertain sources.

[25] See the later chapters for an explanation of those terms.

Chapter 2

From Metadata to Formal Ontologies

The first chapter presented some of the recent trends towards semantic computing in applied research as well as in commercial applications. In this chapter, we will replace the vague term *semantic* by a formal definition (see Sec. 2.2.2). This is done stepwise by introducing different levels of knowledge representations, starting from semi-formal representations towards expressive formal ontologies (see Sec. 2.2). Formal ontologies can be seen as the theoretical background to most semantic technologies.

After giving an outline of common components in ontologies (see Sec. 2.1) and a hierarchy of knowledge representations with large scale real world applications (see Sec. 2.2), the focus will be on semantic web ontology languages (see Sec. 2.3) specifically OWL DL (see Sec. 2.3.2). OWL DL is based on Description Logic (DL) and has seen increasing interest due to the Semantic Web (SW). A formal definition of the OWL DL syntax and semantics as well as basic inference problems are presented (see Sec. 2.3.3) before we conclude with probabilistic extensions to semantic web languages (see Sec. 2.3.4).

2.1 Ontology Basics and History

In common usage there is no clear distinction between terms like knowledge base, semantic data representation or ontology. What all of those systems have in common is that they adopt some sort of (semi-)formal abstract/meta representation of knowledge as opposed to explicitly storing raw data and concrete observations only.

Here, we will concentrate on the term *ontology* as used in Artificial Intelligence (AI) for a formal representation of a set of concepts within a domain and the relationships and interdependencies between those concepts. Formal ontologies enable the manipulation of and reasoning with knowledge using formal logic. The semantics of a formal representation refers to the fact that the meaning is unambiguous, which makes the information machine processable (cmp. Sec. 2.2.2).

Originally, the term ontology was developed in *philosophy* referring to a particular system of categories which are accounting for a specific vision of the

world. Since the early 1960s ontologies were developed in AI to store information in a way that a machine could use it to answer questions. In the process the answer should not only result from an exact repetition of previously inserted knowledge, but additional facts should be deducible from the given facts. Thus, the use in AI refers to an engineering artifact describing a specific reality by defining a vocabulary and the meaning of the terms in the vocabulary plus inference rules. For that, an internal representation based on logical languages was developed.

An example for one of the first "semantic representations" is SIR (Semantic Information Retrieval) [Raphael, 1964]. The sentences that SIR could deal with involved entities and relations among these entities.

Most of the early semantic representations concentrated on taxonomies rather than arbitrary relations with individual semantic. Although we understand taxonomies best by thinking about them in the form of trees, a collection of special data structures is used when encoding them for computers. These structures are originally called *frames*, termed by Marvin Minsky [Minsky, 1974]. Typically, there would be a frame for each class of individuals or entities in a taxonomy as well as for each of the instances themselves. Frames for classes would name the superclass to which it belonged and the subclasses belonging to it. The frame would also specify properties of the entities belonging to the class.

Frame-based knowledge representation systems like KL-ONE [Brachman and Schmolze, 1985] later became the basis for the knowledge representation chosen in this thesis, namely *description logic* (DL) (see Sec. 2.3.2). Like most knowledge representations DL is mostly a subset of first-order logic allowing the use of reasoning and inference mechanisms to deduce implicit knowledge.

Table 2.1 lists terms used by different research communities for related concepts of components of knowledge representations along with an informal description from the perspective of machine learning. We are aware that this abbreviated format oversimplifies and overgeneralizes most terms. However we think that this nevertheless helps to make the topic more accessible. For a precise and formal definition of some terms see Sec. 2.2.2.

2.2 Towards Expressive Formal Ontologies

Ontologies are often equated with class definitions, taxonomic hierarchies of classes, and the subsumption relation, but ontologies are not limited to these forms. Ontologies - as used here - are also not limited to conservative definitions that only introduce terminological knowledge and do not add any observed instance knowledge about the world. The other extreme are ontologies that are in use and are heavily populated but lack formal semantic. In this thesis we are focusing on ontologies that offer both, a large number of instances and expressive formal semantic.

To measure the complexity of a concrete ontology in the real world we proposed an allocation along two dimensions:

Computational expressivity of an ontological language used in the ontology determines the possible complexity of components and axioms imposing restrictions on possible worlds. We use the term computational expressiv-

Term	Description
individual, *instance*, *fact*, *statement*, *ground(ed) atom*	real world evidence or observation; measurable or countable concrete event or entity
relation, *predicate*, *property*, *role*	a relation between individuals (relation instance) or concepts
resource, *individual*, object	instance of an entity class
domain, universe	set of possible entities that can be quantified
extension of a class	set of all instances of one class
ABox	set of observed instantiations (e. g. , of entity classes and relation classes)
concept, *entity (class)*, class, entity, type, category	a grouping of objects
interpretation, possible world	valid and consistent concrete assignment of instances to relations and concepts
vocabulary	set of terms naming concepts, individuals and relations
schema, terminological knowledge, concept level, type level, *TBox*	vocabulary without individuals
taxonomy	tree like hierarchy of classes; special case of ontologies where all relations have the same semantic, namely the *subsumption* relation.
model	In the context of formal semantics a model is an interpretation that satisfies a logical theory (evaluates an interpretation to be true). When working with probabilistic representations, e. g. , in machine learning, model is used to refer to the format how the data and the hypotheses are represented. Model is also commonly used in the context of software engineering to describe data representations or in databases where it defines database model or database schema which is the structure or format of a database.

Table 2.1: Commonly used terms related to ontologies accompanied by a deliberately oversimplified description.

ity - or short *expressivity* - for an expressive and formal knowledge representation. Formal representations are more accessibility to automated processing with less required human interaction. Thus, the user needs less background knowledge about the data and the data representation which allows for more powerful inference and more meaningful processing of the data by the computer. Ultimately, this makes data processing more flexible.

Scale of the ontology defines the complexity as well. This includes the number and complexity of formalisms of the ontology language that are effectively in use, as well as the number of given statements and facts (observations).

More complex ontologies are often described as heavyweight and simpler ones as lightweight (cmp. [Hepp, 2007]). *Lightweight ontologies* are commonly based only on a hierarchy of concepts and a hierarchy of relation. More complex lightweight ontologies include arbitrary relations and attributes of entity classes. *Heavyweight ontologies*, however, are enriched with axioms that are used to specify the semantic interpretation of concepts and relations. An illustration of expressivity vs. scale and lightweight vs. heavyweight ontologies is presented in Fig. 2.3.

This hierarchy of ontologies will be discussed in the following sections. Related existing real world ontologies will be mentioned in each section and discussed in Sec. 2.2.3.

2.2.1 Semi-Formal Knowledge Representations

The least computationally-expressive type of real world knowledge representations that we are covering in this thesis are *collaborative tagging systems*. Such collaborative knowledge platforms have recently emerged as popular frameworks for sharing information between users with common interests. Particularly popular examples of such systems are Amazon[1], Youtube[2], del.icio.us[3], CiteULike[4] or Flickr[5].

A key feature of these systems is that a large number of users upload certain resources of interest and label them with personalized tags. The resources are in most cases some type of high-dimensional raw data such as text documents or pictures. In contrast to taxonomies, where labels are part of predefined categories, there are no restrictions to tags.

Tags are flat and chosen arbitrarily. These free-form strings actually serve the purpose of organizing the resources of one single specific user. Tags might be polysemous and often different users use slightly different variations of tags to express the same semantics (e. g. , consider the tags "*statistical relational learning*", "*statistical-relational-learning*" and "*SRL*"). Furthermore, tags must not even follow any common semantic, i.e. a particular tag makes only sense for the user and not for the whole community. Consider e. g. , the tag "*to_read*". All these aspects compound the task of extracting meaningful information from collaborative systems (cmp. [Bundschus et al., 2009]).

[1] http://www.amazon.com/
[2] http://www.youtube.com/
[3] http://delicious.com/
[4] http://www.citeulike.org/
[5] http://www.flickr.com/

Therefore, the 'human-expressivity' of collaborative knowledge platforms is very high, the computational expressivity however is very low. Thus, tagging can be considered a *semi-formal knowledge representation*, because there is no explicit formal semantics defined for tags. Human background knowledge is needed to determine e. g. , that "*to_read*" is a note what to do with the resource and "*SRL*" describes what it is about.

On the other hand, tagging systems are growing rapidly and have reached scales that make automatic knowledge acquisition necessary (see [Bundschus et al., 2009]). Thus, tagging systems are very complex due to their scale, but concerning automatic processing they are quite restricted without a preprocessing knowledge extraction step.

Other very popular weakly-expressive knowledge representation that usually lack formal semantics are e. g. , schemas, thesauri or taxonomies. Sometimes they are considered ontologies because they potentially are equipped with well-founded formal semantics like description logic. However, most representation languages for simple schemas, thesauri and taxonomies are semi-formal. Examples of such representation languages are XML Schema, UML, Entity-Relationship models or RDF Schema (RDFS), but in the strict sense they cannot be considered formal ontologies. RDFS is a borderline case because it can be assigned a formal semantic, but often is not considered an ontology language if compared to OWL ontology languages. We will consider RDFS to be a formal, but weakly-expressive ontology language. This will become clear with our definition of formal ontologies in Sec. 2.2.2 and RDFS and OWL in Sec. 2.3. For a more in depth comparison of XML Schema, UML, Entity-Relationship models or RDF Schema to formal languages like OWL see e. g. , [Mika, 2007].

2.2.2 Formal Ontologies

In the AI-sense, an ontology refers to an engineering artifact, constituted by a specific vocabulary used to describe a certain reality, plus a set of explicit assumptions regarding the intended meaning of the elements in the vocabulary (cmp. [Guarino, 1998]). The term *formal ontology* is sometimes used to distinguish ontologies used in AI from the original use in philosophy. However, the term *ontology* as used in the computer science sense often refers to non-formal or semi-formal knowledge representation. Thus, not every knowledge representation in computer science that is called "ontology" is formal.

Informally speaking, by *formal* we mean that the components of an ontology are expressed in a formal language, consisting of a terminology (alphabet), a syntax (formal grammar) and formal semantic. This makes the knowledge representation precise and unambiguous, alleviating automated machine processing.

Note, that (semi-)formal ontologies are not necessarily more expressive than a non-formal knowledge representation. For instance, the human expressivity of tagging systems (semi-formal ontology) is indeed higher than of most formal ontologies, but this expressivity is not accessible to the computer. However, one could say that being formal is a precondition for an ontology to be computationally expressive.

This section focusses on weakly-expressive ontologies only. Highly-expressive ontologies are covered in section 2.2.3.

Definitions of Formal Ontologies

As many varieties of forms of ontologies, as many attempts there are to define the term. The most cited definition of ontologies in the computer science sense is:

> "An ontology is an explicit specification of a *conceptualization*." [Gruber et al., 1993]

This definition is quite abstract and adds the term *conceptualization* which needs to be defined itself. Gruber provides further explanations on his website[6]. In this context, a conceptualization means an abstract model of some aspect of the world, taking the form of the properties of important concepts and relationships. [Studer et al., 1998] try to give an intuitive explanation of this definition:

> "An ontology is a formal, explicit specification of a shared conceptualization. Conceptualization refers to an abstract model of some phenomenon. Explicit means that the types of concepts used, and the constraints on their use are explicitly defined. Formal refers to the fact that the ontology should be machine-readable. Shared reflects the notion that an ontology captures consensual knowledge, that is, it is not private of some individual, but accepted by a group."

This defines the terms used by [Gruber et al., 1993] in an intuitive manner. [Guarino, 1998] provides another intuitive definition:

> "An ontology is a logical theory accounting for the intended meaning of a formal vocabulary, i.e. its ontological commitment to a particular conceptualization of the world. The intended models of a logical language using such a vocabulary are constrained by its ontological commitment. An ontology indirectly reflects this commitment (and the underlying conceptualization) by approximating these intended models."

In addition, [Guarino, 1998] provides a formal definition of the terms used[7] (cmp. to the informal definition in Tab. 2.1):

Formal vocabulary: V is the vocabulary of a logical ontology language L.

Conceptualization: $C =< D, W, \Re >$ where D is the domain, W are the possible worlds and $\Re$ is he set of conceptual relations on the domain space D, W. Thus, a conceptualization is defined by its intentional/conceptual relations instead of a domain space with ordinary mathematical relations. Specifically the intentional relations are defined on possible worlds W not generally on the domain D.

Intentional / conceptual relations: ρ is defined as a function from W to the set of all relations on D. All conceptual relations of a possible world will contain the admittable extensions of ρ.

[6] http://www-ksl.stanford.edu/kst/what-is-an-ontology.html

[7] We adopt the notation by [Guarino, 1998]. The notation used in the later chapters of this thesis is not related to the terminology introduced here.

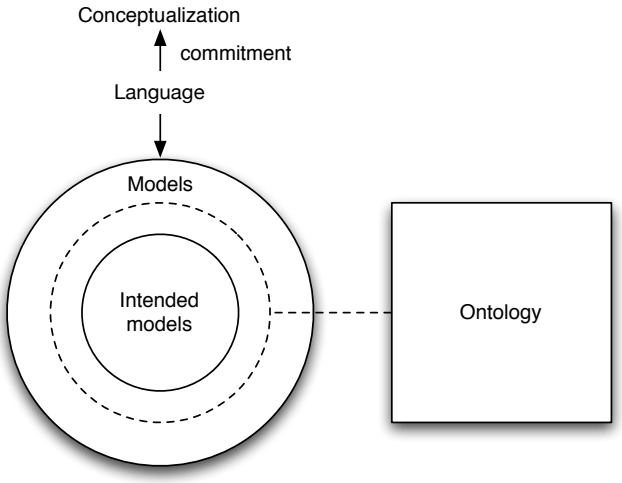

Figure 2.1: The intended models of a logical language reflect its commitment to a conceptualization. An ontology indirectly reflects this commitment (and the underlying conceptualization) by approximating this set of intended models (cmp. [Guarino, 1998]).

Ontological commitment: K is a intensional interpretation of a conceptualization C. An interpretation is defined by assigning elements of the set of conceptual relations $\Re$ in C to predicate symbols of V.

Intended models: The set of all models of a given ontology language L that are compatible with K will be called the set of intended models. Models of L are extensional interpretations in the form of assignments of elements of D and R to V. Intended models can be seen as the subset of models that are consistent with the conceptualization.

Ontology: O for a language L approximates a conceptualization C if there exists an ontological commitment K such that the intended models of L according to K are included in the models of O. In other words, an ontology for L is a set of axioms designed in a way such that the set of its models approximates the set of intended models of L according to K. An illustration is shown in Fig. 2.1

Differentiating between (extensional) relations, interpretations and models from intensional relations, intensional interpretations and intensional models might be of theoretical interest but is at the same time of little relevance to this thesis and to real world problems. However, it is relevant that the detailed formal definition of [Guarino, 1998] shows that the *defining* part of an ontology are the axioms used to approximate a conceptualization. The ability of defining formal semantics is *the* key requirement for ontology languages and the purpose of the ontology is the concrete specification of this knowledge using the ontology language.

According to this strict definition, non-formal or semi-formal knowledge representation languages that do not allow to define axioms cannot be called formal

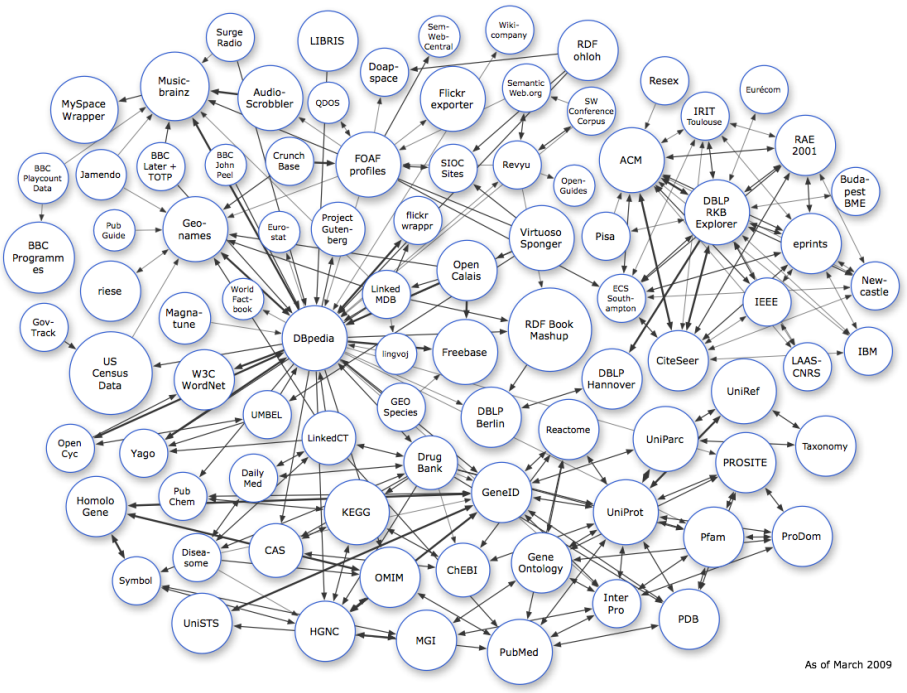

Figure 2.2: Linking Open Data cloud diagram showing published data sets and their interlinkage relationships.

ontology languages, because they do not provide formal semantics that can be used to define interpretations and models.

Existing Data

In recent years, large scale weakly-expressive formal ontologies have become widely available. Especially since the semantic web has gotten more attention numerous ontologies in semantic web languages have been created.

The most extensive data set currently available is a result of the linked data community[8]. The term *linked data* refers to a set of best practices for publishing and connecting structured data on the web. Key technologies that are deployed are URIs, HTTP, and RDF(S).

The current content of the linked data cloud is diverse, comprising data about geographic locations, people, companies, books, scientific publications, films, music, television and radio programmes, genes, proteins, drugs and clinical trials, online communities, statistical data, census results, reviews, and more. An overview of the sources and their links is shown in Fig. 2.2. This web of data currently consists of 4.7 billion RDF triples, which are interlinked by around 142 million RDF links (May 2009). For more details see [Bizer et al., 2009].

Strictly speaking, RDF is a metadata data model and not an ontology language in the sense of formal ontologies. Thus, linked data could also be considered a semi-formal knowledge representation. However, RDF is used as a base

[8] http://linkeddata.org/

data model for ontology languages like RDFS and OWL. If accompanied by RDFS the datasets are an example for weakly-expressive formal ontologies. For more details on semantic web ontology languages see Sec. 2.3. Examples of well-known RDFS vocabularies are FOAF[9], SIOC[10], SKOS[11], DOAP[12], vCard[13], Dublin Core[14], OAI-ORE[15] or GoodRelations[16]. Thus, existing data sets with RDFS vocabulary are weakly-expressive but well populated (see Fig. 2.3).

Another example for an area where large ontologies are already used is biology. The *gene ontology* [Ashburner et al., 2000], for example, has a hierarchical structure which defines where the position of each concept and its relation with others in the ontology are. Such taxonomies are another example for weakly-expressive formal ontologies. Some of the biological databases are also part of the linked data cloud. A specific subset with data from biomedical domains is linked life data [17].

2.2.3 Expressive Formal Ontologies

The effective use of formal ontologies requires not only a well-defined formal ontology language, but also support from reasoning tools. Without reasoning capabilities there is no advantage of more expressive over less expressive ontology languages in terms of the power of automated processing. Formal ontologies do not have an essential advantage over non-formal/semi-formal representations if the computer has no capabilities of automated reasoning. The only remaining advantage would be the fact that formal and expressive ontologies are less ambiguous for human interpretation as well.

Reasoning can be utilized in different phases of the ontology life cycle (cmp. [Baader et al., 2007]). First, during the design of the conceptualization and ontology reasoning can be used to check whether there exist contradictions or unintended interpretations and consequences, like synonymous concepts. Second, if different ontologies need to be integrated the integrated concept hierarchy can be computed and checked for consistency. This can also reduce unintended or missing subsumption relationships and incomplete assertions. Third, reasoning may be used during deployment of the ontology. For instance the hierarchy of concepts of an individual can be determined or (sets of) facts might be checked for consistency with the ontology. This is the main reasoning capability leveraged in this thesis.

There is a vast variety of ontology languages for formal knowledge representation. The two most commonly used logical formalisms for ontology languages are based on are *First Order Logic* (FOL) and *Description Logic* (DL). FOL is more expressive than DL. However, there is a trade-off between expressivity and complexity, making DL computationally less demanding.

The DL supplies a number of reasoning services which allow the construction of classification hierarchies and the checking of consistency of these descriptions.

[9] Friend of a Friend; http://www.foaf-project.org/
[10] Semantically-Interlinked Online Communities; http://sioc-project.org/
[11] Simple Knowledge Organization System; http://www.w3.org/2004/02/skos/
[12] Description of a Project; https://trac.usefulinc.com/doap
[13] http://www.ietf.org/dyn/wg/charter/vcarddav-charter.html
[14] http://dublincore.org/
[15] Open Archives Initiative, Object Reuse and Exchange, http://www.openarchives.org/ore/
[16] http://www.heppnetz.de/projects/goodrelations/
[17] http://www.linkedlifedata.com/

As DLs have clear semantic, it is possible to use all of the knowledge encapsulated in the ontology to reason whether it is consistent and complete. This is not possible with simple representations such as taxonomies.

When distinguishing between ontologies and formal ontologies it also makes sense to distinguish between formal semantics and the colloquial use of semantic. Recall from Sec. 1.2.1 that assigning a meta-description can be considered defining a semantic. That, includes e. g. , tagging, names of relational database tables, even filenames. Unstructured data like videofiles, images, plain text, audio-files are not considered to be a semantic data representation, even in the non-formal sense.

By contrast, formal languages can be assigned a formal semantic. A formal ontology has a formal semantics if there is a precisely defined entailment relation for sentences in this language leading to unambiguous interpretations of those sentences. Expressed in simplified terms, a formal ontology has a formal semantics if it enables deductive inference. We will provide a detailed definition of formal semantics using the example of OWL DL in Sec. 2.3.2.

For the method proposed in this thesis we actually do not need to be concerned with the details of formal semantics of the ontology language in use. This is more relevant for research on ontology sharing, fusion and translation. However, we need to identify key factors which ontologies need to fulfill to be applicable to the learning framework proposed here.

1. The ontology needs to contain vocabulary terms defining some form of concepts and properties of concepts (unary predicates) as well as relations (binary predicates)[18]. Furthermore, a domain of instances must be defined and there needs to be a mechanism to determine the membership of instances to concepts and relations. The DL term for that inference service is *realization*, which finds the most specific classes that an individual belongs to.

2. The ontology must define a logical theory that can be *checked for consistency*. It must provide tools that can be used to ensure that an ontology does not contain any contradictory facts. In DL terminology, this is the operation to check the consistency of an ABox with respect to a TBox.

Prerequisite 1 is essential for all approaches proposed in this thesis (see Ch. 4 and 5), prerequisite 2 is only essential if results need to be consistent with the ontology. This is subject of Ch. 4.

Existing Data

The dilemma of real world formal ontologies is that they either are expressive or large scale, but not both. This is an inherent consequence of practicality issues of expressive formal ontologies. Figure 2.3 visualizes the current situation of existing ontologies.

One of the reasons for the small scale of existing ontologies is the input and output format to and from expressive formal ontologies. Input and output in a formal representation that can be used by the computer for automated reasoning is complex and unintuitive for an uninformed user. However, a large number

[18] Unary predicates can also be seen as unary relations.

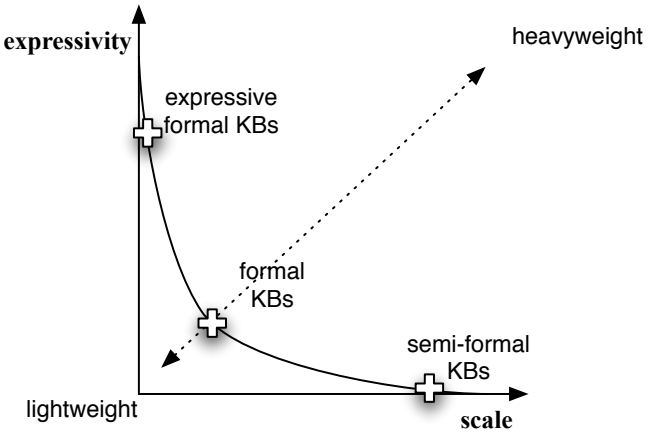

Figure 2.3: Expressivity and complexity of heavyweight vs. lightweight ontologies (dashed line) and existing semantic datasets (crosses and line).

of uninformed users are needed as information providers to make a knowledge base comprehensive, up-to-date and relevant for effective use.

Another problem resulting from the assumption that a large number of uninformed users contribute to an ontology is that it is very difficult to maintain the consistency of such a knowledge base. Representing knowledge in a logical language has proven to be extremely difficult in case of uncertainty, contradictions and changing information resulting from data that comes from distributed potentially untrustworthy (see Ch. 5) sources with varying quality, formatted according to different, but possibly overlapping schemas and containing possibly overlapping sets of instances.

However, if the assumption of a consistent ontology is relaxed the power of automated inference and the computational expressivity is reduced at the same time. This makes expressive formal and large scale ontologies an inherent contradiction. Thus, in our opinion computational expressivity vs. human usability is a more pressing issue for the success of a large scale deductive systems compared to the commonly discussed challenge of computational expressivity vs. computational complexity.

Due to that, semantic systems still are a scientific niche application. The main challenge that remains is to put ontological applications like the semantic web into practice on a large scale. As just mentioned, the essential technical obstacles are scalable reasoning, decentralized storage and control, easy input and output, and dealing with uncertainty. This thesis contributes to the last topic by inferring uncertain facts and dealing with untrustworthy information providers.

However, the potentials of ontologies are obvious at the same time. They use a more compressed knowledge representation compared to unstructured data sources, which can reduce computational complexity considerably. In addition, the semantic representation allows for a more precise and complex search compared to traditional keyword queries and thus can give more efficient results. However, the most promising improvement could be a more intelligent and reliable knowledge management for the user due to automated inference capabil-

ities. One could not only search for information but receive concrete answers. The computer could find information that is not explicitly stated beforehand. This thesis contributes to the last point, by inferring uncertain facts.

2.3 Ontology Languages for the Semantic Web

The vision of the *Semantic Web* (SW) is to provide information on the World Wide Web (WWW) in a machine processable way. Ontologies are the obvious choice for this challenge, as there exists a long history of research that can be leveraged.

The World Wide Web Consortium (W3C)[19] is the main international standards organization for the WWW and develops recommendations for the SW. The Web Ontology Language (OWL) is a set of knowledge representation languages endorsed by the W3C and considered one of the fundamental technologies for the SW. Here, we will shortly introduce RDF Schema (RDFS) (cmp. Sec.2.2.2) before concentrating on OWL DL semanticsthat are based on Description Logic. OWL was introduced in 2004 and in October 2009 OWL2 was announced by the W3C.

RDF is useful for making statements about instances, RDFS defines a schema and subclass hierarchies, and OWL DL is an expressive formal ontology language. Very elegantly, the statements in RDFS and OWL can all be represented as one combined directed graph in RDF. (Figure 2.4). A common semanticsis based on the standard that some of the language components of RDFS and OWL have predefined domain-independent interpretations.

2.3.1 Ontologies in RDF(S)

The recommended data model for the SW is the resource description framework (*RDF*). It has been developed to represent information about resources on the WWW (e. g. , meta data/annotations), but might as well be used to describe other structured data, e. g. , data from legacy systems. A resource stands for an object that can be uniquely identified via a uniform resource identifier, URI, which is sometimes referred to as a bar code for objects on the SW.

The basic statement is a *triple* of the form *(subject, property, property value)* or, equivalently, *(subject, predicate, object)*. For example *(tim, type, Person)*, *(tim, fullName, "Tim Miller")* indicates that Tim is of the concept (or class) *Person* and that Tim's full name is "Tim Miller". A triple can be depicted as a directed arc, labeled by the property (predicate) and pointing from the subject node to the property value node. The subject of a statement is always a URI, the property value is either also a URI or a literal (e. g. , String, Boolean, Float). In the first case, one denotes the property as object property and a statement as an object-to-object statement. In the latter case one speaks of a datatype property and of an object-to-literal statement.

A complete database (triple store) can then be displayed as a directed graph, a semantic net (Figure 2.4). One might think of a triple as a tuple of a binary relation *property(subject, property values)*. A triple can only encode a binary relation involving the subject and the property value.

[19] http://www.w3.org/

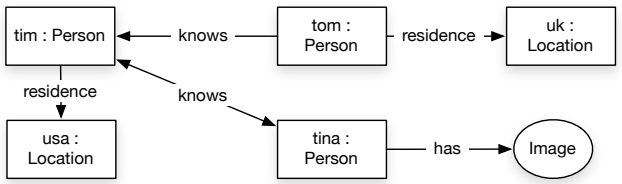

Figure 2.4: Example of a fragment of a RDF-graph from the social network example.

Each resource is associated with one or several concepts (i.e., classes) via the type-property. A concept can be interpreted as a property value in a type-of-statement. Conversely, one can think of a concept as representing all instances belonging to that concept.

Concepts are defined in the RDF vocabulary description language, *RDF Schema* or (*RDFS*). Both, RDF and RDFS form a joint RDF/RDFS graph. In addition to defining all concepts, the RDFS graph also contains certain properties that have a predefined meaning, implementing specific entailment rules. First, there is the subclass property. If an instance is of type *Concept1* and *Concept1* is a subclass of *Concept2*, then the instance can be inferred to be also of type *Concept2*. Subclass relations are essential for generalization in reasoning and learning.

Each property has a representation (node) in RDFS as well. A property can be a subproperty of another property. For example, the property *brotherOf* might be a subproperty of *relatedTo*. Thus if A is a brother of B one can infer that A is *relatedTo* B.

A property can have a domain and range, respectively: *(knows, domain, Person)* and *(knows, range, Person)* states that two resources of the concept *Person* can know each other. Note that, RDF/RDFS statements cannot lead to contradictions in RDF/RDFS, one reason being that negation is missing. This is one of the reasons why RDFS is usually not considered a *formal* ontology language.

2.3.2 Ontologies in OWL DL

As just introduced in the previous section, RDF(S) is a formal knowledge representation to model ontologies with low expressivity. In order to express complex knowledge and to specify a formal semantics more expressive ontology languages are needed. In this chapter, we focus on one particular language named *OWL DL* which is based on well researched description logic (DL). DL basically consists of classes and properties like the just introduced RDFS. However, in OWL DL this classes and properties can be correlated in a complex way. DLs are more expressive than propositional logic and comprise a subset of first-order predicate logic, which results in more efficient decision problems.

Examples of statements that can be expressed with OWL DL but not with RDFS are:

- A person has exactly one birthday.

Constructor	DL Syntax
intersectionOf	$C_1 \sqcap ... \sqcap C_n$
unionOf	$C_1 \sqcup ... \sqcup C_n$
complementOf	$\neg C$
oneOf	$\{o_1, ..., o_n\}$
allValuesFrom	$\forall R.C$
someValuesFrom	$\exists R.C$
value	$\exists R.\{o\}$
minCardinality	$\geq nR.C$
maxCardinality	$\leq nR.C$
cardinality	$= nR.C$

Table 2.2: DL constructors

- A person is either female or male.
- Persons that go to school are not in the age group 0-5.

These complex constructs are described by logical constructors. We will briefly introduce them next before we define the formal semantics of OWL DL.

OWL DL Syntax

OWL DL is a trade of between expressivity and complexity of logical inference tasks. OWL DL is less expressive than first order logic but reasoning is still decidable. However, OWL DL is considered more expressive than RDFS, although OWL DL forbids some RDFS constructs and thus is not a clear superset of RDFS. The description logic equivalent to OWL DL is called $\mathcal{SHOIN}(D)$ (cmp. [Horrocks and Patel-Schneider, 2003]).

The basic building blocks of OWL DL ontologies are classes C, properties R and individuals o (cmp. RDFS in Sec. 2.3.1). Properties are also called roles in OWL DL. *Person(tim)* denotes that the individual *tim* belongs to the class *Person*. The relation $knows(tim, tom)$ is an abstract role. Individuals correspond to constants in First-Order-Logic, classes to unary predicates and roles to binary predicates.

OWL DL class and concept constructs are used to define complex sets. A list of all constructors is shown in Tab. 2.2. The empty class is denoted as $\bot \equiv C \sqcap \neg C$ and the class that contains all individuals $\top \equiv C \sqcup \neg C$. The RDFS range constraint can be expressed by $\top \sqsubseteq \forall R.C$ and the RDFS domain constraint by $\exists R.\top \sqsubseteq C$.

In addition there are a number of axioms used to define restriction on these classes (see Tab. 2.3). For more details on the syntax of OWL DL and the relation of OWL DL to RDFS, OWL Lite and OWL Full we refer to [Hitzler et al., 2008].

Semantic

To define a formal semantics of $\mathcal{SHOIN}(D)$ the entailment relation needs to be defined. This is done by defining an interpretation for individuals, classes and roles on the one hand and for axioms on the other.

Axiom	DL Syntax
SubClassOf	$C_1 \sqsubseteq C_n$
EquivalentClasses	$C_1 \equiv ... \equiv C_n$
SubPropertyOf	$R_1 \sqsubseteq R_n$
SameIndividual	$o_1 = ... = o_n$
DisjointClasses	$C_i \sqsubseteq \neg C_j$
DifferentIndividuals	$o_i \neq o_j$
inverseOf	$R_1 \equiv R_2^-$
Transitive	$R^+ \sqsubseteq R$
Symmetric	$R \equiv R^-$

Table 2.3: DL axioms

The interpretation of identifiers of individuals, classes and roles is defined by a function to elements of the domain D, respectively. Hereby, identifiers of individuals can be assigned to elements of D, identifiers of classes to 2^D and identifiers of roles to $2^{D \times D}$.

After the definition of the interpretation of the vocabulary, the interpretations of complex constructors listed in Tab. 2.2 need to be defined. For instance, $C_1 \sqcap C_2$ denotes the union $\cup$ of all identifiers of the individuals of C and D. For a full list, see [Hitzler et al., 2008].

Next, the interpretation for each DL axiom (see Tab. 2.3) needs to be determined. Here, identifiers are not mapped to domain elements, but assignments of individuals are mapped to truth values. For instance $C(a)$ is interpreted as true if the individual identified with a is element of class C. Or $C_1 \sqsubseteq C_2$ is considered true if each identifier of an individual is member of C_1 and C_2. Again, for a full list, see [Hitzler et al., 2008].

To complete the definition of a formal semantics we have to define the satisfiability of the entailment relation. Given a $\mathcal{SHOIN}(D)$ Knowledge Base (KB) in form of a set of axioms, we define an interpretation to be a model of this KB if every axiom is assigned a truth value. If a model exists this KB is specified to be *satisfiable*.

2.3.3 Deductive Inference with OWL DL

In this section we will list some of the standard inference tasks that can be solved using an OWL DL reasoner. Subsequently, we will briefly mention one of the methods used for reasoning.

Using the syntax and the semantics of OWL DL ontologies, the knowledge can be described formally. In addition, the truth values of assertions not given in the ABox can be computed. This formalism can be used to ask questions about the concepts and instances described. The most basic decision problems are local queries to the KB like membership checking of individuals or more global KB tasks like subsumption and concept consistency of the KB.

Instance Membership: checks whether an instance a is member of a class C. $C(a)$ can be entailed if $KB \cup \{\neg C(a)\}$ is unsatisfiable.

Realization: is the retrieval of all instances that are members of a specific class. It finds the most specific classes that an individual belongs to.

Or, in other words, computes the direct types for each of the individuals. Using the classification hierarchy, it is also possible to get all the types for that individual.

Subsumption: is used to find out if C_1 is a subclass of C_2. This is the case if $KB \cup \{(C_1\ sqcap \neg C_2(a))\}$ is unsatisfiable. It is used to compute the subclass relations between every named class to create the complete class hierarchy. The class hierarchy is essential to answer queries such as getting all or only the direct subclasses of a class. There are similar problems that are focused on the T-Box like checking if two classes are equivalent or two classes are disjoint.

Concept Satisfiability: checks whether a concept is meaningful or more precisely if it is possible for a concept to have any instances. If class is unsatisfiable, then defining an instance of the class will cause the whole ontology to be inconsistent.

Consistency: is the most essential inference task. To check whether a certain KB is consistent, it needs to be decided if the KB is not unsatisfiable. This ensures that an ontology does not contain any contradictory facts. In DL terminology, this is the operation to check the consistency of an ABox with respect to a TBox.

All problems listed can be reduced to the last task of KB consistency. For more details on decision problems in DL and the reduction to consistency checks see [Baader et al., 2007, Hitzler et al., 2008]. The algorithm proposed in this thesis also uses KB consistency during the inference process (see Sec. 4.4).

