TECHNISCHE UNIVERSITÄT MÜNCHEN

Lehrstuhl für Informatik XIX

Improving Semantic Search in the German Legal Domain with Word Embeddings

Jörg Landthaler

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: Prof. Dr. Stephan Günnemann

Prüfer der Dissertation:
1. Prof. Dr. Florian Matthes
2. Prof. Kevin D. Ashley, PhD, University of Pittsburgh

Die Dissertation wurde am 25.10.2019 bei der Technischen Universität München eingereicht und durch die Fakultät für Informatik am 02.03.2020 angenommen.

Zusammenfassung

Die juristische Recherche ist ein wesentlicher Bestandteil juristischer Arbeit. Große juristische Textkorpora werden von Verlagen gepflegt und verändern sich mit der Zeit. Durch die Digitalisierung und die immer größer werdenden juristischen Informationsmengen gewinnt das Juristische Information Retrieval an Bedeutung. Synonymie ist ein wichtiger Aspekt der juristischen Sprache und des Juristischen Information Retrieval. Semantische Suche, insbesondere Query Expansion, kann dazu verwendet werden Synonymie zu adressieren. Daher unterhalten Verlage große juristische Thesauri. Word Embeddings ist eine Technologie, die Worte mit reellwertigen, dichten Vektoren repräsentiert, die semantische Aspekte, im Speziellen Synonymie codieren. Dies führt zu der Frage, ob und wie Word Embeddings dazu genutzt werden können, Query Expansion für das juristische Information Retrieval zu verbessern.

Zuerst wird die Erweiterung deutscher juristischer Thesauri untersucht, in erster Linie die Erweiterung eines großen deutschsprachigen Steuerrechts-Thesaurus. Ein traditionelles und vier Word Embeddings basierte Verfahren werden qualitativ verglichen. Weiterhin werden die Word Embeddings Algorithmen GloVe, word2vec und FastText qualitativ und quantitativ verglichen. Zweitens wird eine implizite Form der Query Expansion untersucht, die auf dem WE-DF Vektorraum basiert. Der WE-DF Vektorraum stützt sich darauf, dass Word Embeddings Vektoren aufaddiert werden, um Textsegmente oder Dokumente zu codieren und benötigt keinen Thesaurus. Ein innovativer Anwendungsfall im deutschen Mietrecht wird untersucht. Anwälte wählen einen Textabschnitt in Mietverträgen aus und Kapitel aus Kommentaren werden für die natürlichsprachliche Suchanfrage vorgeschlagen (Textauswahl-Suche). Ein Ansatz, der Kapitel aus Kommentaren mit WE-DF codiert, ein Ansatz, der Kapitel aus Kommentaren als eine Menge von Sätzen, die mit WE-DF codiert werden, repräsentiert und das TF-IDF Vektorraum-Retrieval werden quantitativ verglichen. Die technischen Ansätze sowie die Suchmethoden Textauswahl-Suche und traditionelle Stichwortsuche werden mit einer Nutzerstudie evaluiert. Drittens wird ein eXplainable AI Ansatz, konkret eine Sensitivitätsanalyse, eingesetzt, um das WE-DF und das TF-IDF Vektorraum-Retrieval auf die Fähigkeit hin zu untersuchen, Kapitel aus Kommentaren zu repräsentieren. Erste empirische Studien und analytische Untersuchungen vergleichen die beiden Vektorraum-Retrieval Verfahren.

Die qualitative Evaluation von mehr als 2000 vorgeschlagenen Begriffen zeigt, dass die Signale, die durch Word Embeddings Algorithmen codiert werden, zu viel Rauschen enthalten, um eine vollständige Automatisierung zu ermöglichen. Dennoch können Word Embeddings Algorithmen dazu einsetzt werden, um Domänen-Experten eine signifikante Anzahl an neuen Synonymen vorzuschlagen. Bis zu 50% richtige Vorschläge innerhalb der ersten zehn Vorschläge pro Synonym-Gruppe können identifiziert werden. Dieses positive Ergebnis kann nicht aus der Evaluation einer Thesaurus-Rekonstruktion abgeleitet werden. Word Embeddings Algorithmen führen zu signifikant besseren Ergebnissen, als traditionelle Verfahren. Das WE-DF Vektorraum-Retrieval ermöglicht eine implizite Query Expansion. Der Ansatz, der Kapitel aus Kommentaren als eine Menge von Sätzen repräsentiert, die mittels WE-DF codiert werden, erzielt unter den ersten Suchergebnissen bessere Ergebnisse als das traditionelle TF-IDF Vektorraum-Retrieval. Das WE-DF Vektorraum-Retrieval ist besonders dazu geeignet kleine Text-Abschnitte zu codieren. Die Ergebnisse der Nutzerstudie deuten darauf hin, dass die Textauswahl-Suche als eine komplementäre Suchmethode zur traditionellen Stichwortsuche einsetzt werden kann.