The standard inference algorithms to check the unsatisfiability of a KB are tableaux algorithms. They try to build a tree-like model, the tableaux, by starting with an empty Tableaux and successively adding logical entailments of the KB. It stops either when a clash occurs or no more rules are applicable. If each branch in the tableaux contains a clash, there is no model and the KB is considered to be inconsistent. For more details on tableaux algorithms for satisfiability checking see e. g. , [Baader and Sattler, 2000, Baader et al., 2007, Hitzler et al., 2008]. The tool used in this thesis for consistency checking is Pellet (see [Sirin et al., 2007]) which is also based on tableaux inferencing.

2.3.4 Probabilistic Extensions of Semantic Web Languages

The formal ontology languages we have discussed so far are deterministic and not intended to handle uncertain knowledge. However, we are interested in methods that enrich ontologies with probabilistic information to support uncertain reasoning inside the ontologies. There are a number of proposals for extending logical languages with probabilistic information. Here, we will only list approaches that directly extend ontology languages, in particular RDF and OWL DL and DL. Logical representations that are not specifically related to SW languages are covered in Sec. 3.4. We adopt the classification of [Predoiu and Stuckenschmidt, 2008].

On the one hand, there are probabilistic extensions of DLs or other ontology languages and on the other hand, there are rule languages that combine rules and ontologies in some way.

First, are models that build on the syntax of established semantic web languages, examples being pRDF and BayesOWL (see [Ding, 2005]). pRDF is a probabilistic extension to RDF which represents the different elements of a Bayesian network and links them to regular RDF statements. Thus, RDF is basically used as the language to describe a Bayesian network. Inference capabilities equal those of a Bayesian networks and RDF(S) inference cannot be integrated.

BayesOWL is a probabilistic extension of OWL. While it represents probabilistic information about class membership within OWL ontologies its main reasoning task is to calculate the membership probability of an instance for all the classes in the ontology. Both support probabilistic reasoning for only a small subset of the expressivity of the languages which they extend.

Second, there are formalisms that consider extensions of description logic, examples being P-$\mathcal{SHOQ}(D)$ (see [Giugno and Lukasiewicz, 2002]) and P-CLASSIC (see [Koller et al., 1997]). P-CLASSIC is a probabilistic extension of the CLASSIC description logic by adding probabilities to roles of typical instances by the use of a Bayesian network. Reasoning tasks in P-CLASSIC compute the probability of a complex concept expression based on the definition of the joint probability distribution over atomic classes and features of relations. Again, only the reasoning capabilities of Bayesian networks can be used. As for P-$\mathcal{SHOQ}(D)$ no reasoning tools have been devised. However, the reasoning tasks that can potentially be solved are promising. For instance it could be determined whether a given knowledge base is consistent.

Third, there are approaches that integrate probabilistic logic programming variants and description logic. Such extensions are e. g. , Bayesian description logic programs (see [Predoiu, 2006]) or probabilistic description logic programs (see [Lukasiewicz, 2007]). They either integrate OWL with logic programming by specifying a logic program and a description logic knowledge base at the same time and allowing them to interact in some way. Or they constitute a translation from OWL to logic programming formalisms that have been extended with probabilities.

For more details about the mentioned approaches we recommend [Predoiu and Stuckenschmidt, 2008]. The comparison to our approach is presented in Sec. 4.1.

Chapter 3

From Propositional Learning to Learning with Ontologies

After introducing formal knowledge representations in the previous chapter, this chapter focusses on the second theoretical background needed, namely Statistical (Multi-)Relational Learning (SRL). The approach proposed in this thesis - the *Infinite Hidden Semantic Model* (IHSM) (see Ch. 4) - belongs to the area of SRL. Representations in SRL describe relationships between objects which partially bridges the gap to symbolic representation presented in the previous chapter. Consequently, SRL approaches are a natural counterpart to handle the relational character of ontologies.

However, by far the majority of traditional machine learning (ML) deals with non-relational *feature-based representations* (also referred to as *propositional representation* or *attribute-value representation*). Only recently statistical relational learning (SRL) is finding increasing interest in the ML community (see [Getoor and Taskar, 2007]). One element that the different approaches to SRL have in common is a more complex representation of the data. As in the preceding chapter we introduce the least computationally-expressive approaches first, adding more complex data representation in each section.

In doing so, we propose a four step hierarchy of representations accompanied by a selection of machine learning algorithms specialized for those representations (Sec. 3.1 - Sec. 3.4). Every step in the hierarchy represents a representation that is more formal and more computationally-expressive. The hierarchy can be compared to the one proposed in [Raedt, 2008], although our perspective is more application and data driven.

We start with the traditional non-relational approach to ML, using feature-based, propositional or attribute-value representations (see Sec. 3.1). We continue with single-relational approaches like matrix decomposition in Sec. 3.2 before focusing on techniques based on full multi-relational representations (see Sec. 3.3). Here, we show how the input representation of IHSM is extracted from an ontology in Sec. 3.3.1. Sec. 3.3.2, lists common approaches to multi-relational learning like kernel- and distance based methods , directed and undirected relational graphical models that are mostly based on a joint probabilistic model of

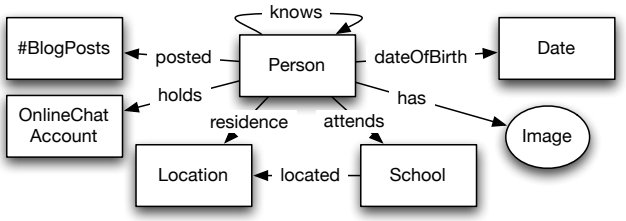

Figure 3.1: Schema or relational model of a social network.

the relational domain and finally the extension to latent class graphical models like IHSM. We conclude this chapter with SRL methods that are based on logical representations in Sec. 3.4.

3.1 Propositional Learning

Statistical learning has developed a large number of powerful analytical tools. The first step in statistical learning is to define a *statistical unit* which is the entity that is the source of the variables or features of interest [Fahrmeir et al., 2004, Casella and Berger, 1990, Trochim, 2000]. The goal is to generalize from observations on a few units to a statistical assembly of units. Typically, a statistical unit is an object of a given type, in the example of social network analysis e. g. , a *Person* (see the schema of a social network in Fig. 3.1).

In general, not all statistical units are of interest but only a particular subset, called the *population* (cmp. [Tresp et al., 2008]). The population might be defined in various ways, for example it might concern all students which are persons that attend a school.

In a statistical analysis only a subset of the population is available for investigation, known as a *sample*. Statistical inference is dependent on the details of the sampling process; the sampling process essentially defines the random experiment and, as a stationary process, allows the generalization from the sample to the population. In a *simple random sample* each unit is selected independently. Naturally, sometimes more complex sampling schemes are used, such as stratified random sampling, cluster sampling, and systematic sampling (cmp. [Tresp et al., 2009]).

The quantities of interest in the statistical investigation are the *features* (or variables) that are derived from the statistical units. The traditional way of representing ML problems is a vector of features for every statistical unit of the scenario to be modeled. The features of one specific observation, sample or example are stored in a single vector $x = x_1, ..., x_M$, thus each observation needs to have the same dimensionality. The observation has M attributes which in most cases are binary, discrete or real valued[1]. In the social network example, features might be the person's age.

In the next step a *data matrix* is formed where each row corresponds to a

[1] In this thesis we focus mostly on discrete random variables (e. g. , Bernoulli distributions and some multinomial distributions). Thus, variables can have two states: Either '1': existing and known, or '0': not known to be existent. In theory, this can be extended to any probability distribution but will change all hyper-parameters and the inference process considerably.

statistical unit and each column corresponds to a feature. Finally, an appropriate statistical model is employed for modeling the data, i.e., the analysis of the features and the relationships between the features. In relational learning this typically is an iterative process, e. g. , based on a first analysis new features might be added and the statistical model might be modified.

In supervised or discriminative ML settings (see e. g. , [Bickel et al., 2007]), where a learning model is optimized for a specic task, such as classication or prediction one has to partition the features in explanatory variables (a.k.a. independent variables, predictor variables, regressors, controlled variables, input variables) and dependent variables (a.k.a. response variables, regressants, responding variables, explained variables, or outcome/output variables).

Most commonly, in the discriminative setting, the vector of features is accompanied by a single output called label or value and denoted y. This 'feature vector + single label' is the dominant representation in use and is the only well established input format to popular machine learning software frameworks like Weka (cmp. [Hall et al., 2009]). There seem to be two reasons for the popularity of this format. First, it can be conveniently and intuitively represented in well established tabular forms like a spreadsheet. Second, it simplifies the learning task by making the so called *i.i.d. assumption* for all the variables except the label.

I.i.d. stands for independent and identically distributed which - simplified and amongst others - means that the joint probability density $P(x_1, ..., x_M)$ should be invariant to permutations of the indices. Accordingly, the terminology for x in the statistical literature is *independent variables* because the elements of x are independent of each other as well as of the label. Only the label y, known as the *dependent variable* in statistical terms depends on $x_1, ..., x_M$.

Examples of tasks where the assumption of identically distributed random variables is violated are situations where the training data and the test data are drawn from a different underlying distribution. This is for instance the case for domain adaptation (cmp. [Daumé and Marcu, 2006]), inductive transfer or multitask learning (cmp. [Rettinger et al., 2006]) problems where the new test data is drawn from a distribution that is related but not identical to the original distribution of the training data. Other examples for non-independent ML tasks are for instance the analysis of time series or sequential data [Dietterich, 2002] where $P(x_1, x_2) \neq P(x_1)P(x_2)$. Typical input data with a sequential structure is plain text or sound.

In the scenario of learning with ontologies and thus with multiple entities and relations it is impossible or at least insufficient to model all dependencies of the schema of the ontology in respect to a single label. On the one hand, given the task of learning with ontologies we want to estimate not a single label but a vector of structured, non-i.i.d. outputs (see Sec. 3.2). On the other hand, the input representation of random variables is also not statistically independent, but needs to be related in complex structures (see Sec. 3.3).

3.1.1 Propositional Representations

An intuitive and simple example for a propositional representation from the domain of social networks is the modeling of a single person and its properties. Every observation consists of one specific individual with a fixed vector of entries

like the name, age and gender of this person. Obviously, this covers only a small part of the information provided in a complex social network (see Fig. 3.1).

Such attribute-value formats can easily be represented in a formal language because each observation corresponds to an interpretation $\{R(x_1, ..., x_N)\}$, where R represents the matrix of all observations (cmp. [Raedt, 2008]).

From the perspective of logical learning such a representation is called *propositional* since it can be represented in propositional logic. Consequently, the process of transforming a more expressive symbolic representation into a propositional representation is called *propositionalization*. Features are extracted from a relational representation and then used for propositional learning algorithms resulting in an incomplete reduction. [Krogel, 2005]. The social network example in a propositional representation could be represented as

person(name, age, gender)

This data can be stored in a single table or matrix with M rows and N columns. M is the number of individual persons observed and N is the number of features per person (3 in this example).

$$\begin{pmatrix} x_{1,1} & ... & x_{1,N} \\ ... & ... & ... \\ x_{M,1} & ... & x_{M,N} \end{pmatrix}$$

In case of a discriminative setting we want to classify in classes:

class(pos/neg) $\leftarrow$ *person(name, age, gender)*

The first drawback of this simple representation is that a relation like *knows* between persons cannot be represented as a single label or class but a matrix of structured, non-i.i.d. outputs. Thus, the i.i.d. assumption does not hold for the input which is the matrix because we need to predict multiple outputs at the same time.

Another drawback is that the relational information of a social network with multiple entity classes like *Persons*, *Schools* and *Locations* and their relations *attends*, *residence* and *located* plus their attributes cannot be trivially condensed in one vector of independent variables. For instance, if we want to input all schools a person attended in the past this results in a different number of schools for every person instance.

An example of an ontology that can be modeled intuitively using a propositional representation are collaborative tagging systems (see Sec. 2.2.1). Here, a document could be considered a statistical unit and the tags of this documents are the features. Real world collaborative tagging systems provide a large amount of instance data which makes them interesting for ML. However, tagging systems provide semi-formal knowledge representations and thus offer no advantages like computationally expressive ontologies do.

3.1.2 Propositional Methods and Applications

Most of the traditional machine learning algorithms have been optimized for propositional data representation and supervised learning problems. Some pop-

ular examples are least squares, naive Bayes, artificial neural networks, kernel methods and numerous more (see e. g. , [Hastie et al., 2001]).

Tasks for these learning techniques - if applied to the example mentioned in the previous section - could be to predict attributes of *Persons* like the *Age* or *Gender*. As mentioned before, they are less suited for predicting relations between entities like *knows*.

Regarding collaborative tagging systems, the most popular application might be tag recommendation. Again, propositional learning is not well suited due to the large number of tags. Thus, collaborative filtering techniques (see [Tso-Sutter et al., 2008, Jäschke et al., 2007]) are preferably used (see Sec. 3.2.2). However, propositional machine learning algorithm such as support vector machines have been applied to this problem and used for the prediction of the most popular tags (see [Heymann et al., 2008]).

Note that for every attribute to be predicted a separate classifier needs to be trained in these systems. The reason for that is, the assumption that rows and columns are independent (i.i.d.) and only one output label is depending on all other elements of x. Specialized propositional algorithms for this category of problems deal with multi-label or multi-category classification.

However, once a relationship like *knows* needs to be predicted the limits of those methods become apparent. In this case we need to predict a structured output in form of a matrix with dependent entries. This will be discussed in the next section.

3.2 Single-Relational Learning

The next step in extending learning with propositional representation to more expressive formal representations we call single-relational learning. Here, we need to represent not only properties of one entity class, but two entity classes with a single relation that exists between those two entities[2]. This representation is called *multiple-instance learning* in [Raedt, 2008].

In order to represent relations in the form $R_{i,j}$ between instance i and j we can use the logical formalism of predicates $R(i, j)$. Predicates are not used in propositional logic, but are one of the main elements in more expressive logic like first order logic or description logic, where they are called *roles*.

If grounded or instantiated, all elements of a relation form a matrix of size number of entities N^1 of the first entity class $\times$ number of entities N^2 of the second entity class. A row in the resulting data matrix contains external input elements based on aggregated information (if available) and typically a large number of binary and sparse output elements. A '1' stands for a statement known to be true and a '0' for a statement whose truth value is unknown.

In contrast to propositional learning problems entries in this matrix are not considered independent anymore. The matrix itself is input and output at the same time and all entries are jointly predicted such that statistical strength can be shared between them. The reason is that some or all model parameters are sensitive to all elements, improving the estimates of those parameters.

[2]In this thesis we focus on relations between two entity classes also called binary predicates. In theory all concepts introduced here can be easily extended to n-ary relations. Accordingly, the resulting data matrices are n-dimensional

Although this representation seems not much different to a propositional data matrix, the feature vector representation demands an implicit decision of perspective: Whether the situation should be modeled from the perspective of the first entity class or from the perspective of the second entity class involved in the relation. Depending on the decision the independence assumption of propositional learning assumes different independencies of the features leading to the ignorance of the dependency-information of this dimension.

When adding additional attributes of the entities another insufficiency of propositional learning for single relations becomes apparent. Adding the properties of only one entity is feasible if the propositional representation is modeled from the perspective of this entity. This would be resulting in a $N^1 \times N^2 + K$ matrix where K is the number of properties. However, adding the properties of the other entity cannot be realized intuitively. You could either add it to the matrix, resulting in a $N^1 + K^2 \times N^2 + K$ matrix, where K^2 is the number of properties of the other entity. This will leave a $K^2 \times K$ part of the matrix empty, or the properties of every related entity instance could be redundantly added below every row. This is called a multi-instance representation in [Raedt, 2008].

Summing up, single-relational representation still can be represented in one matrix R, but there is no independence assumption which is one of the limits of propositional learners.

3.2.1 Single-Relational Representations

Extending the social network example from Sec. 3.1.1 to a single-relational representation we can now express the essence of a social network, the *knows*-relation (see Fig. 3.1) between entities of the class *Person*. Thus, the core of our data representation is a matrix $R : Person \times Person \rightarrow \{0, 1\}$ of dimensions number of persons $N \times N$. Here, rows and columns are not assumed independent anymore.

knows(Person, Person)

Recall that a standard relational representation for this example would be the table *knows* with *Person* as rows and columns. Equally, a relational adjacency matrix would have as many rows and columns as there are *Persons* and as many matrix entries as there are possibel true statements. A matrix entry is equal to one if the person instance actually knows this other person instance and is equal to zero otherwise.

In addition, we can also add K properties of persons as described before in Sec. 3.1.1 to the matrix. This additional property matrix can either be added to the matrix, leading e. g. , to a $N \times N + K$ matrix or as additional tuples after each individual leading to a *single table multiple tuple representation* (cmp. [Raedt, 2008]).

The 'knows-example' illustrates the special case of a self reference. The more general case involves two distinct entities for instance N^1 persons and e. g. , N^2 schools connected by e. g. , the relation "attends". The resulting matrix would be a $N^1 \times N + K$ matrix. Such a 2-entity relation can also be expressed as a bipartite graph.

$$\begin{pmatrix} x_{1,1} & \dots & x_{1,N^2} & x_{1,N^2+1} & \dots & x_{1,N^2+K} \\ \dots & \dots & \dots & \dots & \dots & \dots \\ x_{N^1,1} & \dots & x_{N^1,N^2} & x_{N^1,N^2+1} & \dots & x_{N^1,N^2+K} \end{pmatrix}$$

Further popular real world examples of situations that are best expressed as a *single-relational representation* are for example recommendation systems, where users recommend different items like movies[3]. Such recommendation engines can combine both perspectives, the one of determining the rating of a user by finding similar users according to their activities (ratings) as well as similar items to be recommended. This two different views are called *collaborative filtering* and item-based collaborative filtering, respectively.

Another example is link prediction which is needed for instance for webpages or citation networks. Similarly, tasks where the transfer of trained models to comparable but not identical situations like different plants or hospitals is needed. Here, the different hospitals would also be modeled as one vector. Furthermore, analysis of protein interaction (see [Bu et al., 2003]), topologies of power grids (see [Watts and Strogatz, 1998]), word co-occurence (see [Corman et al., 2002]), co-authorships (see [Newman and Girvan, 2004]), named entity recognition, protein structure prediction, or image restoration require a matrix as the output representation (see [Tresp and Yu, 2002]).

3.2.2 Single-Relational Learning Methods and Applications

As mentioned before propositional learners show disadvantages if used for single-relational representations. There is a class of learning approaches that do not have to make the independence assumption for x, predict elements of x instead of a single label y and thus learn only one model for the whole output matrix. Thus, a number of special ML algorithms have been proposed for learning with matrices and bipartite graphs.

Different terms are in use to describe approaches from this area. This includes hierarchical Bayes, multi-label prediction, random-effects models, random parameter models, mixed models, mixed effect models, nested models, multilevel models, hierarchical linear models, generalized mixed models, collaborative filtering, canonical correlation analysis, maximal covariance regression, partial least squares, multivariate regression, structured output prediction, Laplacian spectral clustering, principal component analysis, matrix factorization, tensor decomposition and more. For details see [Tresp and Yu, 2002] or [Ding et al., 2009].

Considering the example of recommendation engines the most popular approaches in use are eCommerce sites like Amazon[4] (see [Linden et al., 2003]). Here, "frequently bought together items" and "similar items" are shown to the user. Such simple approaches calculate a distance measure for users based on their ratings and properties (collaborative filtering) and use the ratings of similar users to make predictions for the active user. The same distance measures based approach can be applied to find similar movies and recommending

[3] http://www.netflixprize.com/
[4] http://www.amazon.com/

those to the active user. Another interesting implementation of a recommendation system that uses collaborative filtering is hunch[5]. Other collaborative filtering methods use measures of the correlation between the two entities like the Pearson product-moment correlation coefficient [Resnick et al., 1994a] or coclustering [Yoon et al., 2007, Shafiei and Milios, 2006].

Social network analysis research is another area that is concerned with the analysis of graphs and adjacency matrices. Traditionally, methods from this area are community detection approaches based on grouping the entities (nodes) to identify classes of homogenous elements. Here, communities are defined as densely connected subset of nodes that are sparsely connected to other densely connected subgraphs. Heuristic based approaches to extract those structures are e. g. , based on splitting using spectral methods, analyzing flows or modularity optimization [Newman and Girvan, 2004].

Another class of algorithms well suited for those types of problems are Matrix decomposition and reconstruction methods. Popular examples are matrix completion methods based on an eigenvector analysis of the data matrix (e. g. , *Singular Value Decomposition* (SVD)), e. g. , [Lippert et al., 2008], matrix completion based on Non-Negative Matrix Factorization (NNMF) [Lee and Seung, 1999] and matrix completion using *Latent Dirichlet Allocation* (LDA) [Blei et al., 2003]. All three approaches estimate unknown matrix entries via a low-rank matrix approximation. SVD is based on a singular value decomposition and NNMF is a decomposition under the constraints that all terms in the factoring matrices are non-negative. *Matrix decomposition/reconstruction* methods, e. g. , Principle Component Analysis (PCA) and other more scalable approaches have been very successful in the prediction of unknown matrix entries [Takacs et al., 2007]. Other approaches, which learn with the relational adjacency matrix, are described in [Yu et al., 2006] and [Yu et al., 2005].

LDA is usually not considered to be a matrix based approach, however it can also be used for matrix completion tasks and has the same basis like the IHSM model proposed in this thesis. It is based on a Bayesian treatment of generative topic models (see [Bundschus et al., 2009]): Given a key entity instance e_i (e. g. , a *Person*), the primary task is to predict the probability of unknown elements of the relational matrix like recommendations of items based on the known elements in the matrix.

To illustrate the connection to IHSM we will briefly give some technical details of the model (see Sec. 4.4 for more on the notation). We denote known elements for the entity as x_{obs,e_i}. The probability distribution for unknown elements x_{e_i} can be calculated using the model as follows:

$$P(x_{e_i}|x_{obs,e_i}) = \int \sum_z p(x_{e_i}|z)p(z|\theta)p(\theta|x_{obs,e_i})d\theta, \tag{3.1}$$

where θ denotes the multinomial distribution over latent variables z for a key entity. $p(\theta|x_{obs,e_i})$ is estimated during training.

After the completion of the matrix by any matrix based approach, the entries are replaced by values in the range of [0..1] and can be interpreted as certainty values that the corresponding statements are true.

More sophisticated approaches that use generative models like LDA also try to identify the latent structure of nodes in the graph to improve predictive per-

[5] http://hunch.com/

formance. In such latent class models, like stochastic block models [Nowicki and Snijders, 2001] each node e_i belongs to a cluster or latent variable Z. The probability of a link between two entity instances e_i is determined by their cluster assignments. For instance, person instance i in entity class *Person* e_i^{Person} has a probability to the school instance j in class *School* e_j^{School} according to their assignment to the component k of latent variable Z_k^{Person} and component l of latent variable Z_l^{School}, respectively. This approach is closely related to the IHSM. The terminology will be explained in detail in Sec. 4.4.

Some approaches extend the idea of latent variables to mixed or multiple component memberships. For instance, in the mixed membership stochastic block models [Airoldi et al., 2008] the vertices are allowed to have fractional class memberships. While those approaches share the idea of latent classes with the IHSM they are not applicable to multi-relational data representations which will be covered in the next section.

The purpose of such systems in regard to social networks could be to analyze entities and recommend new friends or schools to persons or persons to schools. As just mentioned, in other applications links between web pages or authors of papers could be predicted. The big advantage of such matrix completion methods is its applicability to large datasets, because they scale well and are well suited for setups with the open world assumption and sparse data.

3.3 Multi-Relational Learning

The obvious step from single-relational learning to *multi-relational learning* is to consider an arbitrary number of entities and relations. Obviously, this type of data demands a representation in more than one matrix. Similar to symbolic knowledge representations multi-relational representations are meaningful and expressive and still can be very concise.

Although it is possible to transform all information in one large matrix or even a vector representation by propositionalization the resulting data representation explodes in space complexity because of redundancies and important independence information is lost as shown in (see [Raedt, 2008]). Thus, in reality this is often not feasible because of an combinatorial explosion that would make propositionalizing real world data sets computationally intractable.

Multi-relational representations allow to model an arbitrary number of entity classes with properties and an arbitrary number of relations between those entity classes. If all possible relations between entity classes and properties are specified in advance for instance in a schema, the dependencies between the random variables is well defined. Thus, there is no general independence assumption over probability distributions, but only for predicates that are not 'connected' to each other.

Although learning with a multi-relational representation is already much more flexible and demanding compared to a propositional representation relying on the i.i.d. assumption, the assumption that relations can only exist between certain entities can also be seen as a limitation. ML problems that do not make any assumptions of independences, but try to find dependent variables from observations are called *structure learning* tasks (see e. g. , [Kok and Domingos, 2005]). This can also be extended to the point where predicates are actually invented (see e. g. , [Kok and Domingos, 2007]). In this thesis we assume that the

structure is defined by the ontology and the main goal is to learn the parameters of the probability distributions.

3.3.1 Multi-Relational Representations

If a multi-relational representation is used the *context* of a social network can be expressed. Recall the social network example in Fig. 3.1, we now can cover all information expressed in the schema. For every relation one gets one relationship matrix and one matrix for each entity class' properties. This makes such representations closely related to relational databases.

For instance, for the subset of the social network in Fig. 3.1

$$\begin{aligned} \textit{knows(Person, Person)} \\ \textit{attends(Person, School)} \\ \textit{residence(Person, Location)} \\ \textit{located(School, Location)} \end{aligned}$$

four matrices are needed: $R^{knows,M\times M}$, $R^{attends,M\times N}$, $R^{residence,M\times O}$ and $R^{knows,N\times O}$. Where M is the number of Persons, N the number of Schools and O the number of Locations.

There are several data structures that are used in the field of multi-relational learning for this kind of data. Most of them fall in one of the three following categories:

1. Representations based on graphs are the base structure for most multi-relational approaches (and single-relational approaches as discussed in the previous section). Other representations are usually transformed into some sort of graph-like structure at some point of the inference process. Thus, graphs are the most elementary representation for multiple relations.

   Intuitively, nodes in a graph represent an instance of an entity and edges represent a relation between those instances. Nodes and edges need to be labeled to assign them to specific classes of entities and types of relations. If unlabeled, the graph constitutes a single-relational setup with one known entity class, like the *knows*-relation between persons in the social network example. Bipartite graphs can be interpreted as a single relation between two known entity classes, see the movie-recommendation example. Multi-relational problems consist of graphs with more than one type of nodes and edges and restricted connectivity between types. Those restrictions determine the structure between entity classes and thus the independence of random variables.

   The graph is extended to a probabilistic knowledge representation in different ways. The most prominent examples used in SRL are Bayesian and Markov Networks (see Sec. 3.3.2).

2. Representations based on some variant of first order logic are inherently capable of representing multiple entity classes and relations with predicates. These approaches are covered in section 3.4, their representations have partly been covered in Ch. 2.

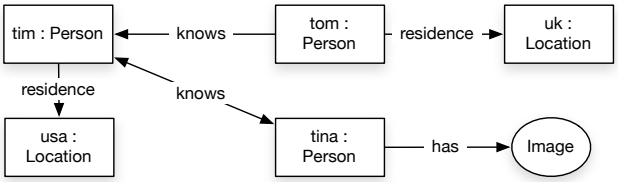

Figure 3.2: Partial sociogram of a social network. Rectangles depict concept individuals, ellipses depict attribute instances, and arrows depict relations.

3. Schema-based approaches are intuitive meta-description languages for graph-representations. They are generally based on conceptual representations of structured data. Popular examples are entity-relationship models in databases [shan Chen, 1976] or DAPER models [Heckerman et al., 2004]. They are closely related to weakly-expressive formal ontologies and are the basis for an intermediate data format used in the approach proposed in this thesis. We call such schema-based data representations *Relational Models* (RM) .

Multi-relational representations are the first representation in our hierarchy that can express the core elements of ontology languages, namely an arbitrary number of concepts and relations. This especially applies to basic vocabularies (like RDFS) and graphs (cmp. Sec. 2.2.2). Such representations are transformable to relational models.

Extraction of Relational Models from Ontologies

As mentioned in the previous section, weakly-expressive formal ontologies are closely related to RMs. More expressive ontologies can partialy be transformed into relational models by making simplifying assumptions and accepting the loss of information. For instance, more expressive constructs like axioms cannot be represented in a relational model.

The IHSM approach proposed in this thesis relies on an RM as an intermediate data representation. As we want to be able to use expressive formal ontologies as an input, a transformation step is needed first, to extract the RM from the logical representation. Later, the more expressive constructs are leveraged during the learning process.

This section will describe our approach to extract a multi-relational representation from a description logic ontology (see Sec. 2.3.2). The resulting RM can be used as the basis for most multi-relational learning algorithms. This includes the *Infinite Hidden Relational Model* (IHRM) which is the basis of the IHSM.

First, an abstract RM of concepts and roles defined in our social network ontology is created. Based on the TBox axioms given by the ontology we can - in case of our social network example - create a simple sociogram as depicted in Fig. 3.2. A sociogram consists of three different elements: concept individuals (individuals that are instances of a concept (e. g. , $tim : Person$)), attribute instances (relations between a concept and a literal (e. g. , $tina : hasImage$)) and role instances (relations between concepts (e. g. , $(tina, tim) : knows$)).

Concepts, attributes and roles are defined in the DL TBox and builds the basis of the RM.

Note, that many TBox elements first need to be deduced from the ontology, so that all individuals can be assigned to their most specific concepts. This process is known as *realization* in DL reasoning and described in Sec. 2.3.3. Realization finds the most specific classes that an individual belongs to or - in other words - computes the direct types for each of the individuals. The procedure of generating a grounded model from a logical representation is known as *knowledge based model construction* (see Sec. 3.4.1). The resulting directed graph of concepts connected by roles and annotated with attributes constitutes our RM. Fig. 3.1 shows the full RM we use for experiments in Sec. 4.5.

3.3.2 Multi-Relational Learning Methods and Applications

Although research on SRL is still new compared to traditional machine learning numerous approaches have been proposed to cope with the demands of representing probabilities as well as handling learning and inference with multi-relational representations. Aware of the risk of oversimplifying the large body of work on multi-relational learning we will mention only a few key methods.

In general, one could separate the methods in two groups: First, the methods that upgrade logical learning approaches like inductive logic programming to probabilistic representation (see Sec. 3.4.1). Second, the methods that start from statistical but propositional approaches and upgrade to multi-relational representation (for more on upgrading see [Raedt, 2008]). The later approach is the focus of this thesis.

Various propositional methods have been extended to relational representation. We will discuss four different approaches. First, we will briefly mention extensions of matrix completion algorithms to multi-relational representations and distance- and kernel-based methods as examples for non Bayesian treatments, before focusing on graphical models in form of directed, undirected and (directed) latent class graphical models. The latter are the basis of IHSM.

Multi-relational Matrix Completion

As shown in Sec. 3.2.2 matrix decomposition and reconstruction methods are well suited for single-relational representations. There are some attempts to extend matrix based approaches to multi-relational data. For instance, [Lippert et al., 2008] have shown how several matrices can be decomposed/reconstructed jointly making it a potential multi-relational algorithm. Their results show that this can increase predictive performance if compared to single-matrix decompositions.

By filling in the unknown entries via matrix decomposition/reconstruction, the approach has an inherent way of dealing with data that is missing at random. However, care must be taken if missing at random is not justified. In [Lippert et al., 2008], one type of statement concerns gene-gene interactions where only positive statements are known.

Scalability of this approach has not been studied in depth but the decomposition scales approximately proportional to the number of known matrix entries.

Note that the approach performs a prediction for all unknown statements in one global decomposition/reconstruction step.

Distance- and Kernel-based Methods

A substantial amount of research has been devoted to distance based methods to multi-relational representations. In a very abstract point of view, those methods try to construct a distance measure for complex structured data representations. In doing so, efficient methods, originally developed for propositional representations, can be effectively upgraded to work with richer structured representations (see [Gärtner et al., 2004]), including ontologies.

Similarity-based methods have been shown to effectively solve supervised and unsupervised learning problems in ontologies (see [d'Amato et al., 2008, Fanizzi et al., 2008a]), particularly those based on classification, clustering and ranking of individuals.

Distance-Based Learning Methods Distance-based methods, like the simple *k-Nearest Neighbor* approach, rely on a notion of similarity coded through a specific function for multi-relational representations (see [d'Amato et al., 2008]). These functions are based on a number of features that may be elicited from the KB through suitable methods based on stochastic search (see [Fanizzi et al., 2008a]).

For instance, because instances lack a syntactic structure that may be exploited for a comparison, on a semantic level, similar instances should behave similarly w.r.t. the same concepts, i.e. similar assertions (facts) should be shared. Conversely, dissimilar instances should likely instantiate disjoint concepts (see [Fanizzi et al., 2008c]).

An example of a multi-relational learning method for inducing a classifier is given by the *Reduced Coulomb Energy* (RCE) network (see [Fanizzi et al., 2009]), a simple form of Radial Basis Function (RBF) networks. In the *training* phase a network based on prototypical individuals (parametrized for each prototype) is trained adjusting hypersphere radii around them w.r.t. their classification some query concept. This network is then exploited during the *classification* phase to make a decision on the class-membership of further individuals w.r.t. the query concept on the grounds of the membership related to the hyper-spheres it lies in and the distance to the centers (prototypes). The efficiency of the method derives from limiting the expensive model construction to a small set of training individuals, while the resulting RCE network can be exploited in the next phase to efficiently classify a large number of individuals.

Kernels for Structured Representations The relational representation of data and background knowledge can be used to form a kernel function, enabling the application of kernel-based statistical learning algorithms (see [Frasconi, 2007]).

Kernel methods represent a family of very efficient algorithms, that ultimately solve linear separation problems in high-dimensional feature spaces where a kernel function implicitly maps the original instance space of the considered dataset (see [Shawe-Taylor and Cristianini, 2004]). Kernels for complex structures can be defined based on simpler kernels (working on specific aspects

of the structure) and exploiting the closure properties of such class of functions w.r.t. many operations. Using these properties, kernels for trees, graphs and other discrete structures have been introduced, that may be applicable to multi-relational representations. In [Gärtner et al., 2004] a principled framework is provided for defining new kernels based on type construction where types are defined in a declarative way.

Structural kernels for richer DL representations have been proposed in (see [Fanizzi et al., 2008b]). Such functions are actually defined for comparing $\mathcal{ALC}$ concepts based on the structural similarity of the AND-OR trees corresponding to a normal form of the input concept descriptions (see [Baader et al., 2003]). However, these kernels are not only structural since they ultimately rely on the semantic similarity of the *primitive* concepts assessed by comparing their extensions through a set kernel.

While these kernels were defined exploiting on specific structures which are language-dependent, a more flexible way to define them based on simple similarity functions parametrized on the semantics of instances w.r.t. a committee of concepts. Such kernels can be integrated with many efficient algorithms, that can implicitly embed feature selection. These functions transform the initial representation of the instances into the related active features, thus allowing for learning the classifier directly from structured data (see [Cumby and Roth, 2003]).

A more recent definition of kernel functions for individuals in the context of the standard SW representations is reported in (see [Bloehdorn and Sure, 2007]). The authors define a set of kernels for individuals and for the various types of assertions in the ABox (on concepts, datatype properties, object properties).

In a sense propositionalization (see Sec. 3.1.1) is related to kernel-based methods for structured representations because it represents relational data in the form of quantified propositions, which can be used with standard propositional and probabilistic learners. This technique allows the expansion of the space of possible propositional features to include many structured features defined over basic features abstracted from the input data instances (see [Cumby and Roth, 2003]).

Relational Graphical Models

The approaches mentioned in the previous section aim at describing the similarity of instances of relational data. In contrast, the matrix decomposition approach in Sec. 3.3.2 and the *Relational Graphical Models* (RGMs) in this and the following section predict the truth values of all possible statements in the RM. Unlike the matrix decomposition techniques (except LDA) and the distance based techniques, the RGMs are probabilistic models and statements are represented by random variables.

One can distinguish two cases of the application of RGMs to ontologies. In the first case - which also is the one taken in this thesis -, an RGM learns a joint probabilistic model over the complete ontology. This might be the most elegant approach since there is only one world and the dependencies between the variables are truthfully modeled. The draw back is that the computational requirements scale with the number of statements whose truth value is known or even the number of all potentially true statements.

More appropriate for large-scale applications might be the second case where

the problem is restricted to a segment of the ontology and in addition a sampling approach as described in [Huang et al., 2009] can be applied where only a subset of all known facts are considered during training.

As an example, consider that the statistical unit is a *Person*. A data point would then not correspond to a set of features but to a local subgraph that is anchored at the statistical unit, e. g. , the *Person*. As before, sampling would make the training time essentially independent of the number of instances in the ontology.

The RGM approaches typically make an open world assumption.[6] The corresponding random variables are assumed missing at random such that the approaches have an inherent mechanism to deal with missing data. If missing at random is not justified, then more complex missing data models need to be applied.

RGMs can be thought of as upgraded versions of traditional *Graphical Models* (GM), e. g. , Bayesian networks, Markov networks, dependency networks and latent variable models (cmp Sec. 3.3.1). RGMs have been developed in the context of frame-based logical representations, relational data models, plate models, entity-relationship models and first-order logic.

The two most common classes of graphical models that are used as the representation for RGMs are Bayesian networks and Markov networks. However, there are a few approaches that combine concepts from both classes like *Relational dependency networks* [Neville and Jensen, 2004] who also belong to the family of directed RGMs but learn the dependency of a node given its Markov blanket.

The underlying semantics of Bayesian networks are based on directed graphs and hence are also called directed GMs. Markov networks are based on undirected graphs, thus they are also called undirected graphical models. Next we will discuss relational extensions to both classes of GMs.

Directed RGMs *Bayesian networks* are a compact representation for a set of conditional independence assumptions about a probability distribution. The probability distribution in a directed RGM, i.e., relational Bayesian model, can be written as

$$P(\vec{U} = \vec{u}) = \prod_{U \in \vec{U}} P(U|par(U)).$$

U is represented as a node in a Bayesian network and arcs are pointing from all parent nodes $par(U)$ to the node U. One now partitions all elements of $\vec{U}$ into node-classes. Each U belongs to exactly one node-class. The key property of all U in the same node-class is that their local distributions are identical, which means that $P(U|par(U))$ is the same for all nodes within a node-class and can be described by a truth-table or more complex representations such as decision trees. Care must be taken, that no directed loops are introduced in the Bayesian network in modeling or structural learning. Popular examples for directed RGMs are relational Bayesian networks (see [Jaeger, 1997]), or Probabilistic Relational Models which will be discussed next.

Probabilistic Relational Models (PRMs): *Probabilistic relational models* were one of the first published directed RGMs and found great interest in the

[6] There are some exceptions, e. g. , MLN make a closed-world assumption during training.

statistical machine learning community (see [Koller and Pfeffer, 1998, Getoor et al., 2007]). PRMs combine a frame-based logical representation with probabilistic semantics based on directed graphical models. The nodes in a PRM model constitute the probability distribution of object attributes whereas the relationships between objects are assumed known. Naturally, this assumption simplifies the model greatly. PRMs have been extended to also consider the case that relationships between objects are unknown, which is called *structural uncertainty* in the PRM framework (see [Getoor et al., 2007]). The simpler case, where one of the objects in a statement is known, but the partner object is unknown, is referred to as *reference uncertainty*. In reference uncertainty the number of potentially true statements is assumed known, which means that only as many random nodes need to be introduced. The second form of structural uncertainty is referred to as *existence uncertainty*, where binary random variables are introduced representing the truth values of relationships between objects.

For some PRMs, regularities in the PRM structure can be exploited (encapsulation) and exact inference is possible. Large PRMs require approximate inference; commonly, loopy belief propagation is being used. Learning in PRMs is likelihood-based or based on empirical Bayesian learning. Structural learning typically uses a greedy search strategy, where one needs to guarantee that the ground Bayesian network does not contain directed loops.

Undirected RGMs The second common class of GMs used in SRL are *undirected graphical models* and called *Markov networks* or *Markov random fields*.

The probability distribution of an undirected graphical model or Markov network can be written as

$$P(\vec{U} = \vec{u}) = \frac{1}{Z} \prod_k g_k(u_k)$$

where g_k is a potential function, u_k is the state of the k-th clique of the network graph and Z is the partition function normalizing the distribution. Each potential is simply a table of values for each assignment of u_k that defines a 'compatibility' between values of variables in the clique.

The potential is often represented by a log-linear combination of a set of features f_k:

$$P(\vec{U} = \vec{u}) = \frac{1}{Z} \exp \sum_k w_k f_k(u_k)$$

where the feature functions f_k can be any real-valued function and where w_i are weights $\in \mathbb{R}$.

Two major approaches that use this representation for a relational extension are Markov logic networks (see Ch. 3.4.2) and Relational Markov Networks.

Relational Markov Networks (RMNs): *Relational Markov networks* generalize many concepts of PRMs to undirected RGMs [Taskar et al., 2002]. A RMN specifies a conditional distribution over all of the labels of all of the entities in an instantiation given the relational structure.

Intuitively speaking, it specifies the cliques and potentials between features of related entities at a template level, so a single model provides a coherent distribution for any collection of instances from the schema. To specifiy

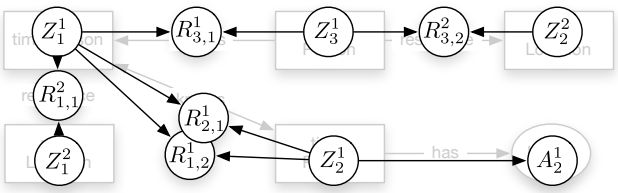

Figure 3.3: Hidden relational model of the sociogram defined in Fig. 3.2.

what cliques should be constructed in an instantiation RMNs use conjunctive database queries as clique templates. By default, RMNs define a feature function for each possible state of a clique, making them exponential in clique size.

RMNs are mostly trained discriminately, as they define a conditional distribution with a set of random variables to condition on and a set of target variables. In contrast to MLN, RMNs, as PRMs, do not make a closed-world assumption during learning.

Latent Class Relational Graphical Models (LCRGM)

Latent class relational graphical models can be seen as an extension to directed or undirected RGMs. They add unobservable latent variables to an RM. The infinite hidden relational model (IHRM) (see [Xu et al., 2006]) presented here is a directed RGM (i.e., a relational Bayesian model) with latent variables.[7] We will concentrate on the IHRM because it is the basis of the IHSM. In this section we will give an intuitive introduction to IHRM. A formal definition of IHRM and IHSM is given in Sec. 4.4.1.

Following [Xu et al., 2006] and [Kemp et al., 2006], we extend the RM to a Hidden Relational Model (HRM) by assigning a hidden variable denoted as Z_i^c to each individual i of concept c with current state k. Given that the hidden variables have discrete probability distributions they can be intuitively interpreted as clusters Z where similar individuals of the same concept c (in our case similar *Persons*, *Locations*, *Schools*,...) are grouped in one specific component k. These assignments of latent states specify the component one individual is assigned to.

The resulting HRM of the sociogram shown in Fig. 3.2 is depicted in Fig. 3.3. Following the idea of hidden variables in *Hidden Markov Models* (HMMs) or *Markov Random Fields*, those additional variables can be thought of as unknown properties (roles or attributes) of the attached concept. We assume that all attributes of a concept only depend on its hidden variable and roles depend on two hidden variables of the two concepts involved. This implies that if the hidden variables were known, attributes and roles can be predicted independently. In addition, the hidden variables in the IHSM incorporate restrictions in the form of constraints imposed by the ontology (see Sec. 4.2.2).

Considering the HRM model shown in Fig. 3.3, information can now propagate via those interconnected hidden variables Z. For instance, if we want to predict whether *tom* with hidden state Z_3^1 might know *tina* (Z_2^1) we need to consider a new relationship $R_{3,2}$. Intuitively, the probability is computed based

[7] Kemp et al. [Kemp et al., 2006] presented an almost identical model independently.

on

- the attributes A_3^1 and A_1^1 of the latent states of immediately related persons Z_3^1 and Z_2^1
- the known relations associated with the persons of interest, namely the role *knows* and *residence* $R_{2,1}$, $R_{3,1}$ and $R_{3,2}$
- higher-order information indirectly transferred via hidden variables Z_3^1 and Z_2^1.

In summary, by introducing hidden variables, information can globally distribute in the HRM. This reduces the need for extensive structural learning, which is known to be difficult.

Since the hidden variables play a key role in the HRM, we would expect that a HRM might require a flexible number of states for the hidden variables. Consider again the simple sociogram example. With little information about past friendships, all persons might look the same; with more information available, one might discover certain clusters in the persons (different habits of making friends); but with an increasing number of past friendships, the clusters might show increasingly detailed structure ultimately indicating that everyone is an individual.

Thus, it makes sense to aim for a not predefined number of clusters by using a Dirichlet process mixture model. This permits the model to 'decide' by itself about the optimal number of clusters and to find the optimal number with increasing observations. For our discussion it suffices to say that we obtain an Infinite HRM (IHRM) by letting the number of clusters potentially approach infinity, $K \to \infty$. Although from a theoretical point of view there are indeed an infinite number of components, a sampling procedure used for learning would only occupy a finite number of components. In the IHRM, an estimate of the optimal number of states is determined as part of the inference process. Due to this ability models with infinite hidden variables are also called non-parametric because the parameter that specifies number of states in the hidden variable do not have to be tuned as part of a complex optimization routine.

Since the dependency structure in the ground Bayesian network is local, one might get the impression that only local information influences prediction. This is not true, since in the ground Bayesian network, common children U with evidence lead to interactions between the parent latent variables. Thus information can propagate in the network of latent variables. Training is based on various forms of Gibbs sampling (e. g. , the Chinese restaurant process) or mean field approximations. Training only needs to consider random variables U corresponding to statements that received evidence, e. g. , statements that are either known to be true or known not to be true; random variables that correspond to statements with an unknown truth value (i.e., without evidence) can completely be ignored.

The IHRM has a number of key advantages compared to other RGM and finite RGMs. First, less structural learning is required, since the directed arcs in the ground Bayesian network are directly given by the structure of the SW graph. Second, the IHRM model can be thought of as an infinite relational mixture model, realizing hierarchical Bayesian modeling. Third, the mixture model allows a cluster analysis providing insight into the relational domain.

Technical details, Learning, Inference and Predictions with the IHSM which is based on the IHRM are discussed in Sec. 4.3, 4.4.1.

3.4 Towards First-order Probabilistic Learning

The approaches described so far do not rely on logical representations to construct models or define semantics and inference capabilities. At the most, they use formalisms like Daper- or ER-Models as a concise way to describe atomic ground networks. However, as described in Ch. 2 formal ontologies use formal knowledge representation languages like description logic (see Sec. 2.3.2. Thus, in order to apply machine learning in formal ontologies, while utilizing the reasoning capabilities and expressive power of the logical representation the learning methods need to be able to incorporate logical languages.

In the past years an increasing number of different SRL techniques with expressive formal representations have been proposed. Here, two different approaches were identified. First, models that start with an existing logical theory and extend it with probabilities and second, statistical models that are extended to structured data formats. Up to this point this chapter concentrated on the second type and in Sec. 2.3.4 some formalisms of the first type were mentioned. In the remainder of this Chapter we will touch on research that strives for formalisms that are no compromise in either way. We will call those approaches *first-order probabilistic learning*.

The first-order probabilistic learning approaches published so far offer - to our knowledge - no significant gain in expressive power compared to the ones in the previous sections. Thus, separating logic-based formalisms for SRL from the ones in the previous section might appear to be unnecessary and is usually not done in the literature. However, the approaches described in this section ultimately strive towards first-order probabilistic inference capabilities, also known as *lifted inference* (see below and in [de Salvo Braz et al., 2007]), which has the potential to be superior to the ones in the previous sections 3.3. First-order probabilistic logic ultimately might have the potential for more powerful inference, like the use of quantifiers for handling infinite numbers of possible worlds.

The approach proposed in this thesis is also using a logical representation and combines probabilistic and logical inference in one framework. The difference to the first-order probabilistic learning approaches presented here is that we are not proposing a probabilistic logic but use probabilistic inductive inference and logical deductive inference combined and in parallel but do not fuse both approaches. Both elements remain separate, without compromising their distinct capabilities. Thus, we do not offer a fundamental approach for first-order probabilistic inference and learning. Thus, the approaches discussed here are inherently different to IHSM.

First-order probabilistic approaches differ in representation and semantics of probabilities, logical formalisms and logical programs deployed, as well as power and way of inference, reasoning and learning. From this point on it is not reasonable anymore to distinguish by possible expression of independence assumptions and relational structures, as done in the previous sections. The differences in all aspects of the methods described here do not allow a general statement in this sense.

Being aware that there are numerous approaches to first-order probabilistic learning that are hard to categorize let alone compare because of their differences and regarding the inherent difference of IHSM to first-order probabilistic learning, this section is not intended to provide a general summary of the main elements of first-order probabilistic logic learning methods. For a more detailed and less selective overview we recommend [Cussens, 2007] or [Raedt et al., 2008]. We will only mention points that are relevant for putting our approach in perspective to existing work.

3.4.1 First-Order Probabilistic Representations, Inference and Learning

All first-order probabilistic learning approaches can be characterized in three aspects: First, their representation of logical formulae and probabilities, second their (deductive) inference mechanisms and third, the learning or inductive inference mechanisms. Those three topics will be selectively covered on a high-level in the following subsection.

First-Order Probabilistic Representations

As shown in Ch. 2 the types of constructs that might be used to express background knowledge in an ontology can be defined in a formal language like description logic. As most DLs are decidable fragments of first order logic, first-order probabilistic representations - in theory - inherently provide the expressivity needed for formal ontologies. Thus, we can cover first-order probabilistic representations in a general manner, without being concerned about the details of concrete ontology languages and their constructs like hierarchies, cardinality restrictions, role reflexivity, role disjointness, and so on.

Possible World Models: Most first-order probabilistic models make use of possible worlds (see Sec. 2.1 for details on possible worlds) to provide semantics for probabilistic statements. All statements in one instantiation or concrete possible state of the model can either be true or false. Reusing the social network example, the formula $attendsSchool(tim)$ is true in a given world if the individual which the constant tim denotes is an element of the set which the predicate symbol $attendsSchool$ denotes. This imposes a restriction on the states of each possibly statement (ground atom) which can either be true or false (binary).

In general this cannot be known to be true in all possible worlds, or it is even known not to be true. In Ch. 4 we give an example where people under the age of six must not go to a school. Thus, we know that either $attendsSchool$ or $underTheAgeOfSix$ can be true in one possible world.

Next this ground atom is assigned a binary random variable with a Bernoulli distribution stating the probability whether this ground atom is true over all possible combinations of ground atoms in all possible worlds. Thus, we can ask whether a probability distribution over all possible worlds satisfies e. g. , $P(attendsSchool(tim)) = 0.2$. This is correct if the set of worlds in which $attendsSchool(tim)$ is true has probability 0.2. The estimation of this probability hence is a marginalization since it is computed by summing over possible world probabilities.

ILP notation	SRL notation
constant	entity instance e_i
term t	constant, variable or compound term
predicate p	relation (class) R
logical atom h	a predicate of terms
ground atom or fact (clause with an empty body which can be assigned a truth value)	relation instance d

Table 3.1: ILP notation and corresponding SRL notation used in later chapters or a description if no direct mapping is possible (cmp. Tab. 4.4).

An assignment of truth values to all random variables is also called Herbrand interpretation. The set of all possible worlds is called Herbrand base. The approaches using possible world semantics differ in how these probabilities are defined and mapped to random variables, and how they are learned and used for inference.

The next step that is needed to get a full probabilistic model is to combine the probabilities on the truth values of atomic formulae. This defines a joint distribution over truth values of atomic formulae. If each possible world has a conjunction that it alone satisfies, this will result in a complete distribution over possible worlds. An interpretation for a relation is a set of 'true ground atomic formulae with the relation as the predicate symbol. This can vary across possible worlds.

For example, in one possible world the relation *attendsSchool* and *is_inAgeGroup* might have the interpretation $attendsSchool(tim, harvard)$, $ageGroup(tim, > 6)$ whereas in another it might be $is\_inAgeGroup(tim, < 6)$

The approach proposed in this thesis also constructs a joint distribution over possible worlds. Conjunctions of ground atomic formulae are checked for satisfiability as specified by the logical theory (see Sec. 4.2.2).

Entailment and Proof based Models Possible world models can be seen as an upgrade of graphical models to a relational representation (for more on upgrading see [Raedt, 2008]). Approach that avoid the construction of Herbrand interpretations by not explicitly encoding a set of joint distributions over possible worlds are entailment and proof based models.

Both approaches extend techniques from *Inductive Logic Programming* (ILP) to probabilistic settings. ILP models use *definite clauses* as key concepts to represent hypotheses. A definite clause cl is a formula of the form

$$h^1 \models h^2, h^3, ...$$

where R are logical atoms and $\models$ is the entailment relation. A clause cl entails another clause cl' if each model of cl is also a model of cl' (see Sec.2.2.2). A logic program consists of a set of clauses and its semantics is defined by all facts (ground atoms) which can be logically entailed. The process of proofing facts is called resolution. ILP is concerned with finding a hypothesis in form of a logic program from a set of positive and negative examples. The notation and how it relates to the notation used for IHSM is summarized in Tab. 3.1.

In Stochastic ILP (SILP) settings, probabilities (more precisely weights) are not attached to facts, but to the entailment relation in definite clauses or to resolution steps in a proof. Thus, the two directions are often called *learning from entailment* and *learning from proofs* . There are also some approaches which build upon other types of 'uncertain logic', for example [Carbonetto et al., 2005]. For more details on probabilistic ILP approaches to first-order probabilistic learning see e. g. , [Raedt et al., 2008].

In this thesis we are only looking at probabilistic approaches, thus purely symbolic approaches like non-probabilistic ILP is not considered. However, also SILP is inherently different to IHSM and other RGMs. Although IHSM and SILP approaches can use logical formulas as background knowledge and examples formalized in a logic language as data, the attachment of stochastically weighted logical formulas compared to joint probability distributions with the help of a given, formal theory which acts as a set of hard constraints is fundamentally different. For instance, the use of latent variables like it is done in IHSM has not yet been achieved in SILP. In addition we are not aware of work that uses ontology languages like description logic as background knowledge and examples for SILP.

First Order Probabilistic Inference

One advantages of first-order probabilistic models compared to non logic-based formalism is that they are potentially more powerful in inferring additional knowledge that is not explicitly given in the data. We will mention two areas where this potential is being used.

Knowledge Based Model Construction Constructing a ground network of probabilities of atomic formulae is not feasible in many real world scenarios. Directly stating all probabilities of interest is too restrictive and so formalisms provide mechanisms for defining probabilities which must be inferred rather than just 'looked up'. Many methods use inference to construct only the relevant parts of those models. This procedure is known as *knowledge based model construction* (KBMC).

If the query to be answered is known in advance only the relevant ground atoms need to be considered to answer the query. In addition conditional probabilities can be used for making learning more efficient. This is known as discriminative learning. The full ground network is never actually constructed. Instead, only just enough of it is constructed to answer any given probabilistic query.

The method proposed in this thesis uses deductive inference to reduce the ground network by constructing only the most specific concepts in the taxonomy. This reduces the size of the ground network in every possible world. The probabilities of less specific concepts can be inferred by combining the probabilities of the grounded network.

Lifted Probabilistic Inference As mentioned above the most powerful advantage of first-order probabilistic models is that they have the potential to make use of (probabilistic) first-order logical inference[8]. So far, the approaches

[8]This relates to *deduction* in non-probabilistic logic

introduced need to generate all related instances to calculate the probability of one unknown fact. This sharply contrast to inference procedures in non-probabilistic first-order logic or clausal logic like those based on resolution. In resolution grounding is avoided whenever possible, which can make inference much more computational efficient. Thus, first-order logical inference can deal with a potentially infinite universe as it is necessary when allowing quantifiers, partially observable data and an open world assumption.

Even at the rudimentary level of defining probabilities for atomic formulae some of the power of first-order methods is apparent. The basic point is that by using variables we can define probabilities for whole families of related atomic formulae. For instance, sometimes it is sufficient to calculate a probability when the number of true related ground instances is known. Which can be done without actually constructing all inferences and counting them. This type of inference is called *lifted probabilistic inference* and is still a largely open research question [Poole, 2003, Milch et al., 2008, De Salvo Braz et al., 2005].

By not integrating probabilities into the logical formalism our approach has the advantage that it can rely on existing powerful and efficient lifted inference techniques of the logical formalism in use (here DL reasoning).

First Order Probabilistic Learning

Lifted probabilistic inference is concerned with inferring the probabilities of unknown facts given the first-order probabilistic model with known parameters. The more relevant topic for this thesis is the preceding step, namely how the model and its parameter are learned from observations. This section does not go into technical details of specific approaches because there are numerous different ways of learning models and parameters depending on the concrete method. In addition there are usually several different proposal for each method, which makes it out of the scope of this thesis. The learning for IHSM will be explained in detail in Ch. 4.

As mentioned before there are two main elements that need to be learned to achieve a probabilistic model that can then be used for inference. On the one hand, the structure (or the logical program) needs to be learned if it is not prespecified by hand. Commonly, a search through the space of possible structures is performed. In most of the cases, some formulae are defined by a domain expert *a priori*. Additional formulae can then be learned by directly optimizing the pseudo-likelihood cost function or by using ILP algorithms. In the first-order learning case the sets of random variables in each of the example interpretations should correspond to those of the probabilistic logic.

On the other hand if the model is given (or learned), parameters need to be estimated. If a grounded GM can be constructed standard parameter estimation techniques for Bayesian and Markov networks can be applied. The most basic inference mechanism for Bayesian networks is the *Expectation Maximization* (EM) algorithm. For the lifted case corresponding parameters of the different instantiations are 'tied together' by the clauses defined by the first-order probabilistic model.

3.4.2 First-Order Probabilistic Methods

This section provides a high-level and very selective overview of probabilistic methods which incorporate both, logic and probability. As mentioned in the previous section logic based probabilistic formalisms usually define probabilities in two ways.

On the one hand, probabilities over interpretations are used which is mostly done using graphical models. Popular formalisms of this type are for example, Bayesian Logic Programs [Kersting and De Raedt, 2001] (BLP), relational Bayesian networks [Jaeger, 1997]. A popular undirected approach of this type is Markov Logic Networks (MLN). BLP and MLP as the most popular representatives will be introduced briefly in the next section.

On the other hand there are approaches using probabilities over proofs or entailments like probabilistic logic programming [Ng and Subrahmanian, 1990], independent choice logic [Poole, 1997], stochastic logic programs [Muggleton, 1996] and PRISM [Sato et al., 2005].

The approach proposed in this thesis is based on the IHRM which is a directed graphical model using possible world semantics (see section 3.3.2). It uses description logic inference to check the satisfiability of possible worlds needed in the inference process.

Bayesian Logic Programs

A *Bayesian logic program* (BLP) [Kersting and De Raedt, 2001] is an intuitive extension of previously described logic programs to a Bayesian setting. BLPs are defined as a set of definite and Bayesian clauses. A Bayesian clause specifies the conditional probability distribution of a random variable given its parents on a template level, i.e. in a node-class.

$$R^1|R^2, R^3, ...$$

Note, the similarity to a definite clause in ILP. In a BLPs, for each clause there is one conditional probability distribution and for each Bayesian predicate (i.e., node-class) there is one combination rule.

A special feature of BLPs is that, for a given random variable, *several* such conditional probability distributions might be given. As an example, $knows(e_1, e_2)|knows(e_1, e_3), knows(e_3, e_2)$ and $knows(e_1, e_2)|knows(e_2, e_1)$ specify the probability distribution for a $person_1$ knowing another $person_2$ if $person_2$ is known by a friend of $person_1$ or if $person_1$ is known by $person_2$. In this case the the truth value for $knows(e_1, e_2)|knows(e_1, e_3), knows(e_3, e_2), knows(e_2, e_1)$ can then be calculated based on various combination rules (e. g. , noisy-or).

BLPs use knowledge-based model construction to ground the Bayesian clauses. The result is a propositional Bayesian Network which defines the joint probability distribution. Then standard Bayesian network learning and inference like the EM-algorithms are used to learn parameters and predict probabilities of facts.

Relational Bayesian networks [Jaeger, 1997] are related to Bayesian logic programs and use probability formulae for specifying conditional probabilities.

Markov Logic Networks

Markov Logic Networks (MLN) combine first-order logic with Markov networks. The logical formulae are seen as soft constraints on the set of possible worlds. If an interpretation does not satisfy a logical formula it becomes less probable, but not impossible as in IHSM. In a MLN this is realized by associating a weight with each formula. The larger the weight, the higher is the confidence that a formula is true. When all weights are equal and become infinite, one strictly enforces the formulas and all worlds that agree with the formulas have the same probability.

Let F_i be a formula of first-order and let $w_i \in \mathbb{R}$ be a weight attached to each formula. Then a MLN L is defined as a set of pairs (F_i, w_i) [Richardson and Domingos, 2006] [Domingos and Richardson, 2007], where F_i is a formula in first-order logic and w_i is a real number. One introduces a binary node for each possible grounding of each predicate appearing in L given a set of constants $e_1, \ldots, e_N$, where N is the number of entity instances. The state of the node is equal to 1 if the ground atom/statement is true, and 0 otherwise (for an s-ary predicate there are N^s such nodes).

Thus, the nodes in the Markov network are the grounded predicates. In addition the MLN contains one feature (cmp. to Markov networks) for each possible grounding of each formula F_i in L. The value of this feature is 1 if the ground formula is true, and 0 otherwise andw_i is the weight associated with F_i in L. A Markov network $M_{L,\{e\}}$ is a grounded Markov logic network of L with

$$P(\vec{U} = \vec{u}) = \frac{1}{Z} \exp\left(\sum_i w_i n_i(\vec{u})\right)$$

where $n_i(\vec{u})$ is the number of formula groundings that are true for F_i. MLN makes the unique names assumption, the domain closure assumption and the known function assumption.

Like BLPs MLNs first construct all groundings of the first-order formulae. Then Markov networks inference can be used. The simplest form of inference concerns the prediction of the truth value of a grounded predicate given the truth values of other grounded predicates. In the first phase, the minimal subset of the ground Markov network is constructed by knowledge-based model construction that is required to calculate the conditional probability. In the second phase usually Markov Chain Monte Carlo algorithms are used to approximate the truth values in this reduced network.

Thus, learning consists of estimating the weights w_i. In learning, MLN makes a closed-world assumption and employs a pseudo-likelihood cost function, which is the product of the probabilities of each node given its Markov blanket. The basic Gibbs step consists of sampling one ground atom given its Markov Blanket. The Markov blanket of a ground atom is the set of ground predicates that appear in some grounding of a formula with it.

Note, that neither MLNs nor BLPs use lifted inference in learning and inference. Although they use powerful logical representations, strictly speaking, they cannot be called first-order probabilistic methods as discussed at the beginning of this section.

Chapter 4

Infinite Hidden Semantic Models for Learning with Ontologies

This chapter presents the main contribution of this thesis, namely the *Infinite Hidden Semantic Model* (IHSM). We will show how the IHSM can learn from facts given in expressive formal ontologies while preserving the consistency of such knowledge bases. In doing so, IHSM is the first approach that combines formal knowledge representations with infinite latent class statistical relational learning.

The background of formal knowledge representations and infinite latent class statistical relational learning have been introduced in Ch. 2 and Ch. 3, respectively. In this chapter, after discussing related work (see Sec. 4.1), we will introduce the concrete knowledge representation (see Sec. 4.2), Bayesian framework (see Sec. 4.3, learning and inference mechanisms (see Sec. 4.4) used in IHSM. In addition, we show how IHSM is implemented and applied in a social network scenario (see Sec. 4.5). To demonstrate the feasibility of our approach we provide experiments using real world data in form of an *OWL DL* ontology (see Sec. 4.6).

4.1 Motivation and Related Work

The proposed incorporation of formal knowledge representations with infinite latent class statistical relational learning offers several advantages. Benefits of the presented approach are (1) the analysis of known entity classes of individuals by means of clustering and (2) the completion of the Knowledge Base (KB) with uncertain predictions about unknown relations while (3) considering constraints as background knowledge for the machine learning process. Thus, it is guaranteed that the learned model does not violate ontological constraints and at the same time the predictive performance is potentially improved.

To provide an intuitive understanding of the presented approach we use a simple example throughout this thesis to illustrate the application of constraints in our learning setting: Consider a social network where, amongst others, the age

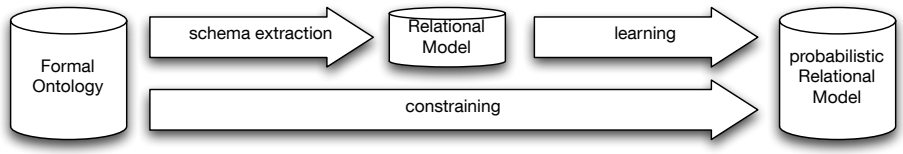

Figure 4.1: High-level overview of the IHSM workflow.

of persons and the schools they are attending is partially known. In addition, an ontology designer has specified that persons under the age of 5 are not allowed to attend a school. All this prior knowledge is provided in a formal ontology and the ultimate task is to predict unknown elements of this network.

Fig. 4.1 shows a high-level overview of the workflow of IHSM. The source of all information is provided by a *formal ontology*. On the one hand, the ontology is used to *extract schema* information for the *relational model* and assign instance knowledge to the schema. This step is also known as *knowledge based model construction*. On the other hand the axioms in the ontology are used to *constrain* the *learning*-process. The output is a *probabilistic relational model* that provides complete and uncertain but constrained predictions about all possible relations between instances.

This modular construction of our learning algorithm allows for a flexible yet tight integration of rich, formal knowledge and machine learning. Even though we use a social network ontology described in the Semantic Web (SW) ontology language OWL DL and settle on Statistical Relational Learning (SRL) as an apparently natural counterpart to logical representations, our general approach is in no way restricted to Description Logic (DL) or SRL and could be easily adapted to other formal and learning frameworks.

While there is some research on data mining for ontologies, like instance-based learning and classification of individuals, considering 'hard' constraints specified in the ontology during machine learning has hardly been tried so far or only in quite restricted and semi-formal ways. Details on related work will be discussed next.

4.1.1 SRL with Ontologies

In the previous chapters we showed to what extend machine learning algorithms can be applied to ontologies. On the one hand there are probabilistic extensions to formal knowledge representations to support uncertain reasoning inside ontologies (see Sec. 2.3.4), however, those approaches do not focus on how to learn the probabilities from observations and known facts.

On the other hand there are learning algorithms that are based on expressive probabilistic logical representations (see Sec. 3.4). While the latter formalisms offer a rich variety of ways on how to learn probabilities and reason under uncertainty none of the existing approaches is focused on formal ontologies. Although there are a few examples like [Domingos et al., 2008, de Oliveira, 2009] that show how to use their SRL approach with SW-ontologies in theory they do not present empirical tests on a large and expressive DL data set. Specifically, ontologies are not interpreted as hard constraints and existing powerful inference capabilities of ontology languages are not used to enforce them. In addition,

those existing first-order probabilistic learning techniques do not integrate latent variables in their models.

As discussed in the previous chapter, most existing algorithms for learning with ontologies ignore expressive ontological constructs and restrict the representation for instance to a schema. On the one hand, this is the result of the increasing complexity of the learning algorithms needed for more expressive formal ontologies. On the other hand, the existing deductive inference capabilities of formal ontologies cannot be easily integrated in a probabilistic reasoning framework.

So far, machine learning has been mainly approached either with statistical methods, or with techniques which aim at the inductive learning of formal knowledge from examples which are also provided using formal logic. Thus, learning approaches that are naturally able to handle formal knowledge representations are mostly not probabilistic. The most important direction in this respect is *Inductive Logic Programming* (ILP) (see Sec. 3.4.1 and [Lisi and Esposito, 2007]). *Probabilistic-* and *Stochastic Logic Programming* (e. g. , [Raedt and Kersting, 2003]) (SLP) are a family of ILP-based approaches which are capable of learning stochastically weighted logical formulas (the weights of formulas, respectively) (cmp. Sec. 3.4.1). In contrast to that, our approach learns probability distributions with the help of a given, formal theory which acts as a set of hard constraints.

Although (S)ILP and SRL are conceptually very closely related and often subsumed under the general term *relational learning*, SRL still is rarely integrated with formal logic or ontologies as prior knowledge. For instance [Reckow and Tresp, 2008] use ontologies in a similar model but only use taxonomic information as additional 'soft' knowledge (i.e., knowledge which can be overwritten during the learning process) in the form of features for learning. They do not restrict their results using formal hard constraints.

Surprisingly, there are also hardly any applications of (pure) SRL algorithms to SW ontologies. The few examples, e. g. , [Kiefer et al., 2008], [N. Fanizzi, 2008], do not consider formal constraints. There are various approaches to the learning of categories in formal ontologies from given instance data and/or similar categories (e. g. , [Fanizzi et al., 2008c]). However, these approaches do not allow for the statistical learning of relations in the sense of SRL and their aims are more related to those of ILP than to our learning goals. Although there are applications of SRL to social networks, such as [Xu et al., 2008]; none of those approaches uses a formal ontology or any other kind of formal knowledge. Furthermore, the social networks examined in this related work are significantly less complex in regard of the underlying relation model.

This thesis focuses on the combination of statistical machine learning (ML) with ontologies specified by formal logic. In contrast to existing approaches to the use of constraints in ML and data mining, we exploit a semantically rich and fully formal representation of hard constraints which govern and support the stochastic learning task. Technically, this is achieved by combining the *Infinite Hidden Relational Model* (IHRM) approach to Statistical Relational Learning (SRL) with inference guided by the constraints implied by a *Description Logic* (DL) ontology used on the SW. In this way, our approach supports a tight integration of formal background knowledge resulting in an *Infinite Hidden Semantic Models* (IHSM). The term *Semantic* in IHSM represents this integration of 'meaningful' symbolic knowledge and the use of deductive reasoning.

4.1.2 Learning with Constraints

The use of *hard constraints* for clustering tasks in purely statistical approaches to learning, as opposed to the ubiquitous use of 'soft' prior knowledge, has been approached in, e. g. , [Davidson and Ravi, 2007]. A common characteristic of all approaches to learning with hard constraints is that they work with a relatively narrow, semi-formal notion of constraints and do not relate constraints to relational learning. Constrained clustering [Basu et al., 2008] can achieve a similar latent analysis of the data like IHSM, however those methods are also not focused on using formal knowledge bases and deductive consistency checking to enforce the constraints.

In contrast to these efforts, our approach allows for rich constraints which take the form of an OWL DL knowledge base (with much higher expressivity) (see Sec. 2.3.2). The notion of forbidden pairings of data points (*cannot-link* constraints [Davidson and Ravi, 2007]) is replaced with the more general notion of logical (un-)satisfiability w.r.t. formal background knowledge. We interpret ontologies as constraints on possible worlds that might be predicted in an inductive inference task.

The closest topic in machine learning literature to our approach of integrating ontologies as hard constraints is known as "*learning over constrained output*" [Punyakanok et al., 2005]. The setting adopted in this line of work relates to structured output classification (see Sec. 3.2), where complex dependencies among the output variables are captured by hard constraints. [Punyakanok et al., 2005] compare three different learning strategies and try to make theoretical statements about which strategy performs best under certain independence assumptions, comparable to our classification of learning algorithms in Ch. 3.

First, only stand-alone local classifiers without leveraging the structure-based inference process are used. This corresponds to learning algorithms for propositional representations (see Sec. 3.1.2). Thus, for every value in the collection of output variables y[1] a distinct propositional learner - in this case a perceptron algorithm - must be trained. All outputs are assumed to be i.i.d. (see Sec. 3.1).

Second, the same local classifiers are learned but are constrained after the learning process is finished. This can be seen as a subsequent pruning of structural inconsistent predictions. The learned classifiers are not altered in the process. This corresponds to our results of IHRM+Constraining (see Sec. 4.6.3).

Third, a global classifier is learned that can produce the correct global classification under consideration of the constraints/structure. Here inference is used while learning the parameters of the classifier and the scheme is called *inference based training*, accordingly. This comes closest to our IHSM approach.

The findings of [Punyakanok et al., 2005] are that the performance of the different schemes are dependent on the number of available trainings samples. First, when the local classification is linearly separable, the local classifier outperforms the other schemes. And second, as the local problems become more difficult and are no longer linearly separable, the global classifier outperforms the others, but only with sufficient number of training examples. These results are quite intuitive and correspond to our empirical results (see Sec. 4.6.3).

[1] In our classification in Sec. 3.2 we assumed y to be a matrix R because this corresponds to the single-relational and most common case in structured output prediction. Here, no specific structure is assumed.

Although this work is quite related to our approach the results can be transferred just to a limited extend to the setting of learning with formal ontologies. First and foremost, the results only apply to linearly separable classification problems, which is almost never the case in complex ontologies. Second, learning local linear classifiers seems not applicable in scenarios used in this thesis. Thus, this line of work does provide useful theoretical background for a simpler scenario but cannot be directly transferred to IHSM and formal ontologies.

4.2 Formal Framework

Our approach requires the specification of formal background knowledge and formal constraints for the learning process. We do so by letting the user of the proposed machine learning algorithm specify a formal ontology or use an existing ontology e. g. , from the SW. In computer science, an ontology is the formal representation of the concepts of a certain domain and their relations (see Ch. 2).

In the context of the (semantic) web and also in our approach, an ontology is typically given as a so-called *TBox* and *ABox* (see Tab. 2.1), each of which consists of a number of description logic formulas. The TBox comprises conceptual knowledge (i.e., knowledge about classes and their relations), whereas the ABox comprises knowledge about the instances of these classes. In our context, the given ontology and also all knowledge which is a logical consequence from it constitutes the 'hard' knowledge for the learning process (i.e., knowledge which cannot be overwritten during learning), as described later.