Abstract

Legal research is an essential part of lawyers' daily work. Large textual legal corpora are maintained by publishers that change over time. Due to the digitization and increasing amounts of legal information, legal information retrieval is gaining in importance. Synonymy is an important aspect of legal language and legal information retrieval. Semantic search, in particular, query expansion, can be used to address synonymy. Thus, legal publishers maintain large legal thesauri, too. Word embeddings is a technology that represents words with real-valued, dense vectors that encode semantic aspects, and besides other aspects, particularly synonymy. This raises the question if and how word embeddings can be used to improve query expansion for German legal information retrieval.

First, the extension of legal thesauri is investigated, primarily, the extension of a large German tax law thesaurus. One traditional approach and four word embeddings based approaches are compared qualitatively. Furthermore, the contemporary word embeddings algorithms GloVe, word2vec and FastText are compared quantitatively and qualitatively. Second, an implicit form of query expansion using the WE-DF vector space model is investigated. The WE-DF vector space model accumulates word embeddings vectors to represent text segments or documents. Thus, the WE-DF vector space model does not require a thesaurus. An innovative use case in tenancy law is investigated. Lawyers select text segments in tenancy contracts and chapters from legal comments are recommended for the natural language search query (Selection Search). One approach that encodes legal comment chapters with the WE-DF text representation, one approach that represents legal comment chapters as a set of WE-DF encoded sentences and the TF-IDF vector space model are compared quantitatively. The approaches, as well as the search methods Selection Search and traditional keyword search, are evaluated qualitatively with a user study. Third, an eXplainable AI approach, in particular, a sensitivity analysis, is used to compare the WE-DF and the TF-IDF vector space models for their capability to represent legal comment chapters. The two vector space models are compared with first empirical experiments and analytically.

The qualitative evaluation of more than 2,000 candidate terms shows that contemporary word embeddings algorithms are too noisy to enable full automation but can be a useful tool to suggest a significant number of synonym candidates to domain experts. Up to 50% reasonable candidate terms among the first ten suggestions per synset can be identified. This positive result cannot be derived from the evaluation of standard thesaurus reconstruction tasks. Word embeddings algorithms significantly outperform traditional count-based distributional semantic models. The WE-DF vector space model does perform an implicit form of query expansion. The approach that represents legal comment chapters as a set of WE-DF encoded sentences performs better than the traditional TF-IDF vector space model in the top-ranked results. The WE-DF vector space model is best suited to encode small text segments. The results of the user study suggest that the Selection Search can be used as a complementary search method to traditional keyword search.

Acknowledgment

First and foremost, I am deeply grateful to my supervisor Prof. Dr. Florian Matthes for providing me this great opportunity, for the good collaboration, for guiding and supporting my research and this thesis, for supporting my personal growth, for helping me developing new skills and establishing a valuable network. I am also highly grateful for enabling and supporting the founding scholarship.

I would like to express my sincere gratitude to Prof. Kevin D. Ashley for reviewing this thesis, for the positive, constructive and valuable feedback and all his support. Additionally, I would like to thank Prof. Dr. Stephan Günnemann for the good collaboration and for accepting the Chair of the examination board.

I would like to thank all my colleagues from the sebis chair for their precious support, in particular (in alphabetical order), Gloria Bondel, Ahmed Elnaggar, Ulrich Gallersdörfer, Ingo Glaser, Dr. Matheus Hauder, Manoj Mahabaleshwar, Adrian Hernandez-Mendez, Patrick Holl, Dominik Huth, Jian Kong, Dr. Felix Michel, Sascha Nägele, Dr. Thomas Reschenhofer, Christof Tinnes, Elena Scepankova, Aline Schmid, Klym Shumaiev, Ömer Uludağ, Dr. Bernhard Waltl, Fatih Yilmaz and Dr. Marin Zec. Special thanks to those who have reviewed parts of this thesis. I appreciate this highly valuable feedback.