However, our approach is not restricted to ontologies, but works in principle with all sorts of formal knowledge bases. We are using ontologies mainly because there is an obvious relatedness of clustering and ontological classification, and because formal languages, reasoners, editors and other tools and frameworks for ontologies on the SW are standardized and widely available. Consequently, we use a description logic for our examples. This is not only because ontologies and other formal knowledge on the Web (which is our application here) are usually represented using DL, but also because the standard DL which we use is a decidable fragment of first-order logic (FOL) for which highly optimized reasoners exist.

4.2.1 Formal Background Knowledge

The constraints, the examples and the initial social network are given as a set of DL axioms (T and ABox). Constraints and the initial social network are considered to be certain knowledge and evidence. Thus, the initial facts cannot be supplemented by adding probabilities. However, this is not a general restriction and can be easily integrated in future work. However, facts already are assumed to be erroneous or noisy. They can even be inconsistent with the TBox as long as the TBox is consistent. During the learning process the algorithms does identify inconsistent individuals and separates them from the sample population.

We settle on the $\mathcal{SHOIN}(D)$ (see [Horrocks and Patel-Schneider, 2003]) description logic, because entailment in the current SW standard ontology language OWL DL can be reduced to $\mathcal{SHOIN}(D)$ knowledge base satisfiability

$$
\begin{array}{c}
Person \sqsubseteq Agent \\
knows^{-} \sqsubseteq knows \\
\exists knows.\top \sqsubseteq Person \\
\top \sqsubseteq \forall knows.Person \\
\exists hasBD.\top \sqsubseteq Person \\
\top \sqsubseteq \forall hasBD.DOB \\
\top \sqsubseteq \leq 1\ hasBD \\
\top \sqsubseteq \geq 1\ hasBD \\
\exists yearValue.\top \sqsubseteq DOB \\
\top \sqsubseteq \forall yearValue.gYear \\
\top \sqsubseteq \leq 1\ yearValue \\
\top \sqsubseteq \forall attends.School \\
\top \sqsubseteq \forall residence.Location \\
\top \sqsubseteq \leq 1\ residence \\
... \\
\end{array}
$$

Table 4.1: Fragment of the FOAF-ontology in $\mathcal{SHOIN}(D)$ used for experiments. DOB means "date of birth" and $hasBD$ means "has birthday".

(see Sec. 2.3.3). We could likewise work with OWL DL syntax directly, but that would not have any technical advantages and would just reduce the readability of our examples.

Our approach requires that the satisfiability or consistency of ontologies can be checked, which is a standard operation of most automated reasoning software for the SW (see Sec. 2.3.2). Allowing to check the satisfiability means that the reasoner is able to check whether a given KB (ontology) has a model. On the syntactic level, satisfiability corresponds to consistency, i.e., there are no sentences in the given ontology which contradict each other.

A $\mathcal{SHOIN}(D)$ *ontology* or *knowledge base* is a non-empty, finite set of TBox axioms and ABox assertions. A rudimentary semantics of $\mathcal{SHOIN}(D)$ is defined in Sec. 2.3.2, the canonical semantics which we assume in this work can be found, e. g. , in [Horrocks and Patel-Schneider, 2003].

4.2.2 Formal Constraints

Constraints in the sense of this work are actually just formal statements. Our approach is expected to work with all kinds of logical frameworks which allow for satisfiability (or consistency) checks over some given set of logical sentences, for example an ontology. This set of given statements is denoted as the KB in the further course of this paper. Formally, we define a set of constraints AX to be the deductive closure $\Omega(KB)$ of a given knowledge base KB, with $\Omega(KB) = \{AX | KB \models AX\}$. The deductive closure contains not only explicitly given knowledge (the knowledge base KB), but also all logical sentences which can be derived from the KB via deductive reasoning. For instance, if the KB would contain the sentences $\neg a$ and $\neg a \rightarrow b$, the deductive closure would also contain b.

The application-specific constraint set which we use as an OWL DL ontology is similar to the well-known *Friend-Of-A-Friend* (FOAF) social network schema (see Sec. 2.2.2), together with additional constraints which will be introduced

$$tim : Person, tina : Person, tom : Person$$
$$(tina, tim) : knows, (tina, tom) : knows$$
$$tim : \neg(\exists knows.\{tom\})$$

Table 4.2: Examples of individuals used in the FOAF-ontology in $\mathcal{SHOIN}(D)$.

$$Pupil \sqsubseteq Person$$
$$Pupil \sqsubseteq \neg UnderSixOld$$
$$Pupil \sqsubseteq \exists\, attendsSchool$$

Table 4.3: Examples of constraints used in the FOAF-ontology in $\mathcal{SHOIN}(D)$.

later. In order to abstract from (for the purpose of our work) unnecessary detail, the ontology SN in Tab. 4.1 comprises only a fragment of the full FOAF-like ontology we have used.

These axioms mainly express certain properties of binary relations (so-called *roles* (see Tab. 2.1 for a disambiguation of terms related to ontologies) between classes. For example, $\top \sqsubseteq \forall attends.School$ specifies that in our example ontology the range (target set) of role *attends* is $School$.

In addition to these, we provide the machine learning algorithm with an ABox which models an incomplete social network. An example for such additional individuals-governing constraints OBS is presented in Tab. 4.2. In ML terminology this could also be denoted as observations, data or evidence. The later machine learning task consists essentially in a (uncertain) completion of this given network fragment.

Note, that these relations among persons cannot be weakened or overwritten by the learning process, even if they contradict observed data. They need to be provided manually by the KB engineer. As further constraints, we assume some specific properties SC of the analyzed social network. The following set of axioms expresses that no one who is younger than six years goes to school (see Tab. 4.3). At this, $UnderSixYearsOld$ is the class which contains persons with an age less than six years (calculated from the given dates of birth).

The complete set of given formal and definite knowledge for our running example is then $IKB = \Omega(SN \sqcup OBS \sqcup SC)$.

The set of data OBS used as examples for the learning tasks takes the form of ABox assertions or ground formulas. We do not demand that examples are mutually consistent, or consistent with the ontology. In order to maintain compatibility with the expected input format for relational learning, we restrict the syntax of examples to the following two description logic formula patterns:

$$instance : category$$
$$(instance_i, instance_j) : role$$

At this, roles correspond to binary relations.

4.3 Bayesian Framework

After describing how formal background knowledge and formal constraints are specified we turn to the probabilistic framework used for the learning process.

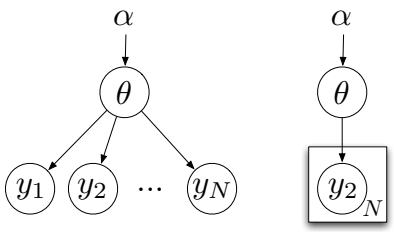

Figure 4.2: A standard Bayesian model (left) and the same model in plate representation (right).

IHSM is based on a specific class of Bayesian models, namely a infinite Dirichlet process mixture model. In this section the basics will be discussed in a nutshell by summarizing parts, adopting the notation and quoting from [Xu, 2007]. The concrete IHSM model, learning and inference is covered in the Sec. 4.4.

4.3.1 Bayesian Model

In a basic *Bayesian model* a set of N observations $OBS = y_1, y_2, ..., y_N$ is distributed according to $P(\cdot|\theta)$, where θ represents the unknown parameters of the distribution. The unknown parameters in turn are *random variables* and are drawn from a distribution $P(\theta|\alpha)$ with hyperparameters α. This simple setup is illustrated in Fig. 4.2 using a graphical representation, called *plate model*. The plate simplifies the illustration by not explicitly showing replications of subgraphs. In a plate model, non-random variables (*parameters*) are represented directly by their names, e. g. , the hyperparameters α. Random variables, e. g. , θ, are represented as circles. The N exchangeable variables $y_1, ..., y_N$ are represented as a single variable y_i in a rectangle. The number N at the corner specifies the number of repetitions of the rectangle. An arrow, e. g. , from α to θ denotes that the probability distribution of θ is conditioned on α, thus each of the N variables y_i depends on θ.

The distribution $P(\theta|\alpha)$ is called *prior*, which represents the uncertainty about parameters θ before seeing the data. Based on Bayes rule, we obtain the *posterior* distribution of θ given data OBS and hyperparameters α

$$P(\theta|OBS, \alpha) = \frac{P(OBS|\theta)P(\theta|\alpha)}{P(OBS|\alpha)}$$

Parameter θ is updated after seeing the observations OBS. The factor $P(OBS|\theta)$ is referred to as *likelihood*, and represents the probability that the model generates the data OBS given parameters θ. The likelihood of the data OBS can be unfolded.

$$P(OBS|\theta) = \prod_{i=1}^{N} P(y_i|\theta)$$

The factor $P(OBS|\alpha)$ is called *marginal likelihood* or evidence. It is calculated by

$$P(OBS|\alpha) = \int_{i=1}^{N} P(OBS|\theta)P(\theta|\alpha)d\theta$$

and can be viewed as a normalization factor to ensure $P(\theta|OBS, \alpha)d\theta = 1$.

Calculating the marginal distribution $P(\theta|OBS, \alpha)$ is in general computationally expensive. To reduce the computational cost a class of distributions denoted *exponential family* distributions are commonly used so that the computation is more efficient and can be executed in closed form. Members of this family include discrete and continuous distributions, such as Bernoulli, binomial, multinomial, Poisson and Gaussian, Gamma, Beta and Dirichlet distributions.

Each member of the exponential family has a simple *conjugate prior*, which is an important property for Bayesian analysis. If the likelihood distribution $P(OBS|\theta)$ of data OBS belongs to the exponential family, then there exists a conjugate prior, which is also in the exponential family. In Bayesian probability theory, a conjugate prior is a prior distribution which posterior distribution also takes the same mathematical form. For example, if the data is drawn from a multinomial distribution with unknown parameters θ and a conjugate prior (e. g. , the Dirichlet distribution) is assumed, then the posterior distribution of parameters θ is still distributed according to a Dirichlet distribution.

Usually, for computational efficiency a conjugate prior is assumed which reduces the function approximation to parameter approximation in computing the posterior distribution. However, the next challenge is that the mathematical form of the prior distribution is expected to be flexible enough not only to represent ones vague prior belief, but also to represent the learned posterior. This issue is addressed in non-parametric Bayesian models.

4.3.2 Non-parametric Bayesian Model

Often, the hardest modeling choice is to decide which likelihood distribution to use, especially to decide on the number of parameters for the model. If an unsuitable mathematical form is specified for the likelihood distribution, then the estimation will be divergent from the real situation, since the inference methods in the parametric models are closely connected with the specific functional forms of the distributions. To meet this challenge, *non-parametric Bayesian models* are used to learn the functions of interest directly from the data, e. g. , the probability distribution in the task of density estimation. The term non-parametric does not mean that there are no parameters in the models, but that the number and properties of the parameters are flexible and not fixed in advance. Non-parametric models are therefore also called distribution free.

Dirichlet Processes

In statistical machine learning, the most common non-parametric Bayesian models use *Dirichlet processes* (DP) or Gaussian processes. Dirichlet processes are used in discrete settings for density estimation and clustering, like in our application. The term process means that the degrees of freedom of the model are infinite. Of central importance in non-parametric framework are the unknown distribution G and its probability model which in our case are DPs.

DPs are generally denoted as $DP(\alpha_0, G_0)$, where α_0 and G_0 are the parameters. Replacing the parametric prior distribution with a sample from a DP, is called Dirichlet enhancement, which extends the flexibility of the parametric Bayesian modeling by encoding the additional uncertainty about the functional form of the prior. As an important result, Dirichlet enhanced models not only

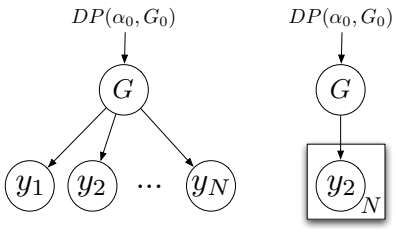

Figure 4.3: A non-parametric Bayesian model (left) and the same model in plate representation (right).

represent ones prior knowledge via the parameters of DP, i.e. α_0 and G_0, but also makes the prior G (e. g. , a sample distribution from a DP) as complex as necessary to model the real situation.

Let $\alpha_0 = \sum_{k=1}^{K} \alpha_k$, the Dirichlet distribution is defined as:

$$P(\theta_1, ..., \theta_K | \alpha) = \frac{\Gamma(\alpha_0)}{\prod_{k=1}^{K} \Gamma(\alpha_k)} \prod_{k=1}^{K} \theta_k^{\alpha_k - 1}$$

where Γ is the Gamma distribution. Note, that θ is a $K-1$ dimensional random vector, since $\sum_{k=1}^{K} \theta_k = 1$. If $K = 2$, a Dirichlet distribution reduces to a Beta distribution.

A Dirichlet distribution can be interpreted as a distribution over possible parameter vectors for a multinomial distribution with a predefined dimensionality $K-1$. Thus, it is in fact a distribution over distributions and at the same time the conjugate prior for the multinomial.

To relax the need for pre-specifying K, Dirichlet Processes (DP) are used. A DP is also a distribution on a set of distributions. It is indexed by two parameters: the base distribution G_0 and concentration parameter α_0. Let θ be random variables. G_0 is a probability distribution over the space of θ. α_0 is a positive real-value scalar. A random measure G is distributed according to a Dirichlet process with parameters G0 and α_0, if for all positive integer K and any partition $B_1, ..., B_K$ on the space of θ, the random probabilities $(G(B_1), ..., G(B_K))$ follows a Dirichlet distribution.

Illustration

In Fig. 4.3 a non-parametric Bayesian model is shown. In contrast to the parametric model in Fig. 4.2, the likelihood is an arbitrary (multinomial) distribution G drawn from $DP(\alpha_0, G_0)$, rather than a distribution with specific mathematical form and unknown parameters.

The two parameters of DPs can be explained intuitively. G_0 represents ones prior belief, α_0 measures the strength of ones belief in G_0. For large values of α_0, a sampled G is likely to be close to G_0. For small values of α_0, a sampled G is likely to put most of its probability mass on just a few atoms.

In discrete settings like the one this thesis the ith sample from a DP is drawn according to

$$y_i|y_1,..,y_{i-1} = \begin{cases} y_k^* & \text{with prob } \frac{num_{k-1}(y_k^*)}{n-1+\alpha} \\ \text{new draw from } G_0 & \text{with prob } \frac{\alpha}{n-1+\alpha} \end{cases}$$

where y_k^* stands for K unique values $k \in \{1,..,K\}$.

More intuitive interpretations of the generative process where proposed. The most popular approaches to draw samples from a DP are the Chinese restaurant process (see Sec. 5.2.2) and the truncated stick breaking construction (see Sec. 4.4.1).

4.3.3 Finite Mixture Model

A popular pattern for modeling relational settings are *mixture models* which also rely on DPs. Mixture models are well suited in situations where the samples are potentially generated under different conditions and are widely used for clustering and classification problems. The idea is to model the data as separate distributions, rather than building a single distribution. For instance, consider a distribution that models two different situation, y denotes the observations, θ_1 and θ_2 are the distribution parameters in the two situations, respectively. π is a 2 dimensional probability vector of being in one of the states. Then the distribution is represented as

$$P(y|\pi,\theta_1,\theta_2) = \pi P(y|\theta_1) + (1-\pi)P(y|\theta_2)$$

.

The atom distributions $P(y|\theta_1)$ and $P(y|\theta_2)$ are referred to as mixture components. When the atom distribution is parameterized, we can directly refer to the parameters (θ_1 and θ_2) as *mixture components*. The parameter *pi* is referred to as mixture proportion or mixture weight and specifies the proportion in which the atom distributions are mixed. The general finite mixture model is then:

$$P(y) = \sum_{k=1}^{K} \pi_k P(y|\theta_k)$$

See Fig. 4.4 (left) for a graphical illustration of a finite empirical mixture model. $\Theta = \theta_1,...,\theta_K$ are the K mixture components and π are the mixture weights. The parameters Θ and π are unknown but not random. Each observation y_i is associated with an auxiliary variable z_i, which specifies the mixture component from which the observation y_i is generated.

In empirical mixture models, the uncertainty in estimating the unknown parameters π and $\Theta = \theta_1,...,\theta_K$ is not considered. In the full Bayesian framework the unknown parameters are also viewed as random variables. The mixture proportions π are multinomial parameters, thus the conjugate prior is a Dirichlet distribution with hyperparameters $\alpha = (\alpha_1,...,\alpha_K)$, e. g. , $\pi \sim Dir(\cdot|\alpha)$. The mixture component θ_k denote the parameters of the distribution of observations with hidden state k. All θ_ks share a common prior G_0. Given the priors $Dir(\pi|\alpha)$ and G_0, the generative process of full-Bayesian mixture model is defined as follows:

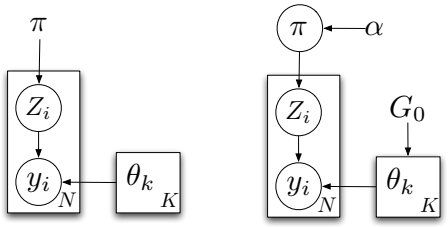

Figure 4.4: A finite empirical mixture model (left) and a finite full Bayesian mixture model (right).

$$\begin{aligned} \pi|\alpha &\sim Dir(\cdot|\alpha) \\ \theta_k|G_0 &\sim G_0(\cdot), k = 1, ..., K \\ z_i|\pi &\sim Mult(\cdot|\pi), i = 1, ..., N \\ y_i|z_i, \Theta &\sim P(\cdot|z_i, \Theta), i = 1, ..., N \end{aligned}$$

One problem in applying finite mixture models is the difficulty to decide the number of mixture components in advance. A principal solution for the problem is to embed the finite mixture model into a non-parametric Bayesian framework, such that the number of mixture components are flexible and can be optimized by the model itself based on the complexity of the data. In the next section we will introduce the infinite mixture models.

4.3.4 Infinite Mixture Model

A possible solution for the problem is to embed the finite mixture model in a non-parametric Bayesian framework using a DP resulting in a *infinite mixture model*. Then, the number of mixture components is not restricted and will be optimized with respect to the data in a self-organized way. The term *infinite* does not mean the number of mixture components are infinite, but the number is flexible and not fixed in advance. Due to the combination with DPs, the infinite mixture model is also referred to as Dirichlet process mixture model.

In a DP mixture model, the underlying sample θ_i generated from a DP is treated as the parameters of the distribution of the observation y_i. The model takes advantage of the discreteness property of Dirichlet processes, in particular, a distribution drawn from a DP places its probability mass on a countably infinite subset of the underlying sample (θ_i) space. The parameters θ_is are viewed as the hidden variables, one for each observation. They are indicators to specify which components the observations are generated from. The observations with identical parameter values are assumed to be members in the same cluster. Thus, DPs provides a clustering effect for the observations. Furthermore, the parameters for a new observation may take on existing values or new values, in specific the new observation is a member of an existing cluster or a member of a new cluster. That means new mixture components continue to emerge with additional data as many as necessary. Therefore the DP mixture model might have an infinite number of clusters and infer the structure of the data automatically and incrementally.

As illustrated in Fig. 4.5, the generative process is defined as:

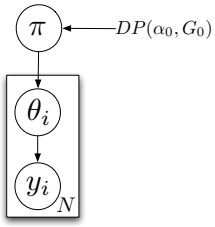

Figure 4.5: An infinite mixture model.

$$\begin{aligned} G|G_0, \alpha_0 &\sim DP(G_0, \alpha_0) \\ \theta_i|G &\sim G(\cdot), i = 1, ..., N \\ y_i|\theta_i &\sim P(\cdot|\theta_i). \end{aligned}$$

We draw a prior distribution G from a DP with hyperparameters α_0 and G_0, draw parameters $\Theta = \theta_1, ..., \theta_N$ from G, and draw observations conditioned on the corresponding parameters. The observations with identical parameter values are generated from the same mixture components, thus have the same cluster assignments. Suppose there are $K \leq N$ distinct values $\theta_1^*, ..., \theta_K^*$ in the N parameters. Then the DP mixture model partitions the observations into K groups in a nature way. The joint probability of the model is defined as:

$$P(G|\alpha_0, G_0) \prod_{i=1}^{N} P(\theta_i|G) P(y_i|\theta_i)$$

As with the non-parametric model there is a practical problem in DP mixture models to draw G directly from a DP given hyperparameters α_0 and G_0, since the probability function space is infinite. The most popular approaches to generative DP mixture models is the Chinese restaurant process (see Sec. 5.2.2), which directly draws the parameters θ_i with an auxiliary variable z_i. This solution supplies an explanation about the clustering effect of DP. Another approach is the stick breaking construction (see Sec. 4.4.1), which explicitly draws a random distribution G from a DP as a sum of infinite weighted components.

For inference in a DP mixture model, the commonly used solution is Markov chain Monte Carlo (MCMC) simulation methods, including collapsed Gibbs sampling, blocked Gibbs sampling and more (see Sec. 4.4.2, 4.6.3, 5.2.3).

4.4 Deductive and Inductive Inference

Based on the notation introduced in the last section we can now define the remaining variables, their probability distributions and model parameters for IHSM. The most important parameters in our case are shown in Fig. 4.6.

Z_k^c represents the latent variable of class c accompanied by π^c which is the *mixing weight* indicating the probability of an individual belonging to the components of Z. A^c is an attribute of class c with probability distribution θ^c. $R_{i,j}^b$ is a relation of class b between entity instances i, j with associated $\phi_{k,\ell}^b$, which indicates the correlation of relation class b between hidden state $Z_k^{c_i}$ and $Z_\ell^{c_j}$. The notation is summarized in Tab. 4.4 and will be explained in detail in the next section.

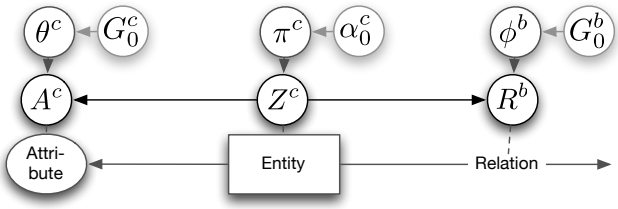

Figure 4.6: Parameters of an IHRM.

The IHSM is based on the idea that formal constraints can be imposed on the *correlation mixture component* $\phi_{k,\ell}$. and thus restrict possible truth values for the roles $R_{i,j}$. This, amongst others, imposes constraints on the structure of the underlying ground network or, more specifically in our application, the structure of the sociogram.

Recap the simple example from Sec. 4.1: According to this a person i known to be younger than 6 years old should not be attending any school j. The IHSM will extract this information from the ontology and set the correlation mixture component $\phi_{k,\ell}$ at entries representing according relations from $Person$ component k to $School$ component ℓ to 0. Here, k and ℓ denote the components the person i and the school j are assigned to. This eliminates inconsistent structural connection from the underlying ground network. More generally, all connections $R_{i,j}$ between two components k and ℓ where inconsistent individuals i and j are (partial) member of are considered void.

However, this redirection of relations by the latent variables allows IHSM not only to restrict possible connections in the ground network but makes this restriction influence the joint probability itself. By restricting ϕ, the mixing weights π are directly affected as well. Ultimately, cluster assignments Z are influenced and information can globally propagate through the network and influence all ϕ, π and θ indirectly (see Sec. 3.3.2).

While Sec. (3.3.2) focused on a conceptual description of IHRM and Sec. 4.3 focused on the Bayesian theory behind IHSM the algorithms for IHSM will be specified in detail in the next sections (4.4.1 - 4.4.3) before Sec. 4.6 presents experimental results.

4.4.1 Constrained Generative Model

To get a better understanding of the inference procedure of a graphical model and to form a conditional probability density function one commonly constructs a generative model first. A generative model is a model for randomly generating observable data, typically given some hidden parameters. It specifies a joint probability distribution over observation and label sequences. We will describe the generative model for the IHSM here.

In order to generate samples from a discrete latent model like the HRM a Dirichlet distribution is used. It is the conjugate prior of the categorical distribution used to represent the latent states. In an *infinite* HRM a Dirichlet process is needed. This topic is covered in Sec. 4.3, [Xu, 2007] and [Tresp, 2006]. There are mainly two methods to generate samples from a Dirichlet Process (DP) mixture model, i.e. the Chinese restaurant process (CRP) [Aldous, 1985]

(see Sec. 5.2.2) and the *Stick Breaking Construction* (SBC) [Sethuraman, 1994]. We will discuss how SBC can be applied to the IHSM.

There are C classes of objects and B classes of relations. The state k of Z_i^c specifies the cluster assignment of the concept (aka entity class) c. K denotes the number of clusters in Z. Here we consider the case that the object attributes and relations are drawn from exponential family distributions. Z is sampled from a multinomial distribution with parameter vector $\pi = (\pi_1, \ldots, \pi_K)$, which specifies the probability of a concept belonging to a component, i.e. $P(Z_i = k) = \pi_k$. π is referred to as mixing weights, and is drawn according to a truncated stick breaking construction with a hyperparameter α_0. The SBC [Ishwaran and James, 2001] is a representation of a DP, by which we can explicitly sample the random distributions of attribute parameters and relation parameters. α_0 is referred to as a *concentration parameter* in DP mixture modeling and acts as a tuning parameter that influences K. K is also limited by a truncation parameter that specifies the maximum number of components per cluster for each entity class.

Attributes A^c are generated from a Bernoulli distribution with parameters θ_k. For each component, there is an infinite number of mixture components θ_k. Each person in the component k inherits the mixture component, thus we have: $P(G_i = s|Z_i = k, \Theta) = \theta_{k,s}$. These mixture components are independently drawn from a prior G_0. The base distributions G_0^c and G_0^b are conjugated priors with hyperparameters β^c and β^b.

The truth values for the role $R_{i,j}$ involving two persons (i and j) are sampled from a binomial distribution with parameter $\phi_{k,\ell}$, where k and ℓ denote cluster assignments of the person i and the person j, respectively. $\phi_{k,\ell}^b$ is the correlation mixture component indexed by potentially infinite hidden states k for c_i and ℓ for c_j, where c_i and c_j are indexes of the individuals involved in the relation class b. Again, G_0^b is the Dirichlet Process base distribution of a role b. If an individual i is assigned to a component k, i.e. $Z_i = k$, the person inherits not only θ_k, but also $\phi_{k,\ell}, \ell = \{1, \ldots, K\}$.

The notation is summarized in Table 4.4).

To understand how this can be incorporated in the generative model of the IHRM we need to show how the SBC is applied to the IHSM. The SBC is a representation of a DP, by which we can explicitly sample the random distributions of attribute parameters θ and relation parameters ϕ.

1. For each object class c,
   (a) Draw mixing weights $\pi^c \sim \text{Stick}(\cdot|\alpha_0^c)$, defined as

$$V_k^c \overset{\text{iid}}{\sim} \text{Beta}(1, \alpha_0^c);$$
$$\pi_1^c = V_1^c, \quad \pi_k^c = V_k^c \prod_{k'=1}^{k-1} (1 - V_{k'}^c),\ k > 1. \tag{4.1}$$

   (b) Draw mixture components $\theta_k^c \sim G_0^c$, $k = 1, 2, \ldots$
2. For each object e_i^c in a class c,
   (a) Draw cluster assignment $Z_i^c \sim \text{Mult}(\cdot|\pi^c)$;
   (b) Draw object attributes $A_i^c \sim P(\cdot|\theta^c, Z_i^c)$.

Symbol	Description
c	entity class (e. g. , *Person*)
C	number of entity classes
N^c	number of entity instances in the class $c = \|c\|$
e_i^c	an entity instance indexed by i in class c
$R_{i,j}^b$	relation of class b between entity instances i, j
d^b	an relation instance in a relation class b
A^c	an attribute of class c
M^c	number of attributes of class $c = \|A^c\|$
Z_k^c	clustering / latent variable of class c indexed by k
K^c	number of components in clustering Z^c / number of states of the latent variable $Z^c = \|Z^c\|$
π^c	mixing weights of entity class c
α_0^c	concentration parameter of an entity class c
$\phi_{k,\ell}^b$	correlation mixture component of relation class b between hidden state $Z_k^{c_i}$ and $Z_\ell^{c_j}$. Index k indicates the component of latent variable Z^{c_i} and ℓ indicates the component of Z^{c_j}, where c_i and c_j are indexes of entity classes involved in the relation class b
G_0^b	base distribution of relation class b
β^b	parameters of the base distribution G_0^b
θ_k^c	mixture component indexed by the hidden state Z_k^c
G_0^c	base distribution of an object class c
β^c	parameters of the base distribution G_0^c

Table 4.4: Notation used for the IHSM.

3. For each role b between two concepts c_i and c_j, draw $\phi^b_{k,\ell} \sim G^b_0$ with component indices k for c_i and ℓ for c_j.

4. For objects $e_i^{c_i}$ and $e_j^{c_j}$ with a relation of class b, draw

$$R^b_{i,j} \sim P(\cdot|\phi^b, Z_i^{c_i}, Z_j^{c_j}).$$

5. Constrain ϕ to satisfiable relations:
For each role b and element of $\phi^b_{k,\ell}$

   (a) For entity cluster component Z_k and ℓ, let $F^b_d = \{(e_i, e_j) : r|e_i, e_j, R, i \neq j\}$ be the set of those generated relations $R_{i,j}$ between two different individuals e_i and e_j where e_i is assigned to component k and e_j is assigned to a component ℓ

   (b) Let F^b_e be the set of all generated entity instances e_k and e_ℓ in cluster component Z_k and Z_ℓ, respectively.

   (c) Let furthermore $\vartheta(F^b_e)$ be the set of all generated relations and entity instances d, e where the elements of F^b_e appear, i.e., $e \in \vartheta(F^b_e)$ iff $e \in F^b_e \wedge (e \equiv e : c \vee d \equiv (e, e_x) : R$ for arbitrary c, e_x and R.

   The following step checks whether the sampling of ϕ is consistent w.r.t. the given set of logical constraints IKB. The overall consistency check for $\phi^b_{k,\ell}$ yields a positive result *iff*

$$\exists\, \mathcal{I} \models F^b_d \cup F^b_e \cup \vartheta(F^b_e) \cup IKB \neq \emptyset$$

   where $\mathcal{I} \models X$ expresses that the set of logical sentences X is satisfiable, $\mathcal{I}$ being an interpretation.

   If the consistency check did not yield a positive result set

$$\phi^b_{k,\ell} = 0$$

6. Repeat step 4. to avoid inconsistent roles.

The basic property of the SBC is that the distributions of the parameters (θ^c_k and $\phi^b_{k,\ell}$) are sampled, e. g. , the distribution of θ^c_k can be represented as $G^c = \sum_{k=1}^{\infty} \pi^c_k \delta_{\theta^c_k}$, where $\delta_{\theta^c_k}$ is a distribution with a point mass on θ^c_k. In terms of this property, the SBC can sample objects independently; thus the SBC is efficient when a large domain is involved.

Due to fact that π is fixed after step 1.(a), the full potential of the constraining is not reflected in the generative model. This constrained generative model does guarantee a consistent model, but is basically the generative model of IHRM plus subsequent constraining. This limitation does not apply to the learned model, because in each Gibbs iteration π is altered according to the constrained ϕ. Thus, the constraining ultimately influences the clustering. This will be demonstrated next.

4.4.2 Constrained Learning

The key inferential problem of the IHSM is to compute the *joint posterior distribution* of unobservable variables given the data. In addition, we need to avoid inconsistent correlation mixture components ϕ during learning. As computation of the joint posterior is analytically intractable, approximate inference methods need to be considered to solve the problem.

We use *Markov chain Monte Carlo* (MCMC) sampling to approximate the posterior distribution. More specifically, we apply the *blocked Gibbs sampling* (GS) with truncated stick breaking representation [Ishwaran and James, 2001] a Markov chain Monte Carlo method to approximate the posterior . In the blocked GS, the posterior distributions of parameters (π^c, Θ^c and Φ^b) are explicitly sampled in the form of truncated stick breaking construction [Ishwaran and James, 2001]. The advantage is that given the posterior distributions, we can independently sample the hidden variables in a block, which highly accelerates the computation. The Markov chain is thus defined not only on the hidden variables, but also on the parameters. At the iteration t, the sampled variables include $Z_i^{c(t)}$, $\pi^{c(t)}$, $\Theta^{c(t)}$ and $\Phi^{b(t)}$.

Truncated stick breaking construction (TSB) fixes a value K^c for each class of objects and lets $V_{K^c}^c = 1$. That means the mixing weights π_k^c are equal to 0 for $k > K^c$. The number of the clusters is thus reduced to K^c. When K^c is large enough, the truncated Dirichlet process provides a close approximation to the true Dirichlet process [Ishwaran and James, 2001]. Note, that K^c is an additional parameter in the inference method. As before, let OBS be the set of all available observations (observed example data, each represented as a logical formula as defined in Sec. 4.2.2) under interpretation $\mathcal{I}$.

At each iteration, we first update the hidden variables conditioned on the parameters sampled in the last iteration, and then update the parameters conditioned on the hidden variables. So, for each entity class

1. Update hidden variable Z_i^c for each e_i^c: Assign to component with probability proportional to:

$$\pi_k^{c(t)} P(A_i^c | Z_i^{c(t+1)} = k, \theta^{c(t)}) \times \prod_{b'} \prod_{j'} P(R_{i,j'}^{b'} | Z_i^{c(t+1)} = k, Z_{j'}^{c_{j'}(t)}, \phi^{b'(t)})$$

2. Update $\pi^{c(t+1)}$ as follows:

   (a) Sample $v_k^{c(t+1)}$ from
   Beta$(\lambda_{k,1}^{c(t+1)}, \lambda_{k,2}^{c(t+1)})$ for $k = \{1, \ldots, K^c - 1\}$ with

$$\lambda_{k,1}^{c(t+1)} = 1 + \sum_{i=1}^{N^c} \delta_k(Z_i^{c(t+1)}),$$

$$\lambda_{k,2}^{c(t+1)} = \alpha_0^c + \sum_{k'=k+1}^{K^c} \sum_{i=1}^{N^c} \delta_{k'}(Z_i^{c(t+1)}),$$

   and set $v_{K^c}^{c(t+1)} = 1$. $\delta_k(Z_i^{c(t+1)})$ equals to 1 if $Z_i^{c(t+1)} = k$ and 0 otherwise.

(b) Compute $\pi^{c(t+1)}$ as: $\pi_1^{c(t+1)} = v_1^{c(t+1)}$ and

$$\pi_k^{c(t+1)} = v_k^{c(t+1)} \prod_{k'=1}^{k-1} (1 - v_{k'}^{c(t+1)}), \quad k > 1.$$

3. Update θ:

$$\theta_k^{c(t+1)} \sim P(\cdot | A^c, Z^{c(t+1)}, G_0^c)$$

4. Update $\phi_{k,\ell}^{b(t+1)}$ and constrain to satisfiable relations:

   For each role b and element of $\phi_{k,\ell}^b$

   (a) For entity cluster component Z_k and ℓ, let $F_d^b = \{(e_i, e_j) : r | e_i, e_j, R, i \neq j\}$ be the set of observed relations $R_{i,j}$ between two different individuals e_i and e_j where e_i is assigned to component k and e_j is assigned to a component ℓ

   (b) Let F_e^b be the set of all observed entity instances e_k and e_ℓ in cluster component Z_k and Z_ℓ, respectively.

   (c) Let furthermore $\vartheta(F_e^b)$ be the set of all observed relations and entity instances d, e where the elements of F_e^b appear, i.e., $e \in \vartheta(F_e^b)$ iff $e \in F_e^b \wedge (e \equiv e : c \vee d \equiv (e, e_x) : R$ for arbitrary c, e_x and R.

   The following step checks whether the sampling of ϕ is consistent w.r.t. the given set of logical constraints IKB. The overall consistency check for $\phi_{k,\ell}^{b(t+1)}$ yields a positive result *iff*

$$\exists\, \mathcal{I} \models F_d^b \cup F_e^b \cup \vartheta(F_e^b) \cup IKB \neq \emptyset$$

   where $\mathcal{I} \models X$ expresses that the set of logical sentences X is satisfiable, $\mathcal{I}$ being an interpretation.

   Where the consistency check described above yielded a positive result:

$$\phi_{k,\ell}^{b(t+1)} \sim P(\cdot | R^b, Z^{(t+1)}, G_0^b).$$

This algorithm is repeated for a number of Gibbs iterations.

4.4.3 Constrained Prediction

After the GS procedure reaches stationarity the role of interest is approximated by looking at the sampled values. After convergence (also known as burn-in period), we collect W samples to make predictions for the relations of interest.

Here, we mention the simple case first where the *predictive distribution* of the existence of a relation $R_{i,j}^b$ between to known individuals e_i, e_j is approximated by $\phi_{i',j'}^b$ where i' and j' denote the cluster assignments of the objects e_i and e_j in sample t.

$$P(R_{i,j}^b | \phi_{k,\ell}^b) = \frac{1}{W} \sum_{t=1}^{W} P(R_{i',j'}^b | \phi_{i',j'}^{b(t)})$$

Note, that $P(R^b_{i',j'}|\phi^{b(t)}_{i',j'}) = 0$ in every sample t if the existence of $R_{i,j}$ would violate any constraints. Thus, no prediction can result in an inconsistent IKB.

This is not the case if the predictive distribution of a relation $R^b_{new,j}$ between a new object e^c_{new} and a known object $e^{c_j}_j$ is approximated without repeating the whole learning process with e^c_{new} added to IKB. Note, that in blocked Gibbs sampling, the MCMC sequence is defined by hidden variables and parameters, including $Z^{c(t)}$, $\pi^{c(t)}$, $\Theta^{c(t)}$, and $\Phi^{b(t)}$. The predictive distribution of a relation $R^b_{new,j}$ between a new object e^c_{new} and a known object $e^{c_j}_j$ is approximated as

$$
\begin{aligned}
& P(R^b_{new,j}|OBS, \{\alpha^c_0, G^c_0\}^C_{c=1}, \{G^b_0\}^B_{b=1}) \\
\approx & \frac{1}{W}\sum_{t=1}^{W} P(R^b_{new,j}|OBS, \{Z^{c(t)}, \pi^{c(t)}, \Theta^{c(t)}\}^C_{c=1}, \{\Phi^{b(t)}\}^B_{b=1}) \\
\propto & \frac{1}{W}\sum_{t=1}^{W}\sum_{k=1}^{K^c} P(R^b_{new,j}|\phi^{b(t)}_{k,\ell})\ \pi^{c(t)}_k P(A^c_{new}|\theta^{c(t)}_k) \prod_{b'}\prod_{j'} P(R^{b'}_{new,j'}|\phi^{b'(t)}_{k,\ell'})
\end{aligned}
$$

where ℓ and ℓ' denote the cluster assignments of the objects j and j', respectively. The equation is quite intuitive. The prediction is a weighted sum of predictions $P(R^b_{new,j}|\phi^{b(t)}_{k,\ell})$ over all clusters. The weight of each cluster is the product of the last three terms, which represents to what extent this cluster agrees with the known data (attributes and relations) about the new object. Since the blocked method also samples parameters, the computation is straightforward.

Since the assignment of e^c_{new} to any cluster k might lead to inconsistent predictions due to the fact that e^c_{new}-governing constraints might not be present in any other instance assigned to k, the whole parameters need to be learned again. A more pragmatic approach, leading to similar results, would constrain the prediction afterwards as done in IHRM+C.

4.4.4 Implications

The ultimate goal of IHSM is to group entities into clusters. A good set of partitions allows to predict the parameters ϕ by their mere cluster assignments and does not allow inconsistent predictions. In the ground truth, our model assumes that each entity belongs to exactly one cluster. It simultaneously discovers clusters and the relations in-between clusters that are best supported by the data, ignoring irrelevant attributes.

Although the value of attributes is determined entirely by the cluster assignment of associated entities, there is no need for direct dependencies between attributes or extensive structural learning. The cluster assessment of a entity is influenced by all corresponding attributes and cluster assessments of related entities and the constraints specified in the formal knowledge base. This way information can propagate through the whole network while the infinite hidden variables Z act as 'hubs'.

This allows for a collaborative filtering effect. Cross-attribute and cross-entity dependencies can be learned, something which is not possible with a 'flat' propositional approach that assumes independent and identical distributed (i.i.d.) data. At the same time the number of clusters does *not* need to be

fixed in advance. Thus, it can be guaranteed that the representational power is unrestricted.

4.5 Implementation and Data

The increasing popularity of *social networking* services like MySpace[2] and Facebook[3] has fostered research on social network analysis in the last years. The immense number of user profiles demands for automated and intelligent data management capabilities, e. g. , formal ontologies.

In particular, recent applications of data mining techniques have been focused on social networks. While data mining techniques can handle large amounts of simple facts, little effort has been made to exploit the semantic information inherent in social networks and user profiles. There is almost no work on statistical relational learning with formal ontologies in general and with SW data in particular. The lack of experiments on large and complex real world ontologies is not only due to the absence of algorithms but also due to missing suitable datasets. In this section we will present both, a large and complex SW dataset and the methodology of how to apply IHSM in practice. Ultimately, we evaluate our approach by presenting results of an empirical comparison of IHSM and IHRM in this domain.

As mentioned before our core ontology is based on Friend of a Friend (FOAF) data. The purpose of the FOAF project is to create a web of machine-readable pages describing people, the links between them and the things they create and do. The FOAF ontology is defined using OWL DL/RDF(S) and formally specified in the FOAF Vocabulary Specification 0.91[4]. In addition, we make use of further concepts and roles which are available in the data (see Sec. 4.2.2). We gathered our FOAF dataset from user profiles of the community website LiveJournal.com[5] (This specific ontology will be called *LJ-FOAF* from now on).

In our case we deliberately stopped crawling the site after gathering 32,062 FOAF-profiles although the LiveJournal.com community is by far larger (see LiveJournal.com[6]). Statistics of the downloaded data set are shown in Table 4.5. Note that relations like *knows* are typically very sparse in social network.

In addition we used only a subset of the data for our experiments. This has two reasons: First, the approach described in this thesis is not concerned with scalability. This challenge is the focus of another line of research we are working on. For more information on scalability, data retrieval and sampling strategies see [Huang et al., 2009] or for more general coverage of crawling the semantic web see [Hausenblas et al., 2008]. We chose the size of the dataset to be convenient for experiments with training times of under 1 hour. Second, restricting the number of profiles, we already have a representative situation for the SW: Only knowing a subset of all possible instances in a population is an intrinsic characteristic of the Web. Because the only way to gather data is by following links, one cannot reach unlinked parts of the population. In this case

[2]http://www.myspace.com/
[3]http://www.facebook.com/
[4]http://xmlns.com/foaf/spec/
[5]http://www.livejournal.com/bots/
[6]http://www.livejournal.com/stats.bml

Concept	#Individuals
$Person$	32,062
$Location$	5,673
$School$	15,744
$Interest$	4,695
$On.ChatAcc.$	5
$Date$	4
$\#BlogPosts$	5
Role	**#Instances**
$knows$	530,831
$(sparsity)$	0.05%
$residence$	24,368
$attends$	31,507
has	9,616
$holds$	19,021
$dateOfBirth$	10,040
$posted$	31,959

Table 4.5: Number of individuals and number of instantiated relations in the LJ-FOAF set.

one does not know that this part exists. Considering this, the subset we are using exhibits this unavoidable characteristic and represents a typical situation on the SW. This size is appropriate to test the generalization ability of our learning algorithms to new data.

All extracted concepts and roles are shown in Fig. 3.1. Tab. 4.6 lists the number of different individuals (left column) and their known instantiated roles (middle column) used for experiments. Please note that $Date$ and $\#BlogPosts$ are reduced to a small number of discrete states. As expected for a social networks $knows$ is the primary source of information. This real world data set offers both, a sufficiently large set of individuals for inductive learning and a formal ontology specified in RDFS and OWL. However, while LJ-FOAF offers a taxonomy there are no complex constraints given. Thus, to demonstrate the full potential of IHSM, we additionally added constraints that are not given in the original ontology (see Sec. 4.2.2).

To implement all features of IHSM we made use of additional open source software packages: Protege[7] was used to adjust the FOAF ontology to OWL DL and add additional axioms. The SW framework Jena[8] is used to load, store and query the ontology and Pellet[9] provides the OWL DL reasoning capabilities. The Gibbs sampling procedure was implemented in Java with the help of Colt[10] an open source library for high performance scientific and technical computing.

This outlines the workflow: First, the TBox axioms are designed and loaded into Jena. Next, all ABox assertions are added and loaded into Jena. Then, by using the taxonomy information from the ontology and the ABox assertions we extract the RM as described in Sec. 3.3.1. This RM is transferred into a

[7] http://protege.stanford.edu/
[8] http://jena.sourceforge.net/
[9] http://pellet.owldl.com/
[10] http://acs.lbl.gov/ hoschek/colt/

Concept	#Individuals	Role	#Instances
Location	200	*residence*	514
School	747	*attends*	963
OnlineChatAccount	5	*holdsAccount*	427
Person	638	*knows*	8069
		hasImage	574
Date	4	*dateOfBirth*	194
#BlogPosts	5	*posted*	629

Table 4.6: Number of individuals, no. of instantiated roles in the reduced LJ-FOAF data set

IHSM by adding hidden variables and parameters, accordingly. Finally, the parameters are learned from the data, while constraints are constantly checked as described in Sec. 4.4.2. The trained model can now be used for statistical analysis like prediction of unknown relation instances.

In our experiments the standard setting for the truncation parameter were $\#Individuals/10$ for entity classes with over 100 instances and $\#Individuals$ for entity classes with less individuals. The standard iterations of the Gibbs sampler are 2000. We did not engage in extensive parameter tuning because the purpose of our experiments is mainly to examine the influence of the constraints and not to find the optimal predictive performance compared to other relational learning algorithms. Thus, we fixed $\alpha_0 = 5$ for every entity class and $\beta_0 = 20$ for every relation class.

4.6 Experimental Results

We will now report our results on learning and constraining with IHSM mainly on the LJ-FOAF social network data set. We focus on investigating the difference between IHRM and IHSM, as there are - to the best of our knowledge - no comparable constrained multi-relational algorithms. However, we compare IHRM to related algorithms.

First, the computational complexity of IHSM is investigated (Sec. 4.6.1), before information extraction by latent clusters is analyzed (Sec. 4.6.2). Finally, the predictive performances for link prediction tasks of different approaches are examined (see Sec. 4.6.3). We proceed by reporting results from a movie recommendation data set of different inference methods for IHRM as well as IHRM to other single relational learning algorithms. This is intended to demonstrate that IHRM shows a competitive performance to related approaches in an unconstrained setting. On this basis, IHSM is compared to IHRM to investigate the influence of constraining on the predictive performance.

4.6.1 Computational Complexity:

The additional consistency check for every individual per iteration made training slower by approximately a factor of 6 if performed with Jena and Pellet. Pellet deploys a Tableaux-method for consistency checks as mentioned in Sec. 2.3.3.

After implementing a non-generic constraining module that is optimized for the simple example introduced in Sec. 4.1 we could reduce the additional com-

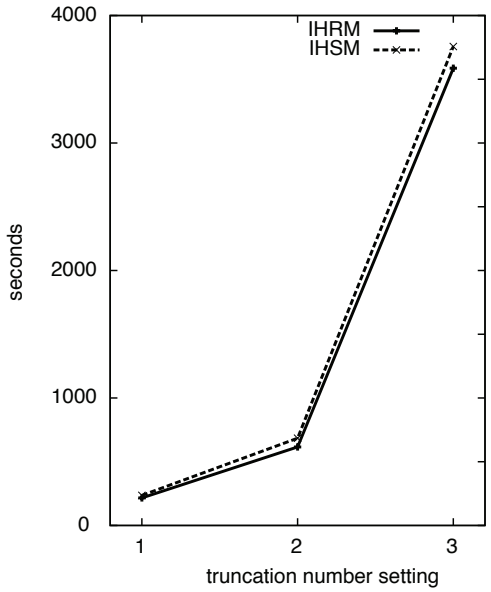

Figure 4.7: Run time of IHRM vs. IHSM if using our optimized constraining implementation.

putations considerably. Fig. 4.7 shows, that there is almost no computational overhead for our own implementation. We assume that this is due to the fact that Pellet can not cache any intermediate results. Every iteration a new set of individuals is checked against the complete TBox, thus the Tableaux is generated each time from scratch. In our own implementation only the relevant subset of the TBox is checked and intermediate results of known ABox combinations are cached. However, this makes the procedure less general.

A comparison between IHSM and IHRM for different truncation parameter settings for our implementation is given in Fig. 4.7. As mentioned before, the performance stays almost identical. Besides that, it is interesting to see that allowing a more flexible cluster size by increasing the truncation parameter increases the running time considerably.

Evaluating the convergence of the cluster sizes is another interesting aspect in the comparison of IHSM and IHRM. Fig. 4.8 shows the number of individuals for the two largest components of the entity cluster Z^{Person} plotted over Gibbs sampler iterations for one exemplary training run.

Most importantly, the smooth curvature indicates that constraining does not affect the convergence speed which is desirable. Concerning the average number of individuals per cluster we could not find any statistical significant differences between IHSM and IHRM. In this case the largest component has more individuals after the first 100 iterations. However, our observations showed that this usually converges to a comparable number after a 1000 iterations for both largest components. In contrast, IHRM has a tendency to produce a larger number of components per cluster, which is discussed in the next Sec. 4.6.2.

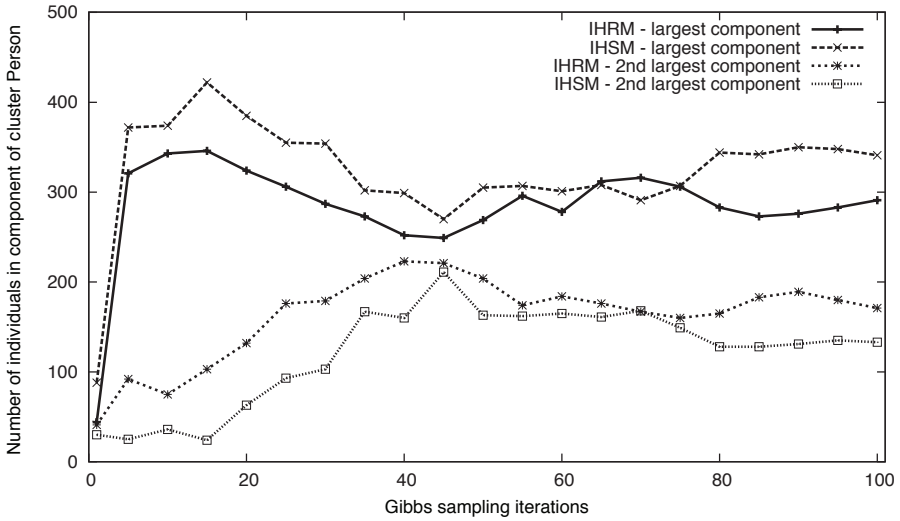

Figure 4.8: Convergence of number of individuals of the two largest components in the Person cluster.

4.6.2 Cluster Analysis:

Investigating the result of the latent classes learned by the algorithm can provide interesting information about the domain.One interesting outcome of the comparison of IHRM and IHSM is the number of components per hidden variable after convergence (see Table 4.7 right column). In both cases, if compared to the initialization, the algorithms converged to a much smaller number of components. Most of the individuals were assigned to a few distinct components leaving most of the remaining components almost empty.

However, there is a noticeable difference between IHRM and IHSM concerning the concepts *School* and *Person* which needed more components after training with IHSM (see bold numbers in Table 4.7). A closer analysis of the components revealed that IHSM generated additional components for inconsistent individuals, because both concepts are affected by constraints. Inconsistent individuals are in this example persons that specified both, being under the age of 6 and attending a school.

In contrast, the last concept affected by the constraints (*Date*) shows an opposite effect if constraining is used: the results show fewer components. Here, IHSM divided more generally into age groups 'too young' and 'old enough' which also reflects the constraints.

All things considered, this demonstrates that the restriction of roles in form of consistency checks does influence the states of the latent variables. In this example the changes to the states appear to be intuitive.

Another interesting outcome that can be analyzed is how the constraining changes the probability of uncertain relations. Fig. 4.10 compares the learned parameter ϕ^{attend} of IHRM to the one learned by IHSM. A brighter cell indicates stronger relations between two components. Although hard to generalize, a cell with 50% gray might indicate that no significant probabilistic dependencies for individuals in this component are found in the data.

Concept	Role	#Compo. IHRM	#Compo. IHSM
Location	*residence*	18	17
School	*attends*	**36**	**48**
OnlineChatAccount	*holdsAccount*	4	4
Person	*knows*	**38**	**45**
Date	*dateOfBirth*	**4**	**2**
#BlogPosts	*posted* 4	4	

Table 4.7: Number of components found.

The most obvious results are the rows with black cells which represent *Person* components that have no relation to any school. In fact, all of those cells contained at least one persons that conflicted with the ontology by having specified an age under 5. This illustrate that one of the main goals of IHSM is achieved, namely the exploitation of constraints provided by the ontology and the constraining of inconsistent relations.

Note, that the learned clusters can also be used to extract meta-knowledge from the data, similar to latent topics that are extracted from documents in LDA (cmp. [Bundschus et al., 2009]). The lower dimensional latent space can be used to define more general concepts. This abstraction can be used to simplify reasoning and inference. How this latent structure could be transformed into symbolic knowledge and feed back to the ontology is a promising direction of future research.

4.6.3 Predictive Performance:

We focus on social network analysis as a possible application of IHSM. In this domain one could for instance want to predict 'who knows who' in case either this information is unknown or the systems wants to recommend new friendships. Other relations that could be interesting to predict in case they are unknown are the school someone attends/attended or the place he lives/lived. Furthermore one could want to predict unspecified attributes of certain persons, like their age.

Our strategy for evaluating the predictive performance of missing relations is separated in 2 steps. First, we report results that demonstrate that the algorithm without constraining (IHRM) is competitive if compared to other relational learning algorithms (see Sec. 4.6.3). Then, we compare the difference of IHRM to IHSM with constraining (see Sec. 4.6.3). The reason why we do not directly compare IHSM to other approaches is that there are - to the best of our knowledge - no comparable standard constrained multi-relational learning algorithms.

IHRM vs. Relational Learning Algorithms

Here, we report results of IHRM compared to other single relational learning algorithms if applied to social network analysis. IHRM has been evaluated on other domains like medical or gene databases as well (see [Xu et al., 2006]). The experiments presented in this section have been published in [Xu et al., 2009].

We first evaluate the IHRM on the MovieLens data [Sarwar et al., 2000b]. There are in total 943 users and 1680 movies, and we obtain 702 users and 603

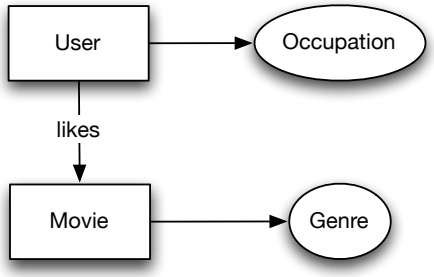

Figure 4.9: An ER-model for movie a recommendation system. For readability, only two attributes (user's occupation and movie's genre) are shown in the figure.

movies after removing low-frequent ones. Each user has about 112 ratings on average.

The model is shown in Figure 4.9. There are two classes of objects (*Users* and *Movies*) and one class of relations (*like*). The task is to predict preferences of users. The users have attributes *Age*, *Gender*, *Occupation* and the movies have attributes *Published-year*, *Genres* and so on. The relations have two states, where $R = 1$ indicates that the user likes the movie and 0 otherwise. The user ratings in MovieLens are originally based on a five-star scale, so we transfer each rating to binary value with $R = 1$ if the rating is higher than the user's average rating, vice versa. To evaluate the predictive performance, we perform 4 sets of experiments which respectively select 5, 10, 15 and 20 ratings for each test user as the known ratings, and predict the remaining ratings. These experiments are referred to as *given5*, *given10*, *given15* and *given20* in the following. For testing the relation is predicted to exist (i.e., $R = 1$) if the predictive probability is larger than a threshold $\varepsilon = 0.5$.

We implement the following 3 inference methods for IHRM: Chinese restaurant process Gibbs sampling (CRPGS), truncated stick-breaking Gibbs sampling (TSBGS), and the corresponding mean field method TSBMF. The truncation parameters Ks for TSBGS and TSBMF are initially set to be the number of entities. For TSBMF we consider $\alpha_0 = \{5, 10, 100, 1000\}$, and obtain the best prediction when $\alpha_0 = 100$. For CRPGS and TSBGS α_0 is 100. For the variational methods, the change of variational parameters between two iterations is monitored to determine the convergence. For the Gibbs samplers, the convergence was analyzed by three measures: Geweke statistic on likelihood, Geweke statistic on the number of components for each class of objects, and autocorrelation. For CRPGS, the first $w = 50$ iterations are discarded as burn-in period, and the last $W = 1400$ iterations are collected to approximate the predictive distributions. For TSBGS, we have $w = 300$ and $W = 1700$.

The prediction results are shown in Table 4.8. All IHRM inference methods under consideration achieve comparably good performance; the best results are achieved by the two Gibbs sampling methods. To demonstrate the performance of the IHRM, we also implement Pearson-coefficient based collaborative filtering (CF) method [Resnick et al., 1994b] and an SVD-based CF method [Sarwar et al., 2000a]. It is clear that the IHRM outperforms the traditional CF methods, especially when there are few known ratings for the test users. The main

	CRPGS	TSBGS	TSBMF	Pearson	SVD
Given5	65.13	65.51	65.26	57.81	63.72
Given10	65.71	66.35	65.83	60.04	63.97
Given15	66.73	67.82	66.54	61.25	64.49
Given20	68.53	68.27	67.63	62.41	65.13
Time(s)	164993	33770	2892	-	-
Time/iteration	109	17	19	-	-
#Components user	47	59	9	-	-
#Components movie	77	44	6	-	-

Table 4.8: Performance of different inference methods of IHRM on MovieLens data.

advantage of the IHRM is that it can exploit attribute information. If the attribute information is removed, the performance of the IHRM becomes close to the performance of the SVD approach. For example, after ignoring all attribute information, the TSBMF generates the predictive results: 64.55% for Given5, 65.45% for Given10, 65.90% for Given15, and 66.79% for Given20.

IHRM vs. IHSM

The purpose of this section is not to show superior predictive performance of IHSM compared to other multi-relational relational learning algorithms. This has been evaluated in the previous section. In addition, the comparison of IHSM to other first-order probabilistic learning algorithms (see Sec. 3.4) is difficult due to the fact that only IHSM uses hard constraints, as described in Sec. 4.1.1. Thus, our main concern is to show the influence of the constraining on the results of the learning process, as it was examined by [Punyakanok et al., 2005] (cmp. Sec. 4.1.2). In particular, we want to show the influence of constraining on the predictive performance for IHSM compared to IHRM. In addition, we show the performance for IHRM with a subsequent constraining of the predictions. Thus, inconsistent predictions are also avoided, however the learned model remains the same. This setup is denoted IHRM+C.

We ran a 5-fold cross validation to evaluate the predictions of different relation classes. In specific, the non-zero entries of the relation matrix to be predicted were randomly split in 5 parts. Each part was once used for testing while the remaining parts were used for training. The entries of each testing part were set to zero (unknown) for training and to their actual value of 1 for testing. Each fold was trained with 1000 iterations of the Gibbs sampler, where 500 iterations are discarded as the burn-in period. After this, the learned parameters are recorded every 50th iteration. In the end we use the 10 recorded parameter sets to predict the unknown relation values, average over them and calculate the area under the ROC curve (AUC) as our evaluation measure[11]. Finally we average over the 5 folds and calculate the 95% confidence interval.

The obvious roles to evaluate are *attends* and $dateOfBirth$. Both are constrained by the ontology, so IHSM should have an advantage over IHRM because

[11]Please note that SW data has no negative samples, because zero entries do not represent negative relations but unknown ones (open world assumption). Still, the AUC is appropriate because it has been shown to be a useful measure for probabilistic predictions of binary classification on imbalanced data sets.

Role	attends	dateOfBirth	knows
IHRM	0.577 (±0.013)	0.548 (±0.018)	0.813 (±0.005)
IHRM+C	0.581 (±0.012)	0.549 (±0.016)	0.814 (±0.006)
IHSM	**0.608** (±0.017)	**0.561** (±0.011)	**0.824** (±0.002)

Table 4.9: Predictive performance for different LJ-FOAF roles: AUC and 95% confidence intervals

it cannot predict any false positives. The results in Table 4.9 confirm this observation. In both cases IHSM did outperform IHRM. Interestingly, IHRM+C only gives a slightly improved performance. This is due to the fact, that still a unconstrained model is learned. Thus, only a few inconsistent predictions are avoided, but no global model consistent with the constraints is learned. This confirms the results of [Punyakanok et al., 2005].

A less obvious outcome can be examined from the influence of the constraining on a relation that is not directly constrained by the ontology like *knows*. Still, in our experiments IHSM showed a slight advantage over IHRM. Thus, there seems to be a positive influence of the background knowledge, although a lot of users specify an incorrect age. However, there is the potential that the opposite may occur likewise. If the given constraints are conflicting with the empirical evidence there could even be a decrease in predictive performance. It is the ontology designers choice to decide whether to enforce a constraint that conflicts with the observed evidence.

Considering the numerous ongoing efforts concerning ontology learning for the SW more data sets with complex ontologies should become available in the near future. Thus, we expect to achieve more definite results of IHSM in those domains.

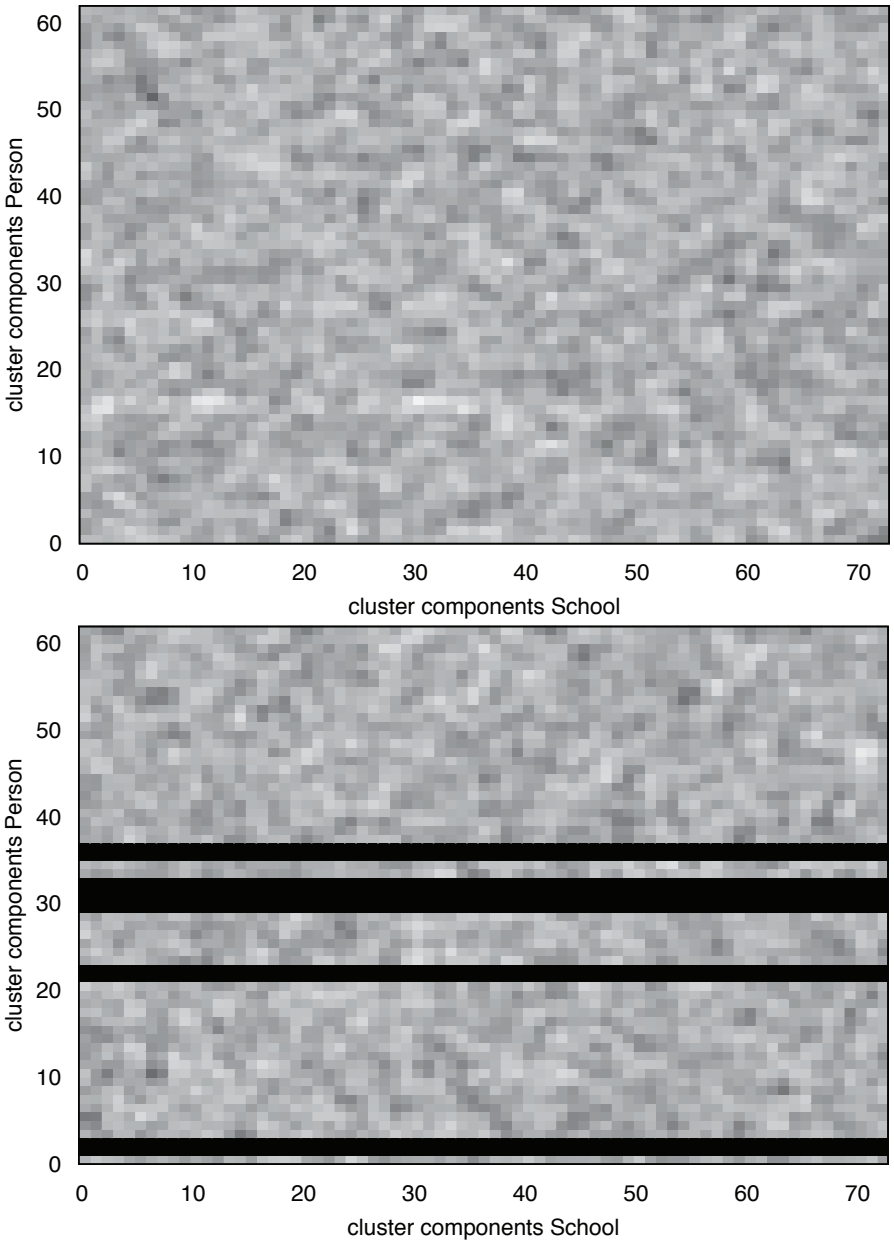

Figure 4.10: Correlation mixture component ϕ^{attend} for each combination of components Z^{Person} and Z^{School}. Top: without constraining (IHRM). Bottom: with constraining (IHSM).

Chapter 5

The IHSM for Computational Trust Learning

This chapter presents the final scientific contribution of this thesis. It addresses the problem of handling ontologies which comprise statements from multiple, potentially untrustworthy sources. This situation is common in many real world systems where content is generated in an uncontrolled and collaborative manner.

Such *collaborative knowledge platforms* have become exceedingly popular on the internet and are the key idea of web 2.0 and the Semantic Web (SW). Here, we address the issue of how a user (trustor) can assess his expectations regarding another user's trustworthiness especially in certain statements provided by this potentially untrustworthy user (trustee). We propose a way how such trust situations can be modeled using IHSM and how trust based on past observations and context information can be learned and integrated into the ontology.

After discussing some recent trends and motivating research on Computational Trust (CT) (see Sec. 5.1) we give a broad interdisciplinary overview from different perspectives on how trust can be defined and formalized (see Sec. 5.1.1). Next, we discuss related work on Computational Trust Learning (CTL) (see Sec. 5.1.2), before we define the concrete situation of *uncertain ontologies* and show how to model and learn context-sensitive relational trust in this context using our *Infinite Hidden Semantic Trust Model* (IHSTM) (see Sec. 5.2).

The practicability and effectiveness of this approach is evaluated empirically (see Sec. 5.4) on three different data sets including user-ratings gathered from eBay (see Sec. 5.3 for the experimental setup). Our results suggest that (i) the inherent clustering allows the trustor to characterize the structure of a trust-situation and provides meaningful trust assessments (see Sec. 5.4.1); (ii) utilizing the collaborative filtering effect associated with relational data does improve trust assessment performance (see Sec. 5.4.2); (iii) by learning faster and transferring knowledge more effectively we improve cold start performance and can cope better with dynamic behavior in open multiagent systems. The later is demonstrated with interactions recorded from a strategic two-player negotiation scenario (see Sec. 5.4.3).

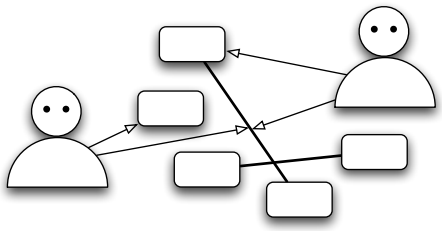

Figure 5.1: Collaborative knowledge platform with facts provided by potentially untrustworthy sources

5.1 Motivation and Related Work

The problem of untrustworthy information inherent in collaborative knowledge platforms has become relevant since open systems like the internet have been established. In recent years, the issue has gained importance as the concept of user generated content has become the core idea of web 2.0 and platforms like Delicious[1], Flickr[2], Wikipedia[3], Blogger[4] and Facebook[5] have become popular.

This trend is expected to continue in the future when the ideas of the SW have been put into practice in a wider range. To face those issues, concepts related to 'trust' are being proposed. For instance, computable measures for trust are developed and integrated into the SW stack [Horrocks et al., 2005]. Thus, a growing research community is concerned with topics like privacy, *provenance*, authentification, web of trust and many related concepts.

What all those scenarios have in common is that users, or any interacting entity in such systems which we will call *agents*, show a highly contingent behavior. Moreover, it is often not feasible to implement effective mechanisms to enforce socially fair behavior as pursued in mechanism design or preference aggregation. A potential solution to these problems is the transfer of the human notion of trust to a machine-computable model, realizing *Computational Trust* (CT).

Although CT has been a focus of research in Artificial Intelligence for several years (for an overview see [Ramchurn et al., 2004]), current approaches still lack certain features of human trustability assessment which we consider to be of high importance for the computational determination of trust values in open systems. For instance recent studies in psychology (see [Willis and Todorov, 2006]) have shown that people can robustly draw trait inferences like trustworthiness from the mere facial appearance of unknown people after a split second. Although seemingly neither the time span nor the available information allow to make a well-founded judgement, the derived trust (or distrust) provides after all a foundation for immediate decision making, and a significant reduction of social complexity especially under time pressure (see [Luhmann, 2001]).

Whereas the 'quality' of such so-called *initial trust* (i.e., trusting someone without having accumulated enough experiences from relevant past behavior

[1] http://delicious.com/
[2] http://www.flickr.com/
[3] http://www.wikipedia.org/
[4] http://www.blogger.com
[5] http://www.facebook.com/

of the trustee) might be limited in the described scenario, this example shows that humans are able to estimate the trustability of others using information which are at a first glance unrelated to the derived expectation. (e. g. , the facial appearance, or any contextual information in general).

In contrast, the vast majority of approaches to empirical CTL in artificial intelligence lack this ability, as these approaches strongly rest on well-defined past experiences with the trustee, from which it is directly concluded that the trustee will behave in the future as he did in the past, regardless of the concrete context (see Sec. 5.1.2 for related work).

These approaches come to their limits in cases where the trustor could not make such experiences and thus has to rely on 'second order' information such as the context of the respective encounter instead. In order to make such initial trust computationally feasible, we not only need to relate trust values to a specific context, but we also need to provide a mechanism in order to take over contextualized trust to a new, possibly somewhat different context.

In particular, the general requirements that we are concerned with are:

Context sensitivity and trust transfer: Contextual information that might be related to the trust decision to be made needs to be incorporated. This shall include attributes of the person one needs to trust, attributes of the external circumstances under which the trust assessment is made, and actions and promises the person has given to seek one's confidence. Furthermore, specific trust values gained in a certain context need to be *transferrable* to new, unknown 'trigger' situations.

Multi-dimensionality: Most trust models assign a single trust value per agent. This ignores the fact that human trust decisions are made in relation to a whole spectrum of aspects (e. g. , what a person is likely to do, such as the expected outcome of some information trading, even in the same context). For instance a certain information supplier agent might be trustworthy in terms of up-to-dateness, but not in terms of information quality (e. g. , precision, credibility...). Combining several trust related measures as in our approach is considerably much more flexible. In contrast, most existing approaches to trust still relate trust to 'whole persons' only instead of to their contextualized behavior.

Here, we focus on *interaction-trust* (i.e., (dis-)trust formed by agents during the course of an interaction regarding their opponents' behavior) in order to tailor our model to the specifics of the probably most relevant application field for empirical trustability assessment.

5.1.1 Trust Basics

Trust is an inherently multidisciplinary concept that is the focus of research in various different disciplines. To get an overview of the trust concepts that are relevant for computational trust we focused mainly on work from the field of *multiagent systems* (see e. g. , [Klos and la Poutré, 2005], [Ramchurn et al., 2004] and system management (see e. g. , [Herrmann et al., 2005], [Jensen et al., 2004]). This section will treat different basic aspects of trust that are important for computational trust learning.

Definitions of Trust

Various areas of science are currently concerned with trust. Examples are political sciences, theology, philosophy, sociology, psychology, economics, computer science and information technologies. Obviously, there is no established definition of trust. This is also true for computer science and the concept of CT. It turns out that the attempt to narrow down the concept of trust is most promising when relating trust to the context of a concrete application

For the development of a CTL approach we found the following informal specifications of trust useful:

> "Trust is the extent to which one party is willing to participate in a given action with a given partner, considering the risks and incentives involved [Ruohomaa and Kutvonen, 2005]."

> "Trust is a certain extend of subjective probability that an actor assumes that a certain action will be executed by another actor [Gambetta, 2001]."

> "Trust is a belief an agent has that the other party will do what it says it will (being honest and reliable) or reciprocate (being reciprocative for the common good of both), given an opportunity to defect to get higher payoffs. [Ramchurn et al., 2004]."

Starting from this initial specifications we developed a formal definition of trust that is most suitable for our purpose and applications to ontologies (see Sec. 5.2.1).

Relevant Related Concepts

Previous work not only holds a wide range of approaches to the basic concept of trust it also offers various additional concepts which are closely related to trust (cmp. [Gambetta, 2001]). The most important concepts concerning the prediction of future actions of interacting agents are concepts like reliability, expectability and credibility. The implementation of this sort of 'predictability' is often achieved with the help of further concepts like experience and reputation which allow to draw conclusions about the interacting entities.

In situations where actions of agents cannot be observed or are only partially observable concepts like confidence or belief are used. Besides that, there are applications where trust-specific elements of uncertainty need to be considered. Concepts discussed in previous work cover e. g. , risk, doubt, mistrust, distrust and untrust.

Levels of Trust

An important distinction that can be made regarding computational trust in multiagent systems is *individual level trust* vs. *system level trust*. This separates trust between interacting agents and trust in a system on a global scale. System level trust denotes the trust that is enforced by rules and regulations of the system, like penalties for breach of contract or authentication mechanisms. System level trust rules are intended to reduce abuse and unanticipated behavior.

In a hypothetical system with perfect system level trust individual level trust is not needed anymore. However, the examples mentioned in the beginning of this section have shown that in real world open systems, like the shared ontologies covered in this thesis, trust needs to be achieved on the individual level. This implies the use of a decentralized approach where every agent has to have its own mechanism to assess trust values. Without that, participants in such systems would not be capable of making decisions and acting autonomously.