I would also like to thank all my students, in particular Saasha Nair, Christoph Erl, Markus Müller and Nicolas Thule, who helped me tremendously with my research, and Felix Jablonski for implementing the AddIn and the constructive collaboration.

I want to express my sincere gratitude to the industry partners Haufe Group and Datev eG that enabled this research as well as Manuel Reil from Alyne for the excellent collaboration. Special thanks to Hans Lecker for enabling the user study and his precious and continuous support. I am deeply grateful to Dr. Kristian Beckers for his guidance and help, especially in difficult times. Furthermore, I am deeply grateful to all participants of the user study who have spent a significant amount of their valuable time.

I would also like to thank my family, in particular my parents Edeltraud and Walter, for their love, patience and support at all times. Special thanks to my friend Marcus Grochowina who went with me together through good and hard times while writing this thesis.

This thesis is the result of many years of research and work. A lot of people helped me in reaching this point in my life and I would like to thank everyone who was involved in this effort directly or indirectly.

Garching bei München, 30.07.2020

Jörg Landthaler

Table of Contents

List of Figures

List of Tables

All truths are easy to understand once they are discovered; the point is to discover them.

Galileo Galilei

CHAPTER 1

Introduction

1.1. Motivation

Digitalization is a revolution that spreads into all aspects of life. The legal domain is no exception. For more than three decades, the Artificial Intelligence & Law (AI&Law) research community has been striving to support experts in the legal domain with computer science methods and tools. The most important tool in the legal domain is written language. A substantial part of AI&Law research uses Natural Language Processing (NLP) to support legal activities. The progressing digitization of textual data and an ever-increasing amount of textual legal data demand for more effective and more efficient tools to find relevant documents in large text corpora. Legal research is an important activity for lawyers. For example, Lastres (2015) found that law firm associates spend up to 15 hours per week on average on conducting legal research[1]. Consequently, a significant fraction of publications in AI&Law research addresses legal information retrieval (Van Opijnen and Santos (2017)).

A major trend in NLP-based AI&Law research, in general, but also in legal information retrieval, specifically, is to dig deeper into the complex semantics of legal documents. The word embeddings technology has attracted significant interest in NLP research in recent years. Words are represented with dense, real-valued vectors. The word embeddings vectors of a word embeddings model have intriguing properties. Semantic aspects are encoded in the word embeddings vectors, for example, synonymy. Semantic operations can be carried out with linear operations in the vector space of a word embeddings model. A more in-depth review of word embeddings is presented in Chapter 2.

Semantic search attempts to incorporate the meaning of language and user intentions into information retrieval. Legal information retrieval addresses aspects specific to legal research and legal data. This thesis explores if and how the semantic aspects incorporated in word embeddings models can be

[1]The results of this thesis confirm that the results of Lastres can be transferred to German lawyers.

used to improve semantic search for legal information retrieval. The German legal system has several specifics, including specifics of the German language. Thus, the research presented in this thesis seeks to conduct and improve semantic search for the German legal domain. Legal information retrieval, semantic search and specifics of the German legal domain are explained in more detail in Chapter 3.

1.2. Research Questions and Research Methods

The word embeddings technology encodes semantic aspects in real-valued, dense vectors that represent words. A major trend in legal information retrieval is to capture more as well as more complex semantics in legal documents. The question arises if the semantics encoded in word embeddings models can be used to improve German legal information retrieval and is reflected in the main research question:

> ***Main research question:*** Do word embeddings models capture semantic aspects that can be used to improve semantic search for German legal information retrieval?

The main research question is split into several derived research questions. As a first step to answer the main research question, it is necessary to identify contemporary word embeddings algorithms. For the identified word embeddings algorithms, it is necessary to understand the word embeddings algorithms, the accompanying hyper-parameters as well as the semantics encoded in the word embeddings models[2].

> ***Research question 1 (RQ1):*** What word embeddings algorithms exist and what semantic aspects do they encode?

Synonymy is a major aspect that is encoded in word embeddings models. Thus, word embeddings models are well suited to improve query expansion for legal information retrieval. A frequently used approach of query expansion is to use a thesaurus to expand search queries. Legal publishers often maintain (legal) thesauri for query expansion purposes. Textual corpora change over time and thesauri need to be updated. An important use case is the extension of existing thesauri rather than the reconstruction of an existing thesaurus. This gives rise to the question of how word embeddings models can be used to extend thesauri and how well such approaches perform.

> ***Research question 2 (RQ2):*** How can word embeddings be used to extend legal thesauri and how good are the results that can be obtained with such approaches?