In this thesis, we see both levels of trust as complementary. System level trust is needed to some extend, as an identification mechanism for agents. In the case of ontologies this comprises an identifier for the origin of information, also known as *provenance* (see [Carroll et al., 2005]). However, as just mentioned, open knowledge systems need an individual level trust mechanism as well.

Relevance of Trust

Publications in the area of trust research agree on the assumption that coordination between agents is not possible without a trust mechanism. The following three examplary quotes stress the importance of trust:

> "Nowadays the importance of trust in electronic interactions is out of discussion [Carbo et al., 2005]."

> "Trust models have emerged as an important risk mechanism in online environments [Fatima et al., 2005]."

> "Trust is thought to be the essential glue that holds societies together [Kimbrough, 2005]."

As we can see, trust can be thought of as an effective mechanism for the reduction of complexity in situations with uncertain or insufficient knowledge. It allows for assessment of information and decision making.

The reasons for limited trustworthiness of information can have various reasons. The agent to be trusted - in case of ontologies the one providing the information - might have limited knowledge himself or might deliberately provide incorrect information. The means of communication can be limited, the context might be uncertain or the other agent might be unknown. Besides that, uncertainty and inconsistencies are unavoidable in communication in general and in situations where different knowledge sources need to interact or be combined.

Crucial for the value of a trust model is its ability to perform well in so called *initial trust* situations. This involves mechanisms for trust assessment in situations mostly unknown to the agent.

Computational Trust vs. Human Trust

While the main objective of *computational trust* is to allow rational decision making, a defining characteristic of *human trust* is its indeterminableness. However, in our opinion human trust and its main objective as a mechanism for complexity reduction in uncertain and unknown situations still has not yet found an equivalence in computational trust. The essential property of such trust-situations is that the human respectively the agent does *not* have sufficient

information that can be applied directly to assess a trust value. Instead, trust is inferred from related contextual factors. We think that this lack of a certain basis for decision-making is *the* defining characteristic of trust and is not taken into account in most current trust models.

Consequently, situations where trust can *exclusively* be based on the reputation calculated from recommendations or trust-networks does not comply with this strict specification of trust. The same holds for cognitive and game theoretic models of trust based on computable incentives of the trustee or statistical models dependent on repetitive interactions in a restricted context-*independent* environment.

What all those approaches have in common is a well defined way how trust can be assessed and which factors need to be taken into account. Moreover, most existing statistical trust models do not perform well when there is no long history of interactions in a predefined and consistent environment. The same applies to cognitive and game theoretic models when the information needed for reasoning is not easily accessible.

Again, the absence of this information in our opinion *constitutes* a human-trust situation and hence conflicts with current computational-trust models. Moreover, current models are not able to transfer knowledge gained in a specific context to a related context. Humans, have proven to be especially skilled in perceiving traits like trustworthiness in initial trust situations.

Based on those observations our objective is to relax existing restrictions of computational trust by trying to learn trust in a rich context-dependent relational environment: First, we want to increase the degree of automatization of trust calculations by e. g. , learning the relevance of attributes. Second, we want to make trust context-dependent: Modeling the environment *from the perspective of the trustor*, two entities, both described by their respective attributes, constitute a trust situation: (i) the trustee and (ii) the state of the environment. Most importantly, both entities are interconnected by relational dependencies.

If the trustworthiness depends not only on the trustee but also on the state of the environment in which one needs to trust, the trustor can make more precise decisions and can apply learned knowledge to a wider range of situation. For instance, a seller might be trustworthy if offering a specific product, but not when offering another product. Furthermore, in such a situation a relation like the price might help to assess trustworthiness while depending on a particular product and the seller at the same time. By taking all this into account, we can improve predictions, give more meaning to trust and at the same time - by generalizing from different contexts - increase learning efficiency.

5.1.2 Related Approaches to Learning Computational Trust

A predominant fraction of existing trust models assigns a single trust value to each agent as a whole. They do not take into account that trust is a result of past behavior which is a sequence of (re)actions. The (re)action of an agent in a trust situation determines the assessment of trust in future actions of this agent (cmp. [Moranz, 2004]). It does not directly allow to assign a trust value to the agent as a whole.

In addition, most existing models do not allow trust decisions to be situated in a specific context. This applies to implemented trust systems like (see [Zanero,

2005, Kinateder et al., 2005, Fullam et al., 2005]) and to extended trust models.

Examples for such extensions are e. g. , risk (see [Ruohomaa and Kutvonen, 2005]), dis- mis- un-trust (see [Marsh and Dibben, 2005]), context (see [Jøsang et al., 2005]), forgiveness (see [Vasalou and Pitt, 2005]), recommendations (see [Chadwick, 2005]) and affects (see [Hassell, 2005])). However, we aim at including a wide range of contextual relations to be considered during trust assessment.

Given a trust model trust values are in general estimated by counting the success of previous interactions concerning a *trust-metric*. This trust-metric is improved over time (see [Witkowski et al., 2001]). The target function of the agents are the optimization of the outcome of interactions and the cooperation between agents (see [Wu and Sun, 2001]). The evaluation is performed using various different approaches, including e. g. , fuzzy set theory (see [Rehak et al., 2005]) and Bayesian estimation (see [Shi et al., 2005]).

As already pointed out, connecting trust to the trusted agent alone without considering contextual and other aspects (dimensions) of trust is not sufficient in many scenarios. Whereas some research on trust concedes the importance of context information, most of them do not actually use such information for the calculation of trust degrees Especially, most machine learning techniques used for trust learning have not been focused on this issue or do not offer a solution in a general and automated way (see [Teacy, 2006]). So far, only one approach also models context by taking into account identity and state (see [Rehak and Pechoucek, 2007]). Besides that, using contextual information for initial trust assessment and the transfer of trust between contexts is novel to our knowledge.

Regarding its dimensionality, most work represents trust as a single discrete or continuous variable associated with one specific agent. Modeling trust in multiple dimensions is only considered by a few elaborate approaches such as [Maximilien and Singh, 2005]. Our approach leaves it to the actual scenario how trust needs to be modeled in this respect. In principle, IHSTM can handle an arbitrary number of trust variables, each associated with one aspect of the trustor's expectations and represented with any probability distribution needed.

Analogously, we argue that a fine grained modeling of relations between agents and their environment is essential to capture the essence of trust, especially in initial trust situations. There exist a few approaches that can take relationships into account when modeling trust. But in most of this research such relationships are either only considered as reputation or recommendations (see [Sabater and Sierra, 2001]), or as interactions between a group of agents (e. g. , [Ashri et al., 2005]). The diverse kinds of relations that exist between two agents in a specific situational context are not modeled in detail. In addition, most learning techniques are optimized for one specific scenario only and do not make use of well funded techniques from probability theory.

Assessing initial trust values for unknown agents based on pre-specified membership to a certain group has been addressed by [Sun et al., 2005]. Here, a group-based reputation architecture is proposed where new agents are assessed according to their pre-specified membership to a certain group of agents. Likewise, the *TRAVOS-C* system proposed by [Teacy, 2006] includes rudimentary ideas from hierarchical Bayes modeling by assigning parameter distributions to groups of agents but does not come to the point to give a fully automated and intuitive way of how to learn infinite hidden variables.

5.2 Learning Trust in Uncertain Ontologies using IHSM

As mentioned before, real world ontologies commonly contain data from different sources or consist of several ontologies that are merged into one. Integration of the potentially conflicting information is a difficult task that research areas like ontology matching, integration and alignment are focusing on (see [Staab and Studer, 2009] for an overview). Even if the mapping or merging of ontologies is successful and the resulting ontology is consistent and satisfiable it still might contain incorrect facts that cannot be detected by deductive reasoning. We call such a knowledge base that contains potentially untrustworthy facts from different sources an *uncertain ontology*.

In this situation CT is one possible approach to cope with the uncertainty resulting from potentially untrustworthy sources. Consider the social network example from the previous chapters (see e. g. , Sec. 4.1) where a user has specified an incorrect age for the person registered in the social network. If no prior knowledge is given that might cause conflicts with incorrect facts, those facts are hard to detect as incorrect. Even if there is an axiom stating that for example a specified age of 5 has to be incorrect (because the person is allegedly attending a school) does not help to assess the correctness of statements not affected by such axioms. For instance, the list of schools the person has allegedly been attending might not be easily verifiable.

However, it might be possible that this person specified information that conflicts with information by other more trustworthy persons in the social network. This could for instance be the location of the school he supposedly attended. Now, it might be possible to assess the trustworthiness of statements by this person and use those to predict the trustworthiness of other statements. In this section, we demonstrate how such situations can be modeled in a relational way using IHSM. The specialized version of IHSM for Trust Learning will be called *Infinite Hidden Semantic Trust Model* (IHSTM).

5.2.1 The Infinite Hidden Semantic Trust Model

This section covers the general modeling of CT using a relational model first, before giving a concrete example (Sec. 5.2.1). The example describes an online auction scenario like eBay[6] and shows how it is modeled using IHSTM.

Relational Trust Modeling

The basic precondition for the emergence of trust are entities and social interactions between those entities. Hence, we chose a scenario that is interaction-centered as seen from the perspective of one agent who needs to trust (*trustor*) in someone/something (*trustee*). In the case of uncertain ontologies the trustor tries to assess the trustworthiness of a statement provided by the trustee.

As usual in trust scenarios, (dis-)trust is related to the expected occurrence of some promised outcome (e. g. , the provision of correct and precise information as negotiated before, or the delivery of the expected product at an agreed price). Thus, the basic interaction-trust scenario then consists of:

[6] http://www.ebay.com/

1. A set of *agents* A (trustees) that are willing to interact with the trustor, each characterized by a set of observable attributes Att^A. An agent can be considered as a person or more general as any instance that can be trusted, like an information source, a company, a brand, or an authority.

2. A set of *external conditions* or *state* C with corresponding attributes Att^C. An apparent condition would be the type of service provided by the trustee, for instance a specific merchandize or an information supply in case of information providing agents. Moreover, this implies all external facts comprising this particular state like the trustor's own resources or the current market value of the merchandize in question.

3. A relation $interacts(a, c)$ with a set of relationship attributes Att^O capturing all negotiable interaction issues depending on a specific agent $a \in A$ and specific conditions $c \in C$. In general those attributes can be directly manipulated by the interacting agents and separated into two different sets:

   (a) *Promised outcome* O^p: Attributes Att^{O^p} of this set are (in general) observable before the trust-act is carried out.

   A typical attribute of this category is for example the price for the merchandize or the scope of the services offered, such as the amount and precision of information in case of a negotiation among agents regarding the delivery of information. A promised outcome $o^p \in O^p$ is an assignment of values to the corresponding attribute vector Att^{O^p}, which can be negotiated by the trustor and trustee. In game theory this kind of non-binding negotiations among agents before the actual interaction takes place is known as "cheap talk" (see [Murray and Gordon, 2007]).

   (b) *Effective outcome* O^e: The set of attributes Att^{O^e} are not observable until the trust-act has been carried out. Those attributes act as a feedback or judgment for the trustee in respect to his expectations. Att^{O^e} can be thought of as quality aspects of the merchandize, like the correctness or usefulness of the information provided or the delivery time. From a decision theoretic point of view those attributes are the objectives or interests of the trustor and need to be optimized in a multi-criteria optimization problem. From a MultiAgent Learning (MAL) perspective Att^{O^e} depends on the actions Ac carried out by the opponent.

   This way of modeling interaction-trust scenarios allows us to capture almost any context relevant for trust-based decision making.

Our goal is to learn the value function $o^p \rightarrow o^e$ that allows to predict o^e from a given o^p offered by agent a under external conditions c. Moreover, it might be possible to calculate the utility of the trustor for a given o^e. Hence, the ultimate objective is to find the utility function $o^p \rightarrow [0, 1]$. If this function is known the trustor knows what assignment to O^p he should try to achieve (e. g. , in a negotiation) to maximize its payoff / reward.

In the case of uncertain ontologies the trustee is the information provider, and the state is the actual information provided by the trustee. Att^a are known

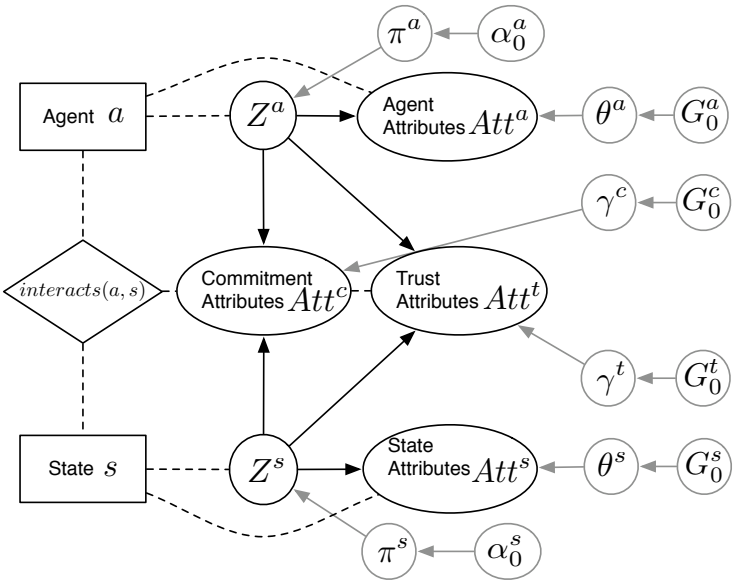

Figure 5.2: IHSTM graphical representation as a DAPER model

properties of the trustee and Att^s are known properties of the information like the date the information was given. The relation $interacts(a, s)$ has in the simplest case the meaning $provides\_information(a, s)$. We now can asses trust values to certain informations provided or used by certain sources. This additional information can be used to rank the results or use them in case of conflicting information that needs to be merged.

Trust Modeling for Online Auctions using IHSTM

Relational models are an obvious formalization of requirements arising from the relational dependencies of entities in the trust scenario just described. The Infinite Hidden Semantic Trust Model (IHSTM) proposed in this thesis is a relational model specialized in this trust scenario. We will illustrate the application of the IHSTM to a specific online auction scenario using a DAPER model.

Figure 5.2 illustrates the IHSTM as a *DAPER model* (cmp. [Heckerman et al., 2004]). Entity classes are depicted as rectangles and the relationship class as a rhombus. Observable evidence *Att* is modeled in attribute classes of entities and relationships (ovals). As in a classical non-relational Bayesian network, direct statistical dependencies are modeled as directed arcs. The DAPER model should be thought of as a template which, based on the actual objects in the domain, is expanded into the ground Bayesian network.

To illustrate the abstract model we will use the *eBay feedback-system* as a concrete example throughout this chapter. Being the most popular online auction and shopping website, fraud on eBay is a serious and well-known problem. An attempt to deal with fraud is the eBay feedback-system where users leave feedback about their past transactions with other eBay-users.

Suppose the trustor agent is a buyer who wants to build a context-sensitive relational trust model to analyze the trust situation on eBay in general and

assess trust values for purchases from eBay in particular. In this scenario, the trustor itself does not need to be modeled explicitly because he learns a *personalized* model based on its own viewpoint and experience. The trustee a however represents sellers on eBay and the state s represents items that are for sale. The relation $interacts(a, s)$ would best be specified as $offers(a, s)$ in this context.

The attributes Att specify the observable features of the trust situation. Att^a describes properties of the seller like the feedback score, the percentage of positive feedback and his length of membership. Att^s specifies features that are associated with the product, for instance its category and its condition (new or used). The price however is represented as a relational attribute Att^c because a different seller could offer the same product for a different price. Thus, Att^c stands for all *commitments* seller and buyer make in the negotiation process. Besides the price or winning bid this can e. g. , be shipping costs, bidding history, extent of warranty, payment details and shipping rates. Finally, Att^t can include all dimensions of trust that are important for the trustor when he finally gives feedback about his purchases. Relevant dimensions might be: actual shipping time, whether the item proved to be as described, if the communication with the seller was as expected and so on.

As an example, one could now express the trustworthiness of an offer concerning product quality Att^t, given the seller a offers item s for price Att^c. Note that more than one attribute per entity or relation can be considered as well.

5.2.2 Technical details

To complete the technical details of our specific relational trust model we now introduce the remaining elements of the IHSTM. Please recall the more general IHRM/IHSM described in Sec. 3.3.2/4.4. Following the ideas of [Xu et al., 2006] and [Kemp et al., 2006] we assign to each entity a hidden variable, denoted as Z^a and Z^s and depicted as circles in figure 5.2. Related to the hidden states in hidden Markov models, they can be thought of as unknown attributes of the entities and are the parents of both the entity attributes and the relationship attributes. The underlying assumption is that if the hidden variables are known, both entity attributes and relationship attributes can be well predicted. A very important result of introducing the hidden variables is that now information can propagate in the ground network, which here consists of attribute variables exchanging information via a network of hidden variables.

Given that the hidden variables Z have discrete probability distributions they intuitively can be interpreted as cluster variables where similar entities (similar sellers or similar items) are grouped together. The cluster assignments (or hidden states) of the entities are decided not only by their attributes, but also by their relations. If both the associated seller and item have strong known attributes Att^a and Att^s, those will determine the cluster assignments and the prediction for Att^t. In terms of a recommender-system terminology we would obtain a content-based recommendation system. Conversely, if the known attributes Att^a are weak, then the cluster assignments Z^a for the seller a might be determined by the relations to items s and cluster assignments of those items cluster assignments Z^s. Accordingly, this applies to items s and its cluster assignment Z^s. In terms of a recommender-system terminology we would obtain a collaborative-filtering system. Consequently, IHSTM provides an elegant way

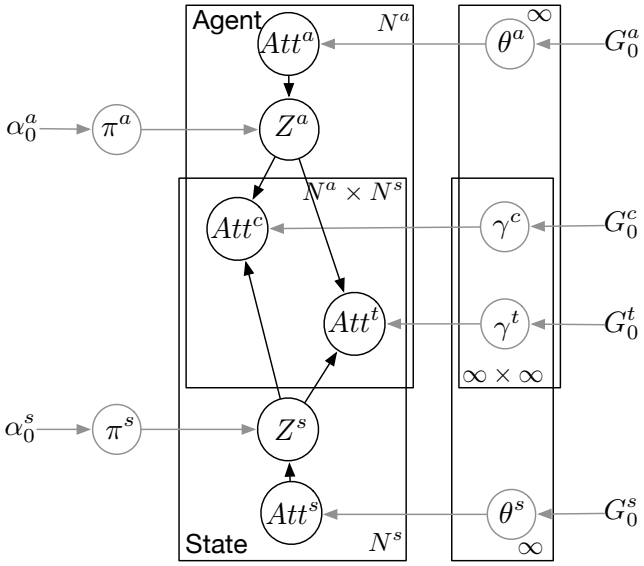

Figure 5.3: IHSTM graphical representation as a plate model

to combine content-based predictions with collaborative-filtering prediction.

In the IHSTM, Z has an infinite number of states. Mixture models with an infinite number of states are Dirichlet process (DP) mixture models, which have the property that the number of actually occupied components is determined automatically in the inference process (cmp. Sec. 4.4 and Sec. 3.3.2). The fundamental idea is, that depending on the complexity of the problem, the model can 'decide' for itself on the optimal number of states for the hidden variables; thus a time consuming optimization of the number of clusters can be avoided.

After sketching the functioning of the infinite hidden variables, we can complete the model by describing the local distribution classes denoting the parameters and hyperparameters of the probability distributions. They are shown as small gray circles in the DAPER model (see Fig. 5.2). As an alternative to the DAPER model, we display the structure of the IHSTM as a *plate model* (see Fig. 5.3), another commonly used graphical representation for statistical relational models (cmp. Sec. 4.3).

Now we consider the variables for the seller entity. For each specific seller i there is a hidden variable Z_i^a with the flexible and potentially infinite number of states K^a. The clustering $Z_i^a = k$ specifies the assignment of seller i to the specific cluster k. The weights $\pi^a = (\pi_1^a, ..., \pi_{K^a}^a)$ are multinomial parameters with $P(Z^a = k) = \pi^a$ and are drawn from a conjugated Dirichlet prior, $\pi^a \propto Dir(\cdot|\alpha_0^a, \alpha^a)$. $\alpha^a = (\alpha_1^a, ..., \alpha_{K^a}^a)$. α_k^a represents our prior expectation about the probability of a seller being in cluster k. $\alpha_0^a > 0$ determines the tendency of the model to either use a large number (large α_0^a) or a small number of clusters in Z (small α_0^a). For additional theoretical background on Dirichlet process mixture models, consult Sec. 4.3.2 or for example [Tresp, 2006].

Since we only consider discrete attributes in our eBay example, a particular attribute Att^a is a sample from a multinomial distribution with multinomial parameters $\theta^a = (\theta_1^a, ..., \theta_{K^a}^a)$. The base distributions G_0^a and G_0^s are the as-

sociated conjugate priors. So, $\theta^a \propto G_0^a$. The same applies to the multinomial parameter γ for each of the $K^a \times K^s$ configurations related to each relational attribute Att^c and Att^t. Again, a Dirichlet-process prior is employed, so that $\gamma^c \propto G_0^c$.

Now we briefly describe the generative models for the IHSTM. The method we use to generate samples from a Dirichlet Process mixture model is the *Chinese Restaurant Process* (CRP, see [Tresp, 2006]). See Sec. 4.4.2 for the Truncated Stick Breaking model used for the IHSM and Sec. 4.6.3 for a comparison and further references.

The clustering of data points in a CRP can be explained by the following analogy: Imagine a restaurant with an infinite number of tables. Now customers enter the restaurant one by one and choose a table to sit down. Each customer either chooses to sit down at an unoccupied table or joins other customers at an already occupied table, where the table selection probability is proportional to the number of persons already sitting at a table. Applying this scenario to the Dirichlet process, the tables are clusters and the customers are data-points. After N data-points are sampled the $N + 1^{th}$ sample is generated as follows:

- The $N + 1^{th}$ agent is assigned to an existing agent cluster i with probability $\frac{N_i}{N+\alpha_0}$ and inherits parameters θ_i and γ.
- With probability $\frac{\alpha_0}{N+\alpha_0}$ the agent is assigned to a new cluster $K + 1$. For the new user cluster, new parameters θ_i and γ are generated as described above.

The procedure is repeatedly applied to all hidden variables in the ground network.

5.2.3 Inference

Based on the generative model presented in the previous section we can now generate samples from the IHSTM. In particular, we are interested in how to generate samples from the unknown states and parameters, given observed data. The most important goal is to infer the conditional distribution of the hidden variables Z^a, Z^s given all known attributes entity attributes Att^a and Att^s as well as relationship attributes Att^c and Att^t. This eventually allows us to make predictions about unknown attributes, like target value Att^t.

A way to approximate this posterior distribution of the hidden variables is by means of Gibbs sampling (GS), an MCMC-method. In our model, it is possible to formulate a GS in which only samples from the hidden variables are generated by integrating out model parameters (see [Xu et al., 2006]). The Markov chain is thus defined only for the hidden variables of all entities in the given domain. The GS iteratively samples the hidden variable Z^a, conditioned on the other hidden variables Z^s until the procedure converges. See Sec. 4.4.2 for the blocked Gibbs Sampling used for the IHSM and Sec. 4.6.3 for a comparison and further references.

In particular, Z is updated as:

1. For Z^a: Pick a random agent i. Assume that for N_k^a agents, $Z^a = k$ without counting user i.

Either assign agent i to cluster k with probability proportional to

$$P(Z_i^a = k | Z_{j\neq i}^a, Att_i^a, \theta^a, \gamma^c, \gamma^t, Z^s) \propto \\ kP(Att_i^a | \theta_k^a, \gamma_{k,*}^c, \gamma_{k,*}^t)$$

where N_k is the number of agents already assigned to cluster k and $\gamma_{k,*}$ notes the relation parameters of agent cluster k and all state clusters.

Or generate a new cluster $K+1$ with probability proportional to

$$P(Z_i^a = K^a + 1 | Z_{j\neq i}^a, Att_i^a, \theta^a, \gamma^c, \gamma^t, Z^s) \propto \\ \alpha_0^a P(Att_i^a | \theta_k^a, \gamma_{k,*}^c, \gamma_{k,*}^t)$$

2. For Z^a: Pick a random state j and update its cluster assignment Z^s, accordingly.

3. If during sampling a state becomes unoccupied, remove that state from the model and reassign indices.

After a burn in period the Markov chain has converged, and standard statistical parameter estimation techniques can be used for estimating the parameters γ_{k^a,k^s}^t of Att^t from given cluster assignments. We extended the algorithm, as just described, to enable the handling of more than one relationship attribute. Being able to use an arbitrary number of relationships is essential to enable a rich representation of the interaction context as well as multidimensional trust values.

5.3 Experimental Setup

To investigate the performance of the IHSTM we employ synthetic data, real world data from the eBay example used for illustration in the previous section and simulated negotiation data. The synthetic data is mainly used to evaluate the cluster analysis capabilities, the eBay data for predictive performance and the negotiation data for learning efficiency. Before the empirical results of our experiments will be presented (see Sec. 5.4), we first describe the experimental setup.

5.3.1 Synthetic Data

To explore the learning and modeling capabilities of our IHSTM we generated synthetic data and evaluated its ability to find clusters in this data. For this purpose we constructed an interaction-trust scenario with the fixed number of 2 entity attributes per entity and 2 relationship attributes, one for O^p and one for O^e. The number of entities $|A|$ and $|C|$ was pre-specified but varied in different runs, as well as the underlying cluster size r^a and r^c for Z^a and Z^c. Each entity was randomly assigned to a cluster and its attributes were sampled from a multinomial distribution with 4 possible outcomes and parameter vector θ each. θ in turn, was once randomly generated for each cluster. Accordingly, $r^a \times r^c$ Bernoulli-parameters γ for relationship attribute att^{O^p} and att^{O^e} were constructed.

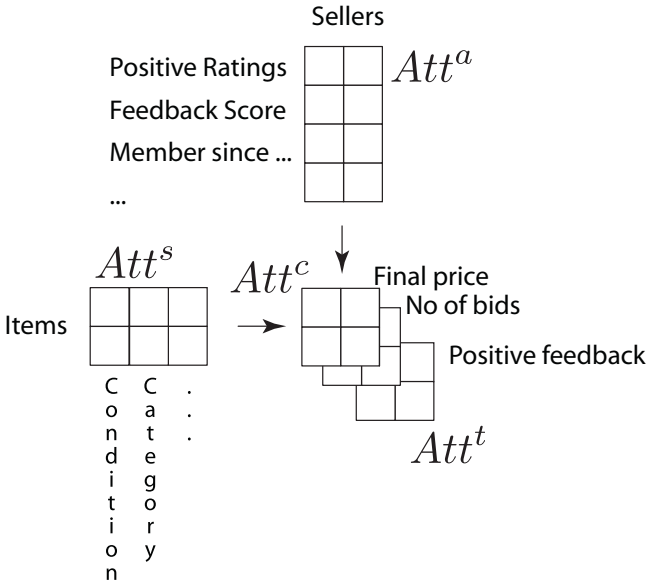

Figure 5.4: Experimental setup for the eBay scenario

5.3.2 eBay-User Ratings

eBay feedback-profiles are a valuable source of easily accessible data that expresses human-trust assessment. Every eBay member has a public feedback profile where all items he has bought or sold in the last 90 days are listed with the associated feedback ratings he received. In addition the feedback profile includes statistics on all transactions of the user.

We gathered data from 47 sellers that on the one hand had at least 10 negative or neutral ratings and on the other hand sold items in at least one of 4 selected categories from the lowest level within the eBay-taxonomy. The former is important because negative or neutral user-ratings on eBay are rather rare. To further balance the ratio of positive vs. negative/neutral ratings we only evaluated as many positive rated transactions as there were negative/neutral ones. This way, the data-set is stratified, meaning that there is an equal number of positive and negative ratings per seller.

Attributes Att^a of the seller were directly extracted from the feedback profile. We picked the positive feedback and the feedback score and discretized both in 2 and 5 classes, respectively. For the item attributes Att^s we chose the top level category in the eBay taxonomy on the one hand, resulting in 47 discrete states. On the other hand, we collected the item condition which is a binary feature: either new or used.

From those 47 hand-picked sellers we gathered a total of 1818 rated sales of 630 different items. Two items were assumed to be alike if they were in the same lowest level category and their attributes were identical. Relation attributes are always of size $seller \times items$, so Att^c and Att^t both are sparse matrices with 47×630 possible entries. The non-zero entries indicate that this seller has sold this item.

As we wanted to keep the computational complexity low we only considered

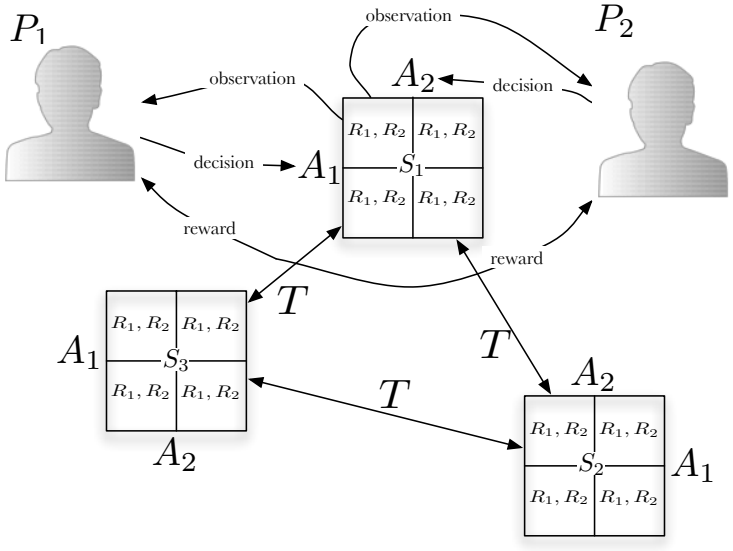

Figure 5.5: Setup for general-sum stochastic games.

binary relational attributes Att^c resp. Att^t. For Att^c we chose the binarized final price of the auction and for Att^t the rating. Negative and neutral ratings were both treated as negatives.

5.3.3 Negotiation Game

Finding an agreement amongst a group of conflicting interests is one of the core issues of distributed artificial intelligence. Auctions, information markets, preference and judgement aggregation, game theory and automated negotiations are all research areas that deal with those kind of problems. However, most of the approaches neglect the fact that finding the best agreeable solution is not sufficient if commitments can not be enforced by the interaction mechanism or if the incentives of the opponents cannot be inferred. In order to investigate this issue we extended the implementation of a multiagent trading framework by an additional negotiation step.

General Framework

Our scenario can be based on one of the most general frameworks for learning interactions in multiagent systems (a.k.a. *MultiAgent Learning* (MAL)) namely *general-sum stochastic games* (see [Hu and Wellman, 1998]). A stochastic game can be represented as a tuple (A, C, Ac, R, T)[7]. A is the set of agents, C is the set of *stage games* (sometimes denoted as *states*), Ac is the set of actions available to each agent, R is the immediate reward function and T is a stochastic transition function, specifying the probability of the next stage game to be played. See Fig. 5.5 for an illustration.

[7]Our notation differs slightly from the commonly used ones, where A denotes actions and S states. Our notation should become clear in the next section

It is in the nature of trust that we are dealing with incomplete and partially observable information. We neither assume the knowledge of the reward function R of the opponent nor their current state C. In fully observable games with perfect monitoring, incentives to betray can be estimated and trust becomes irrelevant because agents can be punished effectively [Murray and Gordon, 2007]. Furthermore trust decisions require general sum games where joined gains can be exploited. Both zero-sum (e. g. , [Littman, 1994]) and common-payoff (e. g. , [Wang and Sandholm, 2003]) games are not relevant because either there are no joint gains or the agents' interests do not conflict.

Building on that formal setting, our goal is to predict trust values O^e associated with the expectation of the next actions Ac given agent A and state C. We neither are trying to learn a strategy or policy nor are we interested in finding equilibria or proofing convergence. But we make contributions on how to scale MAL (cmp. Sec. 5.3.3) to more complex scenarios and show how an opponent model can be learned efficiently:

Predicting the next action of an opponent is an essential part of any model-based approaches to MAL (see [Shoham et al., 2006]). The best-known instance of a model-based approach is fictitious play [Brown, 1951] where the opponent is assumed to be playing a stationary strategy. The opponent's past actions are observed, a mixed strategy is calculated according to the frequency of each action and then the best response is played, accordingly. This technique does not scale well to a large state-space $|C|$ as we experienced in our experiments: The same stage game is on average not observed twice before 400 interactions are performed. Thus, this kind of naive approach does not allow to make an informed decision before the completion of 400 interactions and is obviously not suited for initial trust scenarios.

In our approach we make use of two techniques to face this issue. First, we allow to model any context related to the next trust-decision in a rich relational representation. This includes non-binding arrangements among agents also known as *cheap talk* (see Sec. 5.2.1) which take place before the actual interaction O^e is carried out and which are denoted as O^p. Second, we make use of techniques from the mature field of *transfer learning* (see [Caruana, 1997]) to reuse knowledge from previous interactions for potentially unknown future actions.

Specific Scenario

In the chosen scenario, players try to collect a certain number of resources for selling. Hereby, they need to trade resources but do not have to stick to their agreements.

In the following, we describe the testbed used for the subsequent evaluation and comparison task. Our testbed is designed in a way that makes the negotiation scenario complex enough to draw meaningful conclusions while keeping the negotiation processes comprehensible and analyzable.

The scenario the agents are situated in is a production game. All players receive different kinds of resources. Each player tries to collect a certain number of resources of one type at a time to assemble products. By selling their products agents earn game points. The functionality of a player's resource store is equivalent to that of a FIFO (First In, First Out) queue. Hence, elements are added to one end of the queue (the *tail*), and taken off from the other (the

head). The production unit however resembles a *stack* based on the LIFO (Last In, First Out) principle. Elements are added and removed only on one end. Thus, the game is called *Queue-Stack-Game*. One additional behaviour applies to the production units of this game. They can hold only one type of resources at a time and loose their previous content if new elements of a non-matching resource type are added.

Each round, every player is assigned a sequence of new resources, which are uniformly drawn from the available resource types. These elements are added in sequence to the tail of the queue. Next, a number of resources is taken off the head of the queue and added to the stack. As a consequence, the previous content of the stack might be lost if any of the new resources is of a non-matching type. To avoid this waste, players can negotiate with their peers and offer to give away resources from their queues. In doing so, they might be able to create sequences of identically typed resources of a certain length and thereby succeed in the game.

The following section describes the rules and phases of the Queue-Stack-Game in detail.

Production: There are a number of game parameters and restrictions that apply to the production process of the Queue-Stack-Game, which are listed here:

- Each agent can produce only one product at a time.
- A product consists of a number of identically typed resources, this number being a game parameter, namely *stackCapacity*.
- The types of resources and the order in which they are allocated to the producers are random. The number of resources each player receives per round is fixed though and is a parameter of the game, namely *getPerRound*.
- The incoming sequence of resources cannot be altered by the agent before being added to the queue
- Each player is forced to input *pushPerRound* resources from the head of his queue into the production unit in each round.
- If the type of any newly input resource does not match the type of the product being currently assembled, this product is spoiled and thrown away.
- The players are admitted to remove elements of any types from their queue in order to give them to one of their fellow players.
- If a player receives resources, he is allowed to arrange them in the desired order before they are immediately fed into the production unit.

Allocation: Each round is divided into two phases, namely *allocation* and *negotiation*. In the *allocation* phase, *getPerRound* new random resources are enqueued in all players' resource stores. The resources allocated to the different players are independently generated. Subsequently, each agent is forced to remove the *pushPerRound*-first elements from the head of his queue and to push them onto the stack, maintaining their ordering.

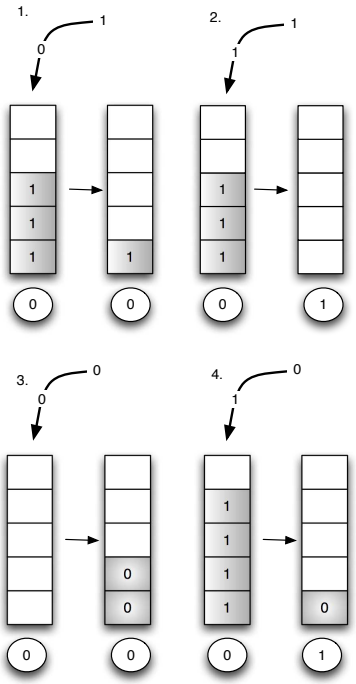

Figure 5.6: Queue-Stack-Game: Examples illustrating the behavior of a player's stack when additional resources are pushed.

If the production unit already contains some elements and their type does not match the newly pushed resources, the old contents of the stack are wasted. Fig. 5.6 illustrates four examples of feeding resources into the stack.

The examples show the state of the stack before and after new resources have been pushed. We assume two different types of resources, 0 and 1. The number of game points owned in the current situation is shown underneath each stack. In situation (1) all elements of the stack are discarded when the 0 token is pushed, as the types do not match. The 0 token itself is also thrown away, when the next resource, a 1 token is pushed. In situation (2), the player has more luck. The two resources pushed complete the product, which the player can sell and thus is rewarded. The production unit is empty now, ready to accept new resources of any type. Situation (3) shows how resources are added to an empty stack. In example (4) the first of the pushed resources completes the stack, the player sells the completed product, earns a reward and the stack is emptied before the next resource is pushed.

Generating Possible Worlds We will now formalize the notion of a state in the Queue-Stack-Game and outline the process of generating a set of possible worlds with respect to a particular state. A state s_c contains the following elements:

- the condition of the queue after resources have been removed, referred to as $queue(s_c)$

- the condition of the stack after transfer received from another player has been pushed, referred to as $stack(s_c)$
- the number of rewards, $rewards(s_c)$
- the set of resources received from another player, $get(s_c)$
- the set of resources removed from the queue in order to be transferred to the other player, $give(s_c)$
- $noWaste(s_c)$, a flag indicating whether elements of the stack were wasted when $get(s_c)$ was pushed
- $earnedReward(s_c)$ a flag being set to 1 if a reward was earned when pushing $get(s_c)$ or 0 otherwise.

The queue of a state s_c can be generated by removing each possible subset of resources from the previous queue $queue(s_{c-1})$. The removed resources are $give(s)$. Which resources can be received from other players is not known to the agent, as he has no insight into his opponent's resource situation. So all possible combinations of resource types up to an arbitrary total amount are considered. As the resources can be pushed in any order, $get(s_c)$ is generated for each permutation of the received transfer. $stack(s_c)$ is the resulting stack, after $get(s_c)$ has been pushed. $rewards(s_c)$ is the number of rewards the agent possesses afterwards. $noWaste(s_c)$ and $earnedReward(s_c)$ are needed when calculating the utility for a state.

The deal that produced a state is implicit to the state. When we speak of the utility of a deal, we mean the utility of the state which results from execution of the deal.

Next, we will describe the course of one round of the Queue-Stack-Game. Each round is divided into two phases, namely *allocation* and *negotiation*. In the *allocation* phase, *getPerRound* new random resources are enqueued in all players' resource stores. The resources allocated to the different players are independently generated. Subsequently, each agent is forced to remove the *pushPerRound*-first elements from the head of his queue and to push them onto the stack, maintaining their ordering. For details on the allocation phase please see Sec. 5.3.3.

Having completed the *allocation* phase, the players enter the *negotiation* phase. The outcome of a successful negotiation is a *deal*, describing which sets of resources are to be exchanged between players. Hence, the agents engage in *practical reasoning*. The exchange of resources is the only means for agents to take action during the game. If a player chooses not to negotiate or not to agree to any deal proposed to him, his succeeding in the game entirely depends on the random resource sequence he is allocated. If players cannot find an agreement, the *default deal* is forced. The default deal entails no actions of the players, thus the resource situation of all players remains unchanged. The available locutions are *propose*, *reject*, *accept* and *inform*. The *negotiation protocol*, i.e. the communication rules are defined as follows:

1. The negotiation terminates immediately after an acceptance message is uttered by one of the participants.

2. The negotiation terminates with the default deal if a player quits the negotiation.

3. The players take turns in proposing deals. If a player cannot propose a new deal, he is forced either to accept a previously offered deal or to quit the negotiation.

4. All deals offered during the negotiation can be accepted at any point in time later on as long as they have not been rejected.

5. A counterproposal can be preceded by a critique and a rejection.

This protocol entails that agents have to receive up to three messages (*inform*, *reject*, *propose*) until they are allowed to respond.

After the outcome of the negotiation is set, the deal is executed. The resources each player receives from fellow players are pushed onto the stack, whereby the player himself can dictate the order in which they are to be pushed. Eventually, the players are rewarded if they were able to complete their stack and thus sold a product.

As defined before, let c be the commitments that the agents are negotiating over. The outcome of this negotiation is specified by a set of binary features Att^c. Now, given a set of commitments c that two agents have agreed on and promised to fulfill, the agents enter an additional trading step in which each of them is free to decide which action to take. This way, the agent can decide whether to stick to a commitment or break it at will.

5.4 Experimental Results

In the following sections, three different aspects of the IHSTM's performance are reported: First, the algorithm's abilities to characterize a trust-situation by clustering are investigated in Sec. 5.4.1. Second, the predictive performance concerning trust values is tested (see Sec. 5.4.2). Finally, the learning efficiency is analyzed in the context of dynamic behavior of non-stationary trustees. As the later cannot be analyzed within the eBay scenario we used interactions recorded from the negotiation game just described. The experimental setup and evaluation is covered in Sec. 5.4.3.

5.4.1 Cluster Analyis

An inherent problem in evaluating the results of a cluster analysis is that commonly a gold standard - the 'perfect clustering' - is not known. Thus, we use synthetic data first, where the original clustering is known, before evaluating real world data without a gold standard.

Synthetic Data

In Figure 5.7 and 5.8 two different error metrics measuring the performance of IHSTM averaged over 10 runs are shown. The top row graphs visualize the classification error metric (CE) for clusterings while the bottom row depicts the accuracy (AC) of classifying att^{O^e} correctly. Both are supplemented by a 95% confidence interval. CE reflects the correspondences between the estimated

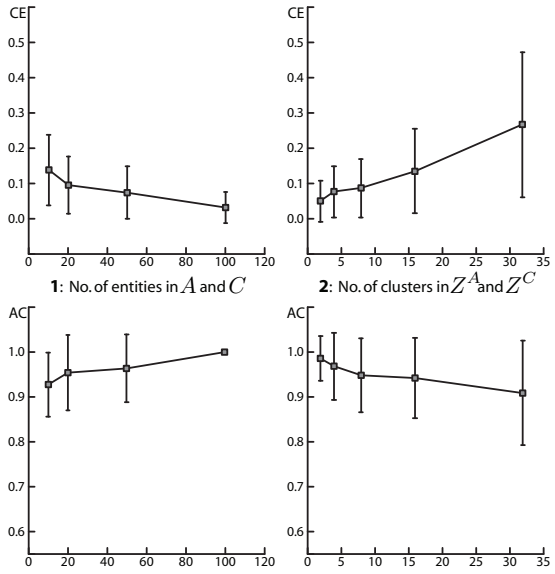

Figure 5.7: Results on experiment 1: Synthetic data, setup 1 and 2. Top row graphs show the classification error metric (CE), subjacent graphs show the related accuracy (AC).

cluster labels and the underlying cluster labels measuring the difference of both (see [Meilă, 2005]). A value of 0 relates to an exact match, 1 to maximum difference. In this experiment AC is a binary classification task and denotes the ratio of classifying att^{O^e} correctly. Results are averaged over both hidden variables Z^a and Z^c.

We considered three different experimental setups:

1. We analyzed the performance for different numbers of entities with fixed cluster sizes $r^a = r^c = 4$. The performance shown in Figure 5.7-1 expectedly suffers for small numbers of entities $|A| = |C| < 20$. Nonetheless, this result suggests that the IHSTM is quite robust even with few training samples. This makes it especially interesting for initial trust problems as discussed in the next section.

2. Correctly recovering different cluster sizes r^a and r^c while the number of entities was fixed to $|A| = |C| = 50$ was the goal of setup 2. In Figure 5.7-2 we see that the IHSTM underestimates the cluster sizes if $r^a = r^c > 16$. This suggests that the number of combinations in such a simple scenario is not enough and entities from different clusters tend to become alike. Still, the AC is almost perfect. Besides that, the number of entities per cluster ($|A|/r^a$ and $|C|/r^c$, respectively) gets so small that not all clusters are represented in the training set.

3. Finally, missing and noisy data sets were used in two different ways for training:

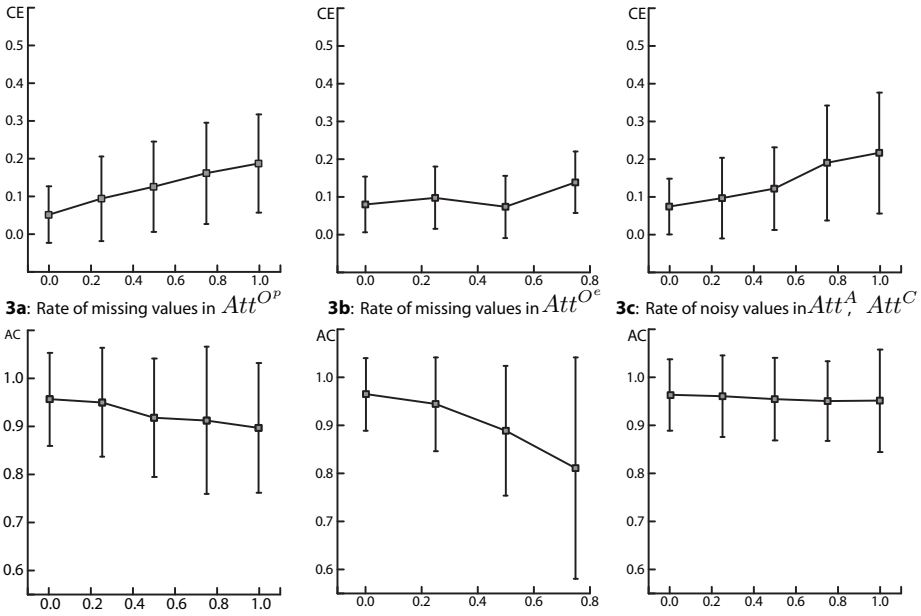

Figure 5.8: Results on experiment 1: Synthetic data, setup 3a-c. Top row graphs show the classification error metric (CE), subjacent graphs show the related accuracy (AC).

(a) Half of the relationship attribute O^e data was omitted while missing values for O^p were varied. The variance of all measures in figure 5.8-3a increases with the increase of missing values. Still, the AC is good although cluster correspondences deviate. This clearly shows that dependencies across relationship-attributes have a significant effect on the performance and can be exploited by IHSTM. As mentioned before, standard techniques working with a 'flat' vector-based attribute-value representation cannot use such information. In contrast IHSTM can propagate information through the network.

(b) First, evidence for O^e was partially omitted. The AC in Figure 5.8-3b expectedly drops because less training samples of the effective outcome that is to be predicted are available. Still, clustering abilities are hardly affected because other attributes can replace the missing information.

(c) Second, in order to measure the influence of the entity attributes we added noise to Att^A and Att^C. With the used parameter settings IHSTM did obviously (see Fig. 5.8-3c) not suffer in predicting AC. However the ability to infer the correct clusters was slightly hindered.

eBay Data

After having extracted the data, the GS-process to train the IHSTM is run. In the beginning, the sellers and items are re-clustered intensely and both cluster assignments and cluster sizes are unstable. Once the Markov chain starts to

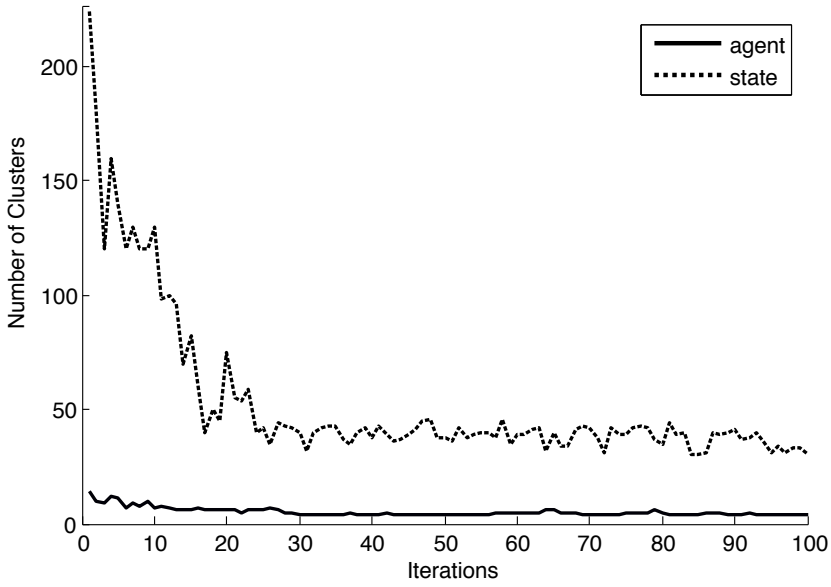

Figure 5.9: Trace of the number of agent- and state-clusters up to 100 iterations.

converge the cluster sizes tend to stabilize and eventually, the training can be stopped. The decrease of the cluster sizes is exemplarily shown in Fig. 5.9 for one cross-validation run.

After the clusters have stabilized we can visualize two interesting facts about the trust situation.

First, we can plot a matrix showing the assignments of each seller to a cluster. This can provide knowledge about how many different clusters exist, which are the most popular clusters and which elements are grouped together. After convergence, the 47 sellers were assigned to 4 clusters as shown on the left half of figure 5.13. The same assignment matrix can be generated for the items clsuter assignment but since there are 613 items and 40 item clusters, we did not plot the matrix and simply show its symbol Z^s on top of the right matrix in figure 5.13.

Second, the posterior probability $P(Att^c, Att^t|Z^a, Z^s)$ can be visualized. The matrix on the right side in figure 5.13 illustrates the probability of getting a positive rating given the cluster assignments of a seller and a item. A darker value indicates a higher probability of being trustworthy in a given interaction. Now, by picking a row (representing an agent cluster) or a column (representing a state cluster) we can identify clusters that are in general more trustworthy than others.

5.4.2 Predictive Performance

In order to judge the performance of predicting the trust value Att^t we compared the results of IHSTM with two other standard machine learning algorithms, namely a *Support Vector Machine* (*SVM*) using a PolyKernel and a *Decision Tree* (*DecTree*) both from the Weka toolbox [Witten and Frank, 2002]. Since those algorithms are both propositional learners, meaning they cannot handle a

	Accuracy	ROC Area
Ratio	48.5334 (±3.2407)	-
SVM	54.1689 (±3.5047)	0.512 (±0.0372)
DecTree	54.6804 (±5.3826)	0.539 (±0.0502)
SVM+ID	56.1998 (±3.5671)	0.5610 (±0.0362)
DecTree+ID	60.7901 (±4.9936)	0.6066 (±0.0473)
SVD	65.4728 (±6.0375)	0.6843 (±0.06421)
IHSTM	**71.4196** (±5.5063)	**0.7996** (±0.0526)

Table 5.1: Predictive performance on eBay user ratings

relational data representation but only a vector of independent and identically distributed features plus a label, we had to 'flatten' the data first (see Sec. 3.1). By transforming the data into a flat representation, also known as *propositionalization* (see Sec. 3.1.1), the structural information can be lost. There is no standard propositionalization procedure (see [Krogel, 2005]). The potential low quality of propositional features is not crucial in our simple scenario but becomes increasingly problematic in more complex relational models.

We propositionalized the data in three different ways: First, we only considered the target trust variable Att^t and tried to predict trustworthiness by the mere rate of positive feedback as it is done in most existing statistical trust models (see *Ratio* in table 5.1). Clearly, the result cannot be better than random guessing as the data-set is stratified. However, this demonstrates that the assumption of context independency made by many trust models is fatal when trust observations are uniformly distributed. Second, we tested the performance of the propositional algorithms with all features - namely Att^a, Att^s, Att^c and again Att^t - as the label. As a result we extracted 1818 samples with 5 features and one label, each. This way, the same features are available to the propositional learners as they are to the IHSTM. Third, we accounted for the missing relational information (which seller sold which product) by introducing two further features: An ID-number for the seller and the item, respectively. The final input to the propositional learning algorithms was a 1818×8 matrix in this setup.

In addition, we compared a standard single relational algorithm namely SVD (see Sec. 3.2.2 and Sec. 4.6.3) to test the performance when taking advantage of the collaborative effect. Note, that SVD can only use Att^t, Att^a and Att^s as inputs, because it is not a multi-relational algorithm.

The result of all setups is shown in table 5.1. We report the accuracy of predicting positive ratings as well as the AUC (Area Under the Curve; also called ROC area). This measure represents the area under the *receiver operating characteristic curve* which is used for evaluating binary classifiers that can output probabilities instead of binary decisions. In all our experiments, we averaged our results using 5-fold cross-validation. The accompanying 95%-confidence intervals are reported as well. Finally, the prediction performance is also evaluated for the IHSTM and compared to the previous attempts (see table 5.1).

In general, the task of predicting eBay-user ratings seems to be difficult, which can be explained when reading the comments assigned to the ratings. The reasons for a positive or a negative evaluation are most of the time not related to specific properties of sellers or items but a unique incident. Besides

that, the high incentives to give positive ratings despite having had negative experience are a general and well known flaw in the eBay-feedback mechanism: sellers usually wait for the buyer's rating before they rate the buyer. Thus, buyers often give positive rating just to receive a positive rating from the seller as well. As a response to this problem, eBay has introduced a new feedback mechanism in May 2008.

Still, the IHSTM's performance clearly outperforms random guessing and could verifiably outperform the propositional and single-relational learners. This is most likely due to the collaborative filtering effect, that can only be utilized by the IHSTM and partly by the SVD. Thus, there seems to be a gain if learning with the assumption that e. g. , when two sellers sell similar items they might be comparable in their trust-ratings. More precisely, if two sellers both got positive ratings after selling one specific item their ratings might be comparable when selling a different item as well. Or the other way round, if two items both got positive ratings after sold by one specific seller their ratings might be comparable when sold by a different seller as well. However, the performance of SVD shows, that modeling the trust-relation alone also gives inferior results.

5.4.3 Learning Efficiency

As mentioned in the introduction, the learning efficiency[8] and the ability to rapidly adapt is crucial, especially in so called initial-trust situations or in situations where the trustee does learn and adapt as well. To evaluate the performance concerning learning efficiency, we had to use a different, more controlled experimental setup as in the previous eBay example. Only if we know about the *stationarity* of agents we can compare the performance of an adapting agent to a stationary agent. For this purpose, we recorded interactions in a simulated strategic two-player negotiation scenario.

Evaluation

Three different agent types with two different negotiation strategies and three different trading strategies were used as opponents in the negotiation game.

The two negotiation strategies are both stationary and are based on a monotonic concession protocol (see [Endriss, 2006]). The agents denoted $Honest$ and $Fictitious$ only propose actions that they actually could perform, while agent $Greedy$ also offers and accepts infeasible actions with the intend to achieve an opponent action with higher payoffs. Both strategies iteratively propose a larger set of actions by lowering their expected utility and offering less favorable outcomes.

Each agent type plays a different trading strategy where $Honest$ and $Greedy$ are stationary and $Fictitious$ is adaptive. $Greedy$ always maximizes its utility regardless of c, while $Honest$-agent always sticks to c. At last, $Fictitious$ plays according to the *fictitious play* algorithm. It's a traditional learning algorithm from game theory for repeated games, where the opponent is assumed to be playing a stationary strategy. The opponent's past actions are observed, a mixed strategy is calculated according to the frequency of each action and then the best response is played, accordingly.

[8] By *learning efficiency* we do not mean computational complexity of the learning algorithm, but numbers of observations needed to make effective predictions.

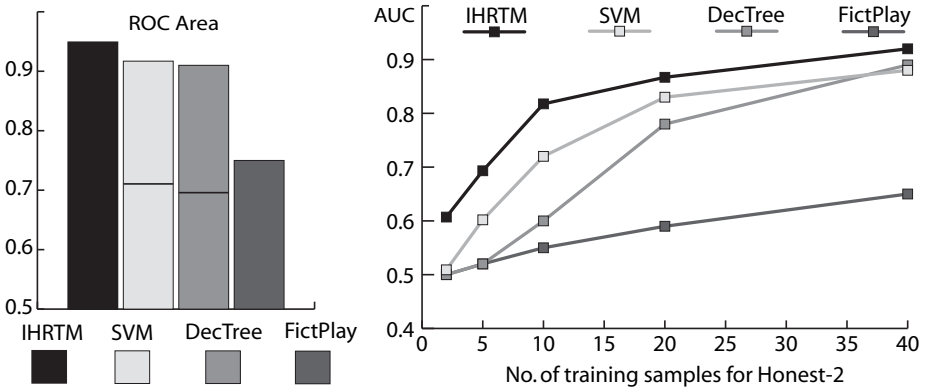

Figure 5.10: Results for play against *Honest*. Bar graph on the left: AUC for classifying Att^t. Graph on the right: learning curve for increasing number of training data for the additional *Honest-2*.

In every round that was played the commitment c and the effective outcome t were recorded and features Att^s, Att^c and Att^t were extracted. No specific attributes for Att^a were available except for the identity of the agent. Three discrete features Att^s from s where calculated describing the average payoff over all possible opponent actions, the maximum possible payoff and the number of feasible actions. Att^c describes a single binary feature stating whether there is a feasible action that could be carried out and would result in a positive reward if the negotiated commitment was carried out by the opponent. The same feature was recorded for Att^t after the actual action took place.

In this way a total of 600 interactions, 200 per agent type, containing a total of 289 different stage games were recorded. The input for the IHSTM consisted of three Att^s vectors with 289 elements, and two 289×3 matrices for Att^c and Att^t. Again, for a comparison with propositional machine learning algorithms the data was propositionalized, resulting in 600 feature vectors with $3 \times Att^s + 1 \times Att^c$ elements and 600 corresponding labels. As before, the content based algorithms were also evaluated with an agent- and state-ID as an additional feature. The evaluation procedure is the same as in the eBay experiments.

The overall performance according to AUC is depicted in the bar graph on the left of Figure 5.10. IHSTM shows a slightly better performance in classifying Att^t than SVM and DecTree. Without the agent-ID as an additional feature the performance of DecTree and SVM drops considerably (black line at around 0.7). Again, we explain the superior performance by IHSTM's ability to exploite cross-entity dependencies. *Fictitious*, as expected, performs much worse as it is not able to generalize over different interactions and canot make use of the context provided by Att^s and Att^c.

The inherent clustering ability of IHSTM suggests that it is especially well suited for rapid adaptation when unknown but related agents and conditions are observed. Actually, entities can be correctly assigned to a cluster without having seen a single effective Att^t related to this entity just by the other attributes. To check this assumption we gathered data from interactions with another *Honest* type agent and evaluated the performance for different numbers of training

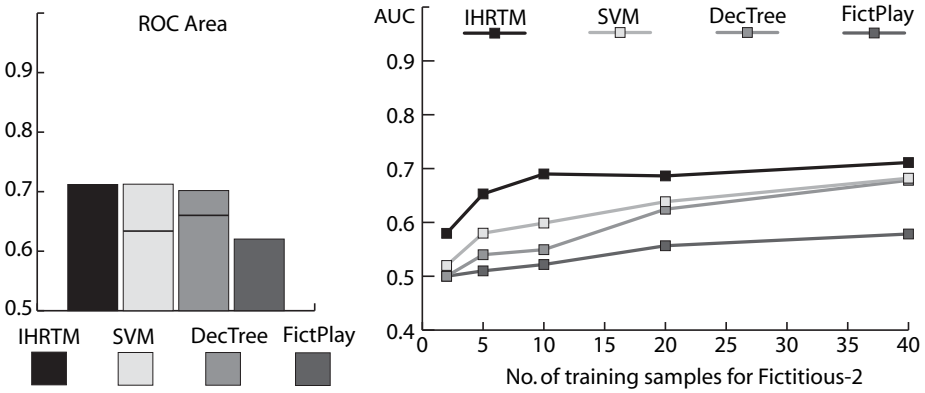

Figure 5.11: Results for play against *Fictitious*. Bar graph on the left: AUC for classifying Att^t. Graph on the right: learning curve for increasing number of training data for the additional *Fictitious-2*.

samples. On the right of Figure 5.10 the learning rates for agent *Honest-2* are plotted. The results confirm that especially for a small sample size ≤ 20 the performance of IHSTM is clearly better compared to the content based approaches.

In contrast, the performance in the task of trying to predict *Fictitious* is clearly worse for all of the techniques (see Figure 5.11). Expectedly, IHSTM, SVM and DecTree cannot handle dynamic opponents. Again, the IHSTM is most competitive in terms of efficient learning.

In addition, the IHSTM offers another advantage over the other techniques. The predictions are based on an inherent construction of clusters of Z^a and Z^s. The fast learning rate indicates that a previously unknown trustee is correctly assigned to an existing cluster if this type of agent has been observed before. Consequently, once *Fictitious-2* is assigned to the '*Fictitious*-cluster' IHSTM could assess its performance on this cluster and eventually suggest a different learning scheme for agents in this cluster. In other words it can identify non-stationary behaving agents.

Figure 5.12 visualizes the final cluster sizes and cluster assignments. The top right matrix shows the assignment of seven different agents to Z^a. All three agent types were clustered correctly into three groups (columns). To evaluate this further we generated data from another stationary opponent with a different trading strategy that is very similar to *Honest*: *Sneaky*-agent only deviates from c if it can increases its utility by a large margin. Interestingly, the assignment of *Sneaky*- and *Honest*-agent to the same cluster suggests that this strategy might effectively build trust.The matrix in the lower left corner of Figure 5.12 visualize Z^s. From 289 stage games (columns) 8 different clusters (rows) emerged. This is an impressive reduction in complexity while still having good classification results. The two stacked matrices in the bottom right corner represent Att^t and Att^c (below). Each row indicates one state cluster, each column an agent cluster. Brighter rectangles indicate a lower probability for a positive reward. As expected, the first column (*Greedy* cluster) is on average brighter than the second and third column (*Honest* and *Fictitious* cluster). All those observations, including the misclassification of *Sneaky*, correspond well

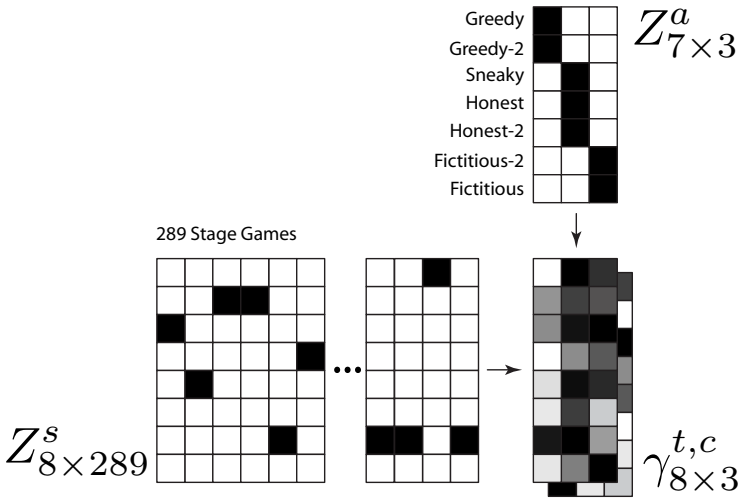

Figure 5.12: Final clustering of agent types and states (Z^s and Z^a). Bottom right: $P(\gamma^t, \gamma^c | Z^s, Z^a)$

to human intuition.

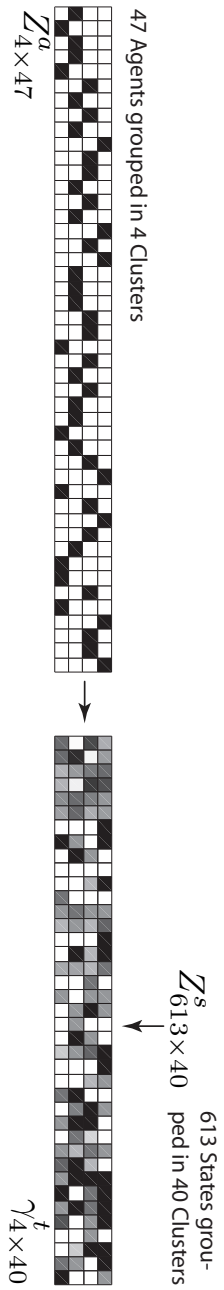

Figure 5.13: Left: Final clustering of trustees Z^a. Top right: Items Z^s. Bottom right: $P(\gamma^t|Z^a, Z^s)$

Chapter 6

Conclusions

We conclude this thesis by summarizing the contributions of the proposed methods and contemplating about promising directions of future research.

6.1 Contributions

In this thesis, the integration of formal ontologies as prior knowledge into machine learning tasks was explored. The Infinite Hidden Semantic Model (IHSM) was developed, a machine learning method from the area of non-parametric latent class graphical models which was upgraded from multi-relational representations to leverage expressive constructs in formal ontologies. IHSM contributes to the integration of inductive and deductive reasoning approaches and more specifically to learning with description logic ontologies and from untrustworthy sources.

Empirical evidence in the context of social network analysis was presented that hard constraints cannot only improve predictive performance of unknown roles, which are directly affected by the constraints, but also unconstraint roles via IHSMs latent variables. This is the commonly used notion of cannot-link constraints is replaced with the more general notion of logical (un-)satisfiability w.r.t. formal background knowledge. Thus, IHSM can leverage the power of logical reasoning to enhance inference about unknown facts in knowledge bases. In our experiments this leads to an improved statistical analysis particularly concerning the predictive performance.

In addition, a context-dependent way to build statistical relational trust models in general and our *Infinite Hidden Semantic Trust Model* (IHSTM) in particular was presented. It was demonstrated how computational trust (CT) can be modeled and learned in theory as well as in two experimental setups: first, a real world data set from the *eBay feedback-system* and second, a simulated negotiation game.

We showed that IHSTM is especially useful for trust learning in initial trust situations, where the trustor interacts with other agents without having recorded sufficiently enough experiences in order to judge trustability using traditional methods. For instance, this would typically be the case in open information markets, where potentially unknown information sellers and buyers interact with each other.

Our experimental results suggest that IHSTM offers advantages in three different dimensions. First, the inherent clustering capabilities increase interpretability of trust situations. Second, the predictive performance can be improved compared to a 'flat', feature-based or single-relational machine learning approach if trained with relational data that exhibit cross-attribute and cross-entity dependencies. Third, IHSTM is especially well suited for rapid adaptation because of its ability to transfer knowledge between related contexts.

6.1.1 Contributions in Detail

The introductory chapter presented thoughts on the recent trend towards semantic knowledge representation illustrated by popular commercial systems (see Sec. 1.2.1). Next, some existing applications of commercial semantic systems were given (see Sec. 1.2.2), before possibilities and challenges for ML using such systems were discussed (see Sec. 1.3).

Ch. 2 started with a short history of ontologies (see Sec. 2.1) accompanied by a tabular list of terms 2.1 used for components of knowledge representations. The main contribution of this chapter was a hierarchy of knowledge representations (see Sec. 2.2) ranging from semi-formal knowledge representations (see Sec. 2.2.1) to formal ontologies (see Sec. 2.2.2) and finally expressive formal ontologies (see Sec. 2.2.3). In the process a formal definition of ontologies was given (see Sec. 2.2.2) and existing real world data sets were presented (see Sec. 2.2.2). In this context the paradox of real world expressive formal ontologies was discussed (see Sec. 2.2.3). Next, the concrete ontology language OWL which was used as a knowledge representation in this thesis was introduced by outlining RDF(S) (see Sec. 2.3.1) and specifying the syntax and semantics of OWL DL (see Sec. 2.3.2. Finally, existing probabilistic extensions of SW ontology languages were presented (see Sec. 2.3.4).

The main contribution of Ch. 3 was a novel hierarchy of ML formats starting from traditional vector-based learning methods (see Sec. 3.1) towards highly-expressive first-order probabilistic logic learning methods (see Sec. 3.4.2). In the first step, the feature vector input and single label output approach (see Sec. 3.1) was extended to single matrix based inputs and outputs (see Sec. 3.2). Next, this was expanded to multi-relational based inputs and outputs (see Sec. 3.3) with the special case of learning with latent graphical models (see Sec. 3.3.2). In this section, our approach to extracting a multi-relational representations from ontologies was proposed (see Sec. 3.3.1). The hierarchy was completed with an outlook on developments towards lifted first-order probabilistic inference and learning (see Sec. 3.4.2). In every step of the hierarchy the representation introduced was accompanied by existing learning methods and available real world datasets using this representation.

In Ch. 4 the main contribution of this thesis was proposed, namely the IHSM which combines non-parametric latent class graphical models with logical satisfiability w.r.t. formal background knowledge. After discussing related work (see Sec. 4.1) the formal framework in the form of description logic ontologies (see Sec. 4.2) and the Bayesian framework in the form of Dirichlet process mixture models (see Sec. 4.3) was defined. Then, novel algorithms for a constrained generative model, constrained learning and constrained inference were introduced (see Sec. 4.4). It was described how those algorithms were implemented and applied to data, e. g. , a real world social network (see Sec. 4.5), before empirical

results were presented (see Sec. 4.6). Hereby, the computational complexity, clustering capabilities and improved predictive performance were evaluated by comparing IHSM to the unconstrained version and in turn to related approaches.

Finally, Ch. 5 presented the second major contribution of this thesis, the IHSTM. A motivation for the need to conceptualize, use and learn trust in collaborative ontologies was given (see Sec. 5.1), before basic trust concepts (see Sec. 5.1.1) and related work were outlined (see Sec. 5.1.2). Next, we presented novel algorithms how to model (see Sec. 5.2.1) and learn (see Sec. 5.2.3) CT based on IHSM. The practicability and effectiveness of this approach was evaluated empirically (see Sec. 5.4) on different data sets, including user-ratings gathered from eBay (see Sec. 5.3). The results demonstrated that IHSTM offers a meaningful trust assessments (see Sec. 5.4.1), while improving trust assessment performance (see Sec. 5.4.2) as well as learning faster and transferring knowledge more effectively (see Sec. 5.4.3).

6.2 Future Work

While IHSM and IHSTM did successfully address some key challenges in integrating formal knowledge bases and assessing computational trust using machine learning there still remain challenging research issues and new ones become evident in respect to different research directions.

On the one hand, there is considerable need for more theoretical analysis of the proposed algorithms. Although latent class graphical models and description logic are based on a well founded theoretical background the combination of both still is a challenge. Comparing the different expressivity and the unification of existing knowledge representations, e. g. , ER-models, tags or Markov logic in a theoretical manner would be very helpful. For the IHSM in particular, future work could concern a detailed theoretical analysis of the effect of constraining on clusters.

Refining the ontology by extracting formal knowledge from the latent model structure (related to predicate invention) is another promising research direction. Automatically integrating learned probabilities in formal ontologies poses a similar challenge. Ultimately, research is directed towards lifted first-order probabilistic inference and learning (cmp. Sec. 3.4).

Besides that, trust is a concept that is inherently incomprehensible and hence CT will remain an open issue at least to a large extend. For instance, IHSTM cannot handle trustees with strategies that are non-stationary effectively, although it can identify non-stationary agents (cmp. Sec. 5.4.3). An adaptive learning strategy could be part of future work. Furthermore, we plan to extend our framework to scenarios with arbitrary numbers of concurrently interacting trustees.

On the other hand, many real world challenges remain. In order to require a minimum of user intervention machine learning should become more 'push-button'. Although, IHSM is a non-parametric approach, handling and tuning the hyper-parameters requires substantial expertise. The effect of tuning the hyper-parameters of a Dirichlet process and of the Gibbs sampling parameters is only comprehensible for experts (cmp. Sec. 4.4). Future work should be concerned with finding automated ways for parameter optimization.

Furthermore, the learning time and space complexity should scale well with

the size of the ontology. It is to be expected, that the size of datasets on the SW will become enormous, causing special demands on computational efficiency. Calculating a global probabilistic model is elegant, but might not be feasible in many application. Discriminative models and more efficient approximate inference are essential tasks for future research. Starting points for improving the efficiency of IHSM would be a DL reasoner optimized for handling instance data. So far, DL reasoner are focused on handling TBox and not ABox knowledge efficiently. Another obvious improvement would be a caching mechanism for satisfiability checks. The DL reasoner implemented is reset every iteration, so identical cluster assignments cannot be utilized (cmp. Sec. 4.4.2). A more sophisticated improvement would be an approach to lifted satisfiability checks, where instances are 'tied together' (like in tying of parameters) and checks are performed for an abstract class of instances only once.

Machine learning should also be specifically suitable to the data situation on the SW with sparse data (e. g. , only a small number persons are friends) and missing information (e. g. , some people do not reveal private information) (cmp. Sec. 4.5). In many cases, link information is so sparse that feature based algorithms are the most promising approach. Here, deductive inference might play a key role in complementing sparse data, which is one issue IHSM addresses.

However, the most pressing issue is concerned with real world formal ontologies. As discussed in Sec. 2.2.3 the dilemma of real world formal ontologies is that they either are expressive or large scale, but not both. This constricts the development and application of powerful probabilistic inference and learning tools in general and makes a large scale evaluation of IHSM difficult. We hope to see more work on inductive learning with SW ontologies and on the other hand complex SW ontologies that can be supplemented with uncertain evidence.

While formal data sets are becoming available (cmp. Sec. 2.2.2) they lack expressive formal semantics which can only be added by humans. However, providing expressive semantics to formal data is still inconvenient. Even the handling of formal data is not yet easily accessible for users. Ultimately, a new human-computer interface is needed for formal data. For instance, user input needs to be more than a keyword query to leverage the potential of ontologies and to enrich them with semantics. Like 'web 2.0' was about making 'web 1.0' more accessible to the user the next step is to make access to ontologies more meaningful and reciprocal.