The maintenance of a thesaurus is a challenging task. This leads to the question if query expansion for legal information retrieval could also be achieved without using a thesaurus. The mathematical operations that can be carried out in the word embeddings model's vector spaces perform semantic operations. This suggests that a vector space model that is based on the accumulation of word embeddings vectors can be used to conduct an automated form of query expansion. This type of vector space model is called WE-DF in this thesis.

> ***Research question 3 (RQ3):*** How could word embeddings be used to conduct query ex-

[2]Word embeddings models are the output of word embeddings algorithms.

pansion without maintaining a (legal) thesaurus and how good are the results that can be obtained with such approaches?

Keyword search is a traditional and widely used human-computer interaction method in legal information retrieval. The WE-DF vector space model that accumulates word embeddings vectors to represent documents also allows for a form of natural language search. Lawyers select text segments in documents as input to a legal information retrieval system (Selection Search). This constitutes an alternative human-computer interaction method. Thus, the questions arise if lawyers would like to use this alternative human-computer interaction method and how lawyers perceive the quality of the search results.

> ***Research question 4 (RQ4):*** How do lawyers judge natural language search that uses WE-DF vector space models in comparison to traditional keyword search?

The results that are obtained during the answering of the previous research question suggest that lawyers rate the results of WE-DF and TF-IDF vector space models similarly[3], while the WE-DF vector space model performs worse than the TF-IDF vector space model according to the quantitative evaluation. This leads to the question of how similar or different the two vector space models are and how such a question can be answered.

> ***Research question 5 (RQ5):*** How similar or different are WE-DF vector space models in comparison to traditional TF-IDF vector space models?

AI&Law research is interdisciplinary research at the interface of computer science and law. The research conducted in this thesis continues with a long tradition of legal information retrieval research. For the most part, the investigations in this thesis can be categorized as data science. The largest fraction of research described in this thesis uses empirical methods. Empirical methods are frequently used in AI&Law research. Both quantitative and qualitative evaluations are used to answer the research questions. For the user study, a prototypical implementation is required. Thus, parts of this thesis also require software engineering methods and tools.

1.3. Thesis Organization and Contributions

The thesis is divided into in seven chapters. Figure 1.1 illustrates the thesis organization, contributions and associated publications.

Chapter 2 provides an overview of the word embeddings technology in several aspects. Contemporary word embeddings algorithms are identified. The roots of the word embeddings technology are reviewed. The hyper-parameters of the word embeddings algorithms are listed and existing knowledge on the hyper-parameters is summarized. Further, the literature is consulted for the semantic aspects that are encoded in word embeddings models. Chapter 2 answers the first research question.

Next, in Chapter 3, related work is reviewed in the areas and topics of AI&Law, legal information retrieval, thesaurus extension, text representations, vector space models and eXplainable AI (XAI).

[3] Here, the results of the CHAPTER Approach are meant and not the results of the SENT Approach.

Chapters (II-VII)	Word Embeddings Review	Foundations and Related Work	Thesaurus Extension	Semantic Text Matching	eXplainable Semantic Text Matching	Conclusion
Research Questions	RQ1	RQ3	RQ2	RQ3, RQ4	RQ5	
Research Contributions	Word embeddings review		Thesaurus extension approaches Use case challenges Quantitative evaluation Qualitative evaluation	Semantic Text Matching WE-DF based approaches Use case challenges Evaluation set construction process Quantitative evaluation Legal information retrieval tools User study	XSTM Approach Single match case study Aggregated matches case study Analytical comparison of TF-IDF and WE-DF vector space models	Summary Limitations Outlook
Publications			Landthaler et al. (2017)	Landthaler et al. (2016) Landthaler et al. (2018b) Landthaler et al. (2019)	Landthaler et al. (2018a)	

Figure 1.1.: Thesis overview and contributions.

Required foundations on the subject of evaluating information retrieval technologies, specifics of legal data and the German legal system, as well as additionally used algorithms, are introduced.

Chapter 4 is dedicated to answering the second research question. The use case of thesaurus extension is introduced. Several approaches to extend thesauri are identified. The approaches encompass one traditional Distributional Semantic Model (DSM), three approaches that directly leverage word embeddings models and one approach that uses label propagation that also uses word embeddings models. The hyper-parameters of the three contemporary word embeddings algorithms word2vec, FastText and GloVe are explored using the Ranking Position Score (RP-Score) evaluation measure. The approaches to thesaurus extension are compared qualitatively and where appropriate, also quantitatively. An analysis of sample results explains the results of the quantitative and qualitative evaluations more in-depth. Moreover, challenges specific to German legal information retrieval and German tax law thesaurus extension that arose while conducting the research are summarized.