While, the SW is starting to produce large scale expressive formal ontologies, machine learning has the potential to realize a number of exciting applications using and complementing this resource - if axiomatic inference can be integrated with exploitation of statistical regularities.

Bibliography

[Airoldi et al., 2008] Airoldi, E., Blei, D., Fienberg, S., and Xing, E. (2008). Mixed membership stochastic blockmodels. *The Journal of Machine Learning Research*, 9:1981–2014.

[Aldous, 1985] Aldous, D. (1985). Exchangeability and related topics. In *Ecole d'Ete de Probabilites de Saint-Flour XIII 1983*, pages 1–198. Springer.

[Ashburner et al., 2000] Ashburner, M., Ball, C., Blake, J., Botstein, D., Butler, H., Cherry, J., Davis, A., Dolinski, K., Dwight, S., Eppig, J., et al. (2000). Gene Ontology: tool for the unification of biology. *Nature genetics*, 25(1):25–29.

[Ashri et al., 2005] Ashri, R., Ramchurn, S. D., Sabater, J., Luck, M., and Jennings, N. R. (2005). Trust evaluation through relationship analysis. In *AAMAS '05: Proceedings of the fourth international joint conference on Autonomous agents and multiagent systems*, pages 1005–1011, New York, NY, USA. ACM Press.

[Baader et al., 2003] Baader, F., Calvanese, D., McGuinness, D., Nardi, D., and Patel-Schneider, P., editors (2003). *The Description Logic Handbook*. Cambridge University Press.

[Baader et al., 2007] Baader, F., Horrocks, I., and Sattler, U. (2007). Description Logics. In van Harmelen, F., Lifschitz, V., and Porter, B., editors, *Handbook of Knowledge Representation*. Elsevier.

[Baader and Sattler, 2000] Baader, F. and Sattler, U. (2000). Tableau algorithms for description logics. In *Automated Reasoning with Analytic Tableaux and Related Methods*, pages 1–18.

[Basu et al., 2008] Basu, S., Davidson, I., and Wagstaff, K. (2008). *Constrained clustering: Advances in algorithms, theory, and applications*. Chapman & Hall/CRC.

[Bickel et al., 2007] Bickel, S., Brückner, M., and Scheffer, T. (2007). Discriminative learning for differing training and test distributions. In *ICML '07: Proceedings of the 24th international conference on Machine learning*, pages 81–88, New York, NY, USA. ACM.

[Bizer et al., 2009] Bizer, C., Heath, T., and Berners-Lee, T. (2009). Linked data - the story so far. *International Journal on Semantic Web and Information Systems*.

[Blei et al., 2003] Blei, D. M., Ng, A. Y., and Jordan, M. I. (2003). Latent dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022.

[Bloehdorn and Sure, 2007] Bloehdorn, S. and Sure, Y. (2007). Kernel methods for mining instance data in ontologies. In Aberer, K. and et al., editors, *Proceedings of the 6th International Semantic Web Conference, ISWC2007*, volume 4825 of *LNCS*, pages 58–71. Springer.

[Brachman and Schmolze, 1985] Brachman, R. and Schmolze, J. (1985). An overview of the KL-ONE knowledge representation system. *Cognitive science*, 9(2):171–216.

[Brown, 1951] Brown, G. (1951). Iterative solution of games by fictitious play. In *Activity Analysis of Production and Allocation*. New York: John Wiley and Sons.

[Bu et al., 2003] Bu, D., Zhao, Y., Cai, L., Xue, H., Zhu, X., Lu, H., Zhang, J., Sun, S., Ling, L., Zhang, N., et al. (2003). Topological structure analysis of the protein-protein interaction network in budding yeast. *Nucleic Acids Research*, 31(9):2443.

[Bundschus et al., 2009] Bundschus, M., Yu, S., Tresp, V., Rettinger, A., Dejori, M., and Kriegel, H.-P. (2009). Hierarchical bayesian models for collaborative tagging systems. In *IEEE International Conference on Data Mining series (ICDM 2009)*.

[Carbo et al., 2005] Carbo, J., Garcia, J., and Molina, J. M. (2005). Contribution of referrals in a reputation model based on kalman filtering vs fuzzy sets. In *Proceedings of the Workshop on Trust in Agent Societies (held in conjunction with the 4th International Joint Conference on Autonomous Agents and Multi Agent Systems (AAMAS))*.

[Carbonetto et al., 2005] Carbonetto, P., Kisynski, J., de Freitas, N., and Poole, D. (2005). Nonparametric bayesian logic. In *Proc. 21st UAI*.

[Carroll et al., 2005] Carroll, J. J., Bizer, C., Hayes, P., and Stickler, P. (2005). Named graphs, provenance and trust. In *WWW '05: Proceedings of the 14th international conference on World Wide Web*, pages 613–622, New York, NY, USA. ACM.

[Caruana, 1997] Caruana, R. (1997). Multitask learning. *Mach. Learn.*, 28(1):41–75.

[Casella and Berger, 1990] Casella, G. and Berger, R. L. (1990). *Statistical Inference*. Duxbury Press.

[Chadwick, 2005] Chadwick, D. W. (2005). Operational models for reputation servers. In [Herrmann et al., 2005], pages 108–115.

[Corman et al., 2002] Corman, S., Kuhn, T., McPhee, R., and Dooley, K. (2002). Studying complex discursive systems. *Human Communication Research*, 28(2):157–206.

[Cumby and Roth, 2003] Cumby, C. and Roth, D. (2003). On kernel methods for relational learning. In Fawcett, T. and N.Mishra, editors, *Proceedings of the 20th International Conference on Machine Learning, ICML2003*, pages 107–114. AAAI Press.

[Cussens, 2007] Cussens, J. (2007). Logic-based formalisms for statistical relational learning. In Press, M. M., editor, *Introduction to statistical relational learning*. Morgan Kaufmann Publishers, San Mateo, CA.

[d'Amato et al., 2008] d'Amato, C., Fanizzi, N., and Esposito., F. (2008). Query answering and ontology population: An inductive approach. In Bechhofer, S. and et al., editors, *Proceedings of the 5th European Semantic Web Conference, ESWC2008*, volume 5021 of *LNCS*, pages 288–302. Springer.

[Daumé and Marcu, 2006] Daumé, H. and Marcu, D. (2006). Domain adaptation for statistical classifiers. *J. Artif. Intell. Res. (JAIR)*, 26:101–126.

[Davidson and Ravi, 2007] Davidson, I. and Ravi, S. S. (2007). The complexity of non-hierarchical clustering with instance and cluster level constraints. *Data Min. Knowl. Discov.*, 14(1):25–61.

[de Oliveira, 2009] de Oliveira, P. C. (2009). Probabilistic reasoning in the semantic web using markov logic. Master's thesis, Knowledge and Intelligent Systems Laboratory Cognitive and Media Systems Group Centre for Informatics and Systems of the University of Coimbra, Portugal.

[De Salvo Braz et al., 2005] De Salvo Braz, R., Amir, E., and Roth, D. (2005). Lifted first-order probabilistic inference. In *IJCAI'05: Proceedings of the 19th international joint conference on Artificial intelligence*, pages 1319–1325, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

[de Salvo Braz et al., 2007] de Salvo Braz, R., Amir, E., and Roth, D. (2007). Lifted first-order probabilistic inference. In Getoor, L. and Taskar, B., editors, *Introduction to Statistical Relational Learning*. MIT Press.

[Dietterich, 2002] Dietterich, T. G. (2002). Machine learning for sequential data: A review. In *Proceedings of the Joint IAPR International Workshop on Structural, Syntactic, and Statistical Pattern Recognition*, pages 15–30, London, UK. Springer-Verlag.

[Ding et al., 2009] Ding, C., Wang, F., and Li, T. (2009). Data mining with graphs and matrices. Tutorial at the International Confernece on Data Mining 2009.

[Ding, 2005] Ding, Z. (2005). *BayesOWL: A Probabilistic Framework for Semantic Web*. PhD thesis, University of Maryland, Baltimore County.

[Domingos et al., 2008] Domingos, P., Lowd, D., Kok, S., Poon, H., Richardson, M., and Singla, P. (2008). Just add weights: Markov logic for the semantic web. pages 1–25.

[Domingos and Richardson, 2007] Domingos, P. and Richardson, M. (2007). Markov logic: A unifying framework for statistical relational learning. In Getoor, L. and Taskar, B., editors, *Introduction to Statistical Relational Learning*. MIT Press.

[Endriss, 2006] Endriss, U. (2006). Monotonic concession protocols for multilateral negotiation. In *AAMAS '06: Proceedings of the fifth international joint conference on Autonomous agents and multiagent systems*, pages 392–399, New York, NY, USA. ACM Press.

[Fahrmeir et al., 2004] Fahrmeir, L., Künstler, R., Pigeot, I., Tutz, G., Caputo, A., and Lang, S. (2004). *Arbeitsbuch Statistik*. Springer, fourth edition.

[Fanizzi et al., 2008a] Fanizzi, N., d'Amato, C., and Esposito, F. (2008a). Evolutionary conceptual clustering based on induced pseudo-metrics. *Semantic Web Information Systems*, 4(3):44–67.

[Fanizzi et al., 2008b] Fanizzi, N., d'Amato, C., and Esposito, F. (2008b). Learning with kernels in description logics. In Zelezný, F. and Lavrač, N., editors, *Proceedings of the 18th International Conference on Inductive Logic Programming, ILP2008*, volume 5194 of *LNAI*, pages 210–225. Springer.

[Fanizzi et al., 2008c] Fanizzi, N., D'Amato, C., and Esposito, F. (2008c). A multi-relational hierarchical clustering method for datalog knowledge bases. In *Foundations of Intelligent Systems*. Springer Berlin / Heidelberg.

[Fanizzi et al., 2009] Fanizzi, N., d'Amato, C., and Esposito, F. (2009). ReduCE: A reduced coulomb energy network method for approximate classification. In Aroyo, L. and et al., editors, *Proceedings of the 6th European Semantic Web Conference, ESWC2009*, volume 5554 of *LNCS*, pages 323–337. Springer.

[Fatima et al., 2005] Fatima, S. S., Wooldridge, M., and Jennings, N. R. (2005). A comparative study of game theoretic and evolutionary models of bargaining for software agents. *Artif. Intell. Rev.*, 23(2):187–205.

[Först et al., 2009] Först, A., Rettinger, A., and Nickles, M. (2009). Argumentation- vs. proposal-based negotiation: An empirical case study on the basis of game-theoretic solution concepts. pages 161–180.

[Frasconi, 2007] Frasconi, P. (2007). Learning with kernels and logical representations. In Blockeel, H., Ramon, J., Shavlik, J. W., and Tadepalli, P., editors, *ILP*, volume 4894 of *Lecture Notes in Computer Science*, pages 1–3. Springer.

[Fullam et al., 2005] Fullam, K. K., Klos, T. B., Muller, G., Sabater, J., Topol, Z., Barber, K. S., Rosenschein, J. S., and Vercouter, L. (2005). The agent reputation and trust (art) testbed architecture. In *The Eighth Workshop on Trust in Agent Societies (TRUST 2005), at The Fourth International Joint Conference on Autonomous Agents and Multiagent Systems*, Utrecht, The Netherlands.

[Gambetta, 2001] Gambetta, D. (2001). Kann man dem vertauene vertrauen? In Hartmann, M. and Offe, C., editors, *Vertrauen. Die Grundlage des sozialen Zusammenhalts.*, pages 204–240. Campus Verlag.

[Gärtner et al., 2004] Gärtner, T., Lloyd, J., and Flach, P. (2004). Kernels and distances for structured data. *Machine Learning*, 57(3):205–232.

[Getoor et al., 2007] Getoor, L., Friedman, N., Koller, D., Pferrer, A., and Taskar, B. (2007). Probabilistic relational models. In Getoor, L. and Taskar, B., editors, *Introduction to Statistical Relational Learning*. MIT Press.

[Getoor and Taskar, 2007] Getoor, L. and Taskar, B., editors (2007). *Introduction to Statistical Relational Learning*. The MIT Press.

[Giugno and Lukasiewicz, 2002] Giugno, R. and Lukasiewicz, T. (2002). P-shoq(d): A probabilistic extension of shoq(d) for probabilistic ontologies in the semantic web. In *JELIA '02: Proceedings of the European Conference on Logics in Artificial Intelligence*, pages 86–97, London, UK. Springer-Verlag.

[Gruber et al., 1993] Gruber, T. et al. (1993). A translation approach to portable ontology specifications. *Knowledge acquisition*, 5:199–199.

[Guarino, 1998] Guarino, N. (1998). Formal ontology and information systems, proceedings of fois'98. In Guarino, N., editor, *Formal Ontology in Information Systems*, pages 3–15. IOS Press.

[Hall et al., 2009] Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., and Witten, I. H. (2009). The weka data mining software: An update. *SIGKDD Explorations*, 11(1).

[Hassell, 2005] Hassell, L. (2005). Affect and trust. In [Herrmann et al., 2005], pages 131–145.

[Hastie et al., 2001] Hastie, T., Tibshirani, R., and Friedman, J. (2001). *The Elements of Statistical Learning*. Springer Series in Statistics. Springer New York Inc., New York, NY, USA.

[Hausenblas et al., 2008] Hausenblas, M., Halb, W., Raimond, Y., and Heath, T. (2008). What is the Size of the Semantic Web? In *I-Semantics 2008: International Conference on Semantic Systems*, Graz, Austria.

[Heckerman et al., 2004] Heckerman, D., Meek, C., and Koller, D. (2004). Probabilistic models for relational data. Technical Report MSR-TR-2004-30, Microsoft Research.

[Hepp, 2007] Hepp, M. (2007). Possible ontologies: How reality constrains the development of relevant ontologies. *IEEE Internet Computing*, 11:90–96.

[Herrmann et al., 2005] Herrmann, P., Issarny, V., and Shiu, S., editors (2005). *Trust Management, Third International Conference, iTrust 2005, Paris, France, May 23-26, 2005, Proceedings*, volume 3477 of *Lecture Notes in Computer Science*. Springer.

[Heymann et al., 2008] Heymann, P., Ramage, D., and Garcia-Molina, H. (2008). Social tag prediction. In *SIGIR '08: Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 531–538, New York, NY, USA. ACM.

[Hitzler et al., 2008] Hitzler, P., Krötzsch, M., Rudolph, S., and Sure, Y. (2008). *Semantic Web*. Springer.

[Horrocks et al., 2005] Horrocks, I., Parsia, B., Patel-Schneider, P. F., and Hendler, J. A. (2005). Semantic web architecture: Stack or two towers? In *PPSWR*, pages 37–41.

[Horrocks and Patel-Schneider, 2003] Horrocks, I. and Patel-Schneider, P. F. (2003). Reducing owl entailment to description logic satisfiability. In *Journal of Web Semantics*, pages 17–29. Springer.

[Hu and Wellman, 1998] Hu, J. and Wellman, M. P. (1998). Multiagent reinforcement learning: Theoretical framework and an algorithm. In Shavlik, J. W., editor, *ICML*, pages 242–250. Morgan Kaufmann.

[Huang et al., 2009] Huang, Y., Tresp, V., Bundschus, M., and Rettinger, A. (2009). Scalable relational learning for sparse and incomplete domains. In *Proceedings of the International Workshop on Statistical Relational Learning (SRL-2009)*.

[Ishwaran and James, 2001] Ishwaran, H. and James, L. (2001). Gibbs sampling methods for stick breaking priors. *Journal of the American Statistical Association*, 96(453):161–173.

[Jaeger, 1997] Jaeger, M. (1997). Relational bayesian networks. In *Proceedings of the 13th Conference on Uncertainty in Artificial Intelligence (UAI)*.

[Jäschke et al., 2007] Jäschke, R., Marinho, L., Hotho, A., Schmidt-Thieme, L., and Stumme, G. (2007). Tag recommendations in folksonomies. pages 506–514.

[Jensen et al., 2004] Jensen, C. D., Poslad, S., and Dimitrakos, T., editors (2004). *Trust Management, Second International Conference, iTrust 2004, Oxford, UK, March 29 - April 1, 2004, Proceedings*, volume 2995 of *Lecture Notes in Computer Science*. Springer.

[Jøsang et al., 2005] Jøsang, A., Keser, C., and Dimitrakos, T. (2005). Can we manage trust? In [Herrmann et al., 2005], pages 93–107.

[Kemp et al., 2006] Kemp, C., Tenenbaum, J. B., Griffiths, T. L., Yamada, T., and Ueda, N. (2006). Learning systems of concepts with an infinite relational model. In *Poceedings of the National Conference on Artificial Intelligence (AAAI)*.

[Kersting and De Raedt, 2001] Kersting, K. and De Raedt, L. (2001). Bayesian logic programs. Technical report, Albert-Ludwigs University at Freiburg.

[Kiefer et al., 2008] Kiefer, C., Bernstein, A., and Locher, A. (2008). Adding Data Mining Support to SPARQL via Statistical Relational Learning Methods. In *Proceedings of the 5th European Semantic Web Conference (ESWC)*, volume 5021 of *Lecture Notes in Computer Science*, pages 478–492. Springer-Verlag Berlin Heidelberg.

[Kimbrough, 2005] Kimbrough, S. O. (2005). Foraging for trust: Exploring rationality and the stag hunt game. In [Herrmann et al., 2005], pages 1–16.

[Kinateder et al., 2005] Kinateder, M., Baschny, E., and Rothermel, K. (2005). Towards a generic trust model - comparison of various trust update algorithms. In [Herrmann et al., 2005], pages 177–192.

[Klos and la Poutré, 2005] Klos, T. and la Poutré, H. (2005). Decentralized reputation-based trust for assessing agent reliability under aggregate feedback. In Falcone, R., Barber, S., Sabater, J., and Singh, M., editors, *Trusting Agents for trusting Electronic Societies*. SPRINGER.

[Kok and Domingos, 2005] Kok, S. and Domingos, P. (2005). Learning the structure of markov logic networks. In *ICML '05: Proceedings of the 22nd international conference on Machine learning*, pages 441–448, New York, NY, USA. ACM.

[Kok and Domingos, 2007] Kok, S. and Domingos, P. (2007). Statistical predicate invention. In *ICML '07: Proceedings of the 24th international conference on Machine learning*, pages 433–440, New York, NY, USA. ACM.

[Koller et al., 1997] Koller, D., Levy, A. Y., and Pfeffer, A. (1997). P-CLASSIC: A tractable probablistic description logic. In *AAAI/IAAI*, pages 390–397.

[Koller and Pfeffer, 1998] Koller, D. and Pfeffer, A. (1998). Probabilistic frame-based systems. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*.

[Krogel, 2005] Krogel, M.-A. (2005). *On Propositionalization for Knowledge Discovery in Relational Databases*. PhD thesis, die Fakultät für Informatik der Otto-von-Guericke-Universität Magdeburg.

[Lee and Seung, 1999] Lee, D. D. and Seung, H. S. (1999). Learning the parts of objects by non-negative matrix factorization. *Nature*.

[Linden et al., 2003] Linden, G., Smith, B., and York, J. (2003). Amazon.com recommendations: item-to-item collaborative filtering. *Internet Computing, IEEE*, 7(1):76–80.

[Lippert et al., 2008] Lippert, C., Huang, Y., Weber, S. H., Tresp, V., Schubert, M., and Kriegel, H.-P. (2008). Relation prediction in multi-relational domains using matrix factorization. Technical report, Siemens.

[Lisi and Esposito, 2007] Lisi, F. A. and Esposito, F. (2007). On Ontologies as Prior Conceptual Knowledge in Inductive Logic Programming. In *ECML PKDD 2008 Workshop: Prior Conceptual Knowledge in Machine Learning and Knowledge Discovery PriCKL'07*.

[Littman, 1994] Littman, M. L. (1994). Markov games as a framework for multi-agent reinforcement learning. In *ICML*, pages 157–163.

[Luhmann, 2001] Luhmann, N. (2001). Vertrautheit, zuversicht, vertrauen. probleme und alternativen. In Hartmann, M. and Offe, C., editors, *Vertrauen. Die Grundlage des sozialen Zusammenhalts.*, pages 143–160. Campus Verlag.

[Lukasiewicz, 2007] Lukasiewicz, T. (2007). Probabilistic description logic programs. *Int. J. Approx. Reasoning*, 45(2):288–307.

[Maedche and Staab, 2004] Maedche, A. and Staab, S. (2004). Ontology learning. *Handbook on Ontol ogies*, pages 173–190.

[Marsh and Dibben, 2005] Marsh, S. and Dibben, M. R. (2005). Trust, untrust, distrust and mistrust - an exploration of the dark(er) side. In [Herrmann et al., 2005], pages 17–33.

[Maximilien and Singh, 2005] Maximilien, E. M. and Singh, M. P. (2005). Agent-based trust model involving multiple qualities. In *Proceedings of the 4th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS)*.

[Meilă, 2005] Meilă, M. (2005). Comparing clusterings: an axiomatic view. In *ICML '05: Proceedings of the 22nd international conference on Machine learning*, pages 577–584, New York, NY, USA. ACM Press.

[Mika, 2007] Mika, P. (2007). Social Networks and the Semantic Web (Semantic Web and Beyond).

[Milch et al., 2008] Milch, B., Zettlemoyer, L. S., Kersting, K., Haimes, M., and Kaelbling, L. P. (2008). Lifted probabilistic inference with counting formulas. In *AAAI'08: Proceedings of the 23rd national conference on Artificial intelligence*, pages 1062–1068. AAAI Press.

[Minsky, 1974] Minsky, M. (1974). A framework for representing knowledge.

[Moranz, 2004] Moranz, C. (2004). *Initiales Vertrauen in der virtuellen Anbahnung geschftlicher Kooperationen*. PhD thesis, Technische Universtitt Mnchen, Lehrstuhl fr Psychologie.

[Muggleton, 1996] Muggleton, S. (1996). Stochastic logic programs. In *New Generation Computing*. Academic Press.

[Murray and Gordon, 2007] Murray, C. and Gordon, G. (2007). Multi-robot negotiation: Approximating the set of subgame perfect equilibria in general sum stochastic games. In Schölkopf, B., Platt, J., and Hoffman, T., editors, *Advances in Neural Information Processing Systems 19*. MIT Press, Cambridge, MA.

[N. Fanizzi, 2008] N. Fanizzi, C. d'Amato, F. E. (2008). Induction of classifiers through non-parametric methods for approximate classification and retrieval with ontologies. *International Journal of Semantic Computing vol.2(3)*, pages 403 – 423.

[Neville and Jensen, 2004] Neville, J. and Jensen, D. (2004). Dependency networks for relational data. In *ICDM '04: Proceedings of the Fourth IEEE International Conference on Data Mining*.

[Newman and Girvan, 2004] Newman, M. and Girvan, M. (2004). Finding and evaluating community structure in networks. *Physical review E*, 69(2):26113.

[Ng and Subrahmanian, 1990] Ng, R. T. and Subrahmanian, V. S. (1990). A semantical framework for supporting subjective and conditional probabilities in deductive databases. Technical report, College Park, MD, USA.

[Nowicki and Snijders, 2001] Nowicki, K. and Snijders, T. (2001). Estimation and Prediction for Stochastic Blockstructures. *Journal of the American Statistical Association*, 96(455).

[Poole, 1997] Poole, D. (1997). The independent choice logic for modelling multiple agents under uncertainty. *Artif. Intell.*, 94(1-2):7–56.

[Poole, 2003] Poole, D. (2003). First-order probabilistic inference. In *IJCAI'03: Proceedings of the 18th international joint conference on Artificial intelligence*, pages 985–991, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

[Predoiu, 2006] Predoiu, L. (2006). Information integration with bayesian description logic programs. In *Proceedings of the Workshop on Information Integration on the Web (IIWeb 2006), in conjunction with WWW2006*, Edinburgh, Scotland.

[Predoiu and Stuckenschmidt, 2008] Predoiu, L. and Stuckenschmidt, H. (2008). Probabilistic extensions of semantic web languages - a survey. In Ma, Z. and Wang, H., editors, *The Semantic Web for Knowledge and Data Management: Technologies and Practices*, chapter 5. Idea Group Inc.

[Punyakanok et al., 2005] Punyakanok, V., Roth, D., Yih, W.-t., and Zimak, D. (2005). Learning and inference over constrained output. In *IJCAI'05: Proceedings of the 19th international joint conference on Artificial intelligence*, pages 1124–1129, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

[Raedt, 2008] Raedt, L. D. (2008). *Logical and Relational Learning: From ILP to MRDM (Cognitive Technologies)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA.

[Raedt et al., 2008] Raedt, L. D., Frasconi, P., Kersting, K., and Muggleton, S., editors (2008). *Probabilistic Inductive Logic Programming - Theory and Applications*, volume 4911 of *Lecture Notes in Computer Science*. Springer.

[Raedt and Kersting, 2003] Raedt, L. D. and Kersting, K. (2003). Probabilistic logic learning. *SIGKDD Explor. Newsl.*, 5(1):31–48.

[Ramchurn et al., 2004] Ramchurn, S. D., Hunyh, D., and Jennings, N. R. (2004). Trust in multi-agent systems. *Knowledge Engineering Review*.

[Raphael, 1964] Raphael, B. (1964). *SIR: A Computer Program for Semantic Information Retrieval*. PhD thesis, MIT.

[Reckow and Tresp, 2008] Reckow, S. and Tresp, V. (2008). Integrating Ontological Prior Knowledge into Relational Learning. In *NIPS 2008 Workshop: Structured Input - Structured Output*.

[Rehak and Pechoucek, 2007] Rehak, M. and Pechoucek, M. (2007). Trust modeling with context representation and generalized identities. In Klusch, M., Hindriks, K. V., Papazoglou, M. P., and Sterling, L., editors, *CIA*, volume 4676 of *Lecture Notes in Computer Science*, pages 298–312. Springer.

[Rehak et al., 2005] Rehak, M., Pechoucek, M., Benda, P., and Foltyn, L. (2005). Trust in coalition environment: Fuzzy number approach. In *The Eighth Workshop on Trust in Agent Societies (TRUST 2005), at The Fourth International Joint Conference on Autonomous Agents and Multiagent Systems*, Utrecht, The Netherlands.

[Resnick et al., 1994a] Resnick, P., Iacovou, N., Suchak, M., Bergstorm, P., and Riedl, J. (1994a). Grouplens: An open architecture for collaborative filtering of netnews. In *Proceedings of ACM 1994 Conference on Computer Supported Cooperative Work*, pages 175–186, Chapel Hill, North Carolina. ACM.

[Resnick et al., 1994b] Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P., and Riedl, J. (1994b). Grouplens: An open architecture for collaborative filtering of netnews. In *Proc. of the ACM 1994 Conference on Computer Supported Cooperative Work*, pages 175–186. ACM.

[Rettinger and Nickles, 2009] Rettinger, A. and Nickles, M. (2009). Infinite hidden semantic models for learning with owl dl. In *In Proceedings of the First ESWC Workshop on Inductive Reasoning and Machine Learning on the Semantic Web (IRMLeS 2009)*. CEUR Workshop Proceedings.

[Rettinger et al., 2007] Rettinger, A., Nickles, M., and Tresp, V. (2007). Learning initial trust among interacting agents. In *CIA '07: Proceedings of the 11th international workshop on Cooperative Information Agents XI*, pages 313–327, Berlin, Heidelberg. Springer-Verlag.

[Rettinger et al., 2008] Rettinger, A., Nickles, M., and Tresp, V. (2008). A statistical relational model for trust learning. In *Proceeding of 7th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2008)*.

[Rettinger et al., 2009] Rettinger, A., Nickles, M., and Tresp, V. (2009). Statistical relational learning with formal ontologies. In *ECML PKDD '09: Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 286–301, Berlin, Heidelberg. Springer-Verlag.

[Rettinger et al., 2006] Rettinger, A., Zinkevich, M., and Bowling, M. (2006). Boosting expert ensembles for rapid concept recall. In *AAAI'06: Proceedings of the 21st national conference on Artificial intelligence*, pages 464–469. AAAI Press.

[Richardson and Domingos, 2006] Richardson, M. and Domingos, P. (2006). Markov logic networks. *Journal of Machine Learning Research*, 62(1-2):107–136.

[Ruohomaa and Kutvonen, 2005] Ruohomaa, S. and Kutvonen, L. (2005). Trust management survey. In [Herrmann et al., 2005], pages 77–92.

[Sabater and Sierra, 2001] Sabater, J. and Sierra, C. (2001). REGRET: reputation in gregarious societies. In Müller, J. P., Andre, E., Sen, S., and Frasson, C., editors, *Proceedings of the Fifth International Conference on Autonomous Agents*, pages 194–195, Montreal, Canada. ACM Press.

[Sarwar et al., 2000a] Sarwar, B., Karypis, G., Konstan, J., and Riedl, J. (2000a). Application of dimensionality reduction in recommender systems–a case study. In *Proc. ACM WebKDD Workshop 2000*.

[Sarwar et al., 2000b] Sarwar, B. M., Karypis, G., Konstan, J. A., and Riedl, J. (2000b). Analysis of recommender algorithms for e-commerce. In *Proc. ACM E-Commerce Conference*, pages 158–167. ACM.

[Sato et al., 2005] Sato, T., Kameya, Y., and Zhou, N.-F. (2005). Generative modeling with failure in prism. In *IJCAI'05: Proceedings of the 19th international joint conference on Artificial intelligence*, pages 847–852, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

[Sethuraman, 1994] Sethuraman, J. (1994). A constructive definition of dirichlet priors. *Statistica Sinica*, 4:639–650.

[Shafiei and Milios, 2006] Shafiei, M. M. and Milios, E. E. (2006). Latent Dirichlet co-clustering. In *ICDM '06: Proceedings of the Sixth International Conference on Data Mining*, pages 542–551, Washington, DC, USA. IEEE Computer Society.

[shan Chen, 1976] shan Chen, P. P. (1976). The entity-relationship model: Toward a unified view of data. *ACM Transactions on Database Systems*, 1:9–36.

[Shawe-Taylor and Cristianini, 2004] Shawe-Taylor, J. and Cristianini, N. (2004). *Kernel Methods for Pattern Analysis*. Cambridge University Press.

[Shi et al., 2005] Shi, J., v. Bochmann, G., and Adams, C. (2005). Dealing with recommendations in a statistical trust model. In *Proceedings of the Workshop on Trust in Agent Societies (held in conjunction with the 4th International Joint Conference on Autonomous Agents and Multi Agent Systems (AAMAS))*.

[Shoham et al., 2006] Shoham, Y., Powers, R., and Grenager, T. (2006). If multi-agent learning is the answer, what is the question? Technical Report 122247000000001156, UCLA Department of Economics.

[Sirin et al., 2007] Sirin, E., Parsia, B., Grau, B. C., Kalyanpur, A., and Katz, Y. (2007). Pellet: A practical owl-dl reasoner. *Web Semant.*, 5(2):51–53.

[Staab and Studer, 2009] Staab, S. and Studer, R. (2009). *Handbook on Ontologies*. International Handbooks on Information Systems. Springer, 2 edition.

[Studer et al., 1998] Studer, R., Benjamins, V., and Fensel, D. (1998). Knowledge engineering: principles and methods. *Data & Knowledge Engineering*, 25(1-2):161–197.

[Sun et al., 2005] Sun, L., Jiao, L., Wang, Y., Cheng, S., and Wang, W. (2005). An adaptive group-based reputation system in peer-to-peer networks. In Deng, X. and Ye, Y., editors, *WINE*, volume 3828 of *Lecture Notes in Computer Science*, pages 651–659. Springer.

[Syeda-Mahmood et al., 2006] Syeda-Mahmood, T., Goodwin, R., Akkiraju, R., and Ivan, A.-A. (2006). Matching and mapping for semantic web processes. In *Semantic Web Services, Processes and Applications*, pages 227–245.

[Takacs et al., 2007] Takacs, G., Pilaszy, I., Nemeth, B., and Tikk, D. (2007). On the gravity recommendation system. In *Proceedings of KDD Cup and Workshop 2007*.

[Taskar et al., 2002] Taskar, B., Abbeel, P., and Koller, D. (2002). Discriminative probabilistic models for relational data. In *Uncertainty in Artificial Intelligence (UAI)*.

[Teacy, 2006] Teacy, W. T. L. (2006). *Agent-Based Trust and Reputation in the Context of Inaccurate Information Sources*. PhD thesis, Electronics and Computer Science, University of Southampton.

[Tresp, 2006] Tresp, V. (2006). Dirichlet processes and nonparametric bayesian modelling. Tutorial at the Machine Learning Summer School 2006 in Canberra, Australia.

[Tresp et al., 2008] Tresp, V., Bundschus, M., Rettinger, A., and Huang, Y. (2008). *Uncertainty Reasoning for the Semantic Web I*, chapter Towards Machine Learning on the Semantic Web. Lecture Notes in AI, Springer.

[Tresp et al., 2009] Tresp, V., Huang, Y., Bundschus, M., and Rettinger, A. (2009). Materializing and querying learned knowledge. In *Proceedings of the First ESWC Workshop on Inductive Reasoning and Machine Learning on the Semantic Web (IRMLeS 2009)*.

[Tresp and Yu, 2002] Tresp, V. and Yu, K. (2002). Learning with dependencies between several response variables. In *Tutorial at ICML 2009*.

[Trochim, 2000] Trochim, W. (2000). *The Research Methods Knowledge Base*. Atomic Dog Publishing, second edition.

[Tso-Sutter et al., 2008] Tso-Sutter, K. H. L., Marinho, L. B., and Schmidt-Thieme, L. (2008). Tag-aware recommender systems by fusion of collaborative filtering algorithms. In *SAC '08: Proceedings of the 2008 ACM symposium on Applied computing*, pages 1995–1999, New York, NY, USA. ACM.

[Vasalou and Pitt, 2005] Vasalou, A. and Pitt, J. (2005). Reinventing forgiveness: A formal investigation of moral facilitation. In [Herrmann et al., 2005], pages 146–160.

[Wang and Sandholm, 2003] Wang, X. and Sandholm, T. (2003). Reinforcement learning to play an optimal nash equilibrium in team markov games. In S. Becker, S. T. and Obermayer, K., editors, *Advances in Neural Information Processing Systems 15*, pages 1571–1578. MIT Press, Cambridge, MA.

[Watts and Strogatz, 1998] Watts, D. and Strogatz, S. (1998). Collective dynamics of small-worldnetworks. *Nature*, 393(6684):440–442.

[Willis and Todorov, 2006] Willis, J. and Todorov, A. (2006). First impressions: Making up your mind after a 100-ms exposure to a face. *Psychological Science*, 17(7):592–598.

[Witkowski et al., 2001] Witkowski, M., Artikis, A., and Pitt, J. (2001). Experiments in building experiential trust in a society of objective-trust based agents. In *Proceedings of the workshop on Deception, Fraud, and Trust in Agent Societies held during the Autonomous Agents Conference*, pages 111–132, London, UK. Springer-Verlag.

[Witten and Frank, 2002] Witten, I. H. and Frank, E. (2002). Data mining: practical machine learning tools and techniques with java implementations. *SIGMOD Rec.*, 31(1):76–77.

[Wu and Sun, 2001] Wu, D. and Sun, Y. (2001). The emergence of trust in multi-agent bidding: A computational approach. In *HICSS '01: Proceedings of the 34th Annual Hawaii International Conference on System Sciences (HICSS-34)-Volume 1*, page 1041, Washington, DC, USA. IEEE Computer Society.

[Xu, 2007] Xu, Z. (2007). *Statistical Relational Learning with Nonparametric Bayesian Models*. PhD thesis, LMU München: Fakultät für Mathematik, Informatik und Statistik, Munich, Germany.

[Xu et al., 2009] Xu, Z., Tresp, V., Rettinger, A., and Kersting, K. (2009). Social network mining with nonparametric relational models. In H. Zhang, M. S., Giles, L., and Yen, J., editors, *Advances in Social Network Mining and Analysis*, LNCS. Springer.

[Xu et al., 2006] Xu, Z., Tresp, V., Yu, K., and Kriegel, H.-P. (2006). Infinite hidden relational models. In *Uncertainty in Artificial Intelligence (UAI)*.

[Xu et al., 2008] Xu, Z., Tresp, V., Yu, S., and Yu, K. (2008). Nonparametric relational learning for social network analysis. In *2nd ACM Workshop on Social Network Mining and Analysis (SNA-KDD 2008)*.

[Yoon et al., 2007] Yoon, S., Benini, L., and De Micheli, G. (2007). Co-clustering: A versatile tool for data analysis in biomedical informatics. *Information Technology in Biomedicine, IEEE Transactions on*, 11(4):493–494.

[Yu et al., 2006] Yu, K., Chu, W., Yu, S., Tresp, V., and Xu, Z. (2006). Stochastic relational models for discriminative link prediction. In *Advances in Neural Information Processing Systems 19*.

[Yu et al., 2005] Yu, S., Yu, K., and Tresp, V. (2005). Soft clustering on graphs. In *Advances in Neural Information Processing Systems 18*.

[Zanero, 2005] Zanero, S. (2005). Security and trust in the italian legal digital signature framework. In [Herrmann et al., 2005], pages 34–44.