Chapter 5 covers research questions three and four. First, different problem classes where the WE-DF vector space model can be used are identified. The abstract problem of Semantic Text Matching is introduced. Semantic Text Matching problems are identified as a promising problem class for the application of the WE-DF vector space model. Different approaches to solve Semantic Text Matching problems are identified. Selected approaches are compared quantitatively. A method to construct an adequate evaluation dataset is reported. Further, the different approaches to solve Semantic Text Matching problems and the two search methods that constitute different human-computer interaction methods are implemented and evaluated in a user study with 25 lawyers. Again, use case challenges that arose during conducting the research are summarized.

In Chapter 6, the eXplainable Semantic Text Matching (XSTM) Approach, a variant of an existing eXplainable AI approach, is introduced and used to compare the WE-DF and TF-IDF vector space models empirically. It is investigated how the results of an XSTM analysis (significance scores) can be used to gain more in-depth insights on the text representations, text similarity measures and vector space models. Furthermore, the two vector space models are compared analytically. Chapter 6 addresses research question five.

Chapter 7 concludes the thesis with a summary of the results of the previous chapters and the research questions are answered. Chapter 7 includes limitations and an outlook on future work, too.

You shall know a word by the company it keeps.

John Rupert Firth

CHAPTER 2

Word Embeddings Review

The word embeddings technology represents words of natural language with dense, real-valued vectors that possess intriguing semantic properties. Word embeddings algorithms attracted much attention in NLP research in recent years. A starting point of the increasing interest is a series of publications by Tomáš Mikolov. Figure 2.1 shows the increasing number of publications over the last years that are listed to reference Mikolov et al. (2013a) on Google Scholar.

Historically, word embeddings algorithms draw from two research areas: distributional semantics and distributed representations. Distributional semantics and distributed representations have been independent research areas. Later on, it was shown that the two research areas are intimately related through word embeddings algorithms. The goal of this chapter is to recapitulate the historical developments that lead to word embeddings, to introduce selected contemporary word embeddings algorithms and to summarize properties of word embeddings models known today. Parts of this review have also been summarized in different blog posts[1]. Figure 2.2 shows an overview of key publications described in the following subsections.

2.1. Distributional Semantics

Distributional semantics research is part of the field of linguistic research. The goal of distributional semantics research is to derive meaning from a statistical analysis of corpora. The common basis for distributional semantics is the distributional hypothesis. The distributional hypothesis states that terms that occur in similar contexts tend to have a similar meaning. Sahlgren (2008) and Sahlgren

[1] http://blog.christianperone.com/2018/05/nlp-word-representations-and-the-wittgenstein-philosophy-of-language/, https://www.gavagai.se/blog/2015/09/30/a-brief-history-of-word-embeddings/, http://blog.aylien.com/overview-word-embeddings-history-word2vec-cbow-glove/, https://www.analyticsvidhya.com/blog/2017/06/word-embeddings-count-word2veec/, last accessed April 27, 2019

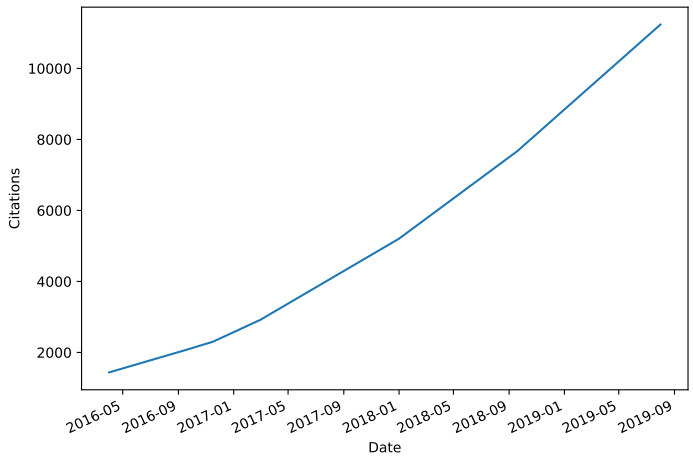

Figure 2.1.: Citation counts of Mikolov et al. (2013a) according to Google Scholar.

(2006) note that the distributional hypothesis is described by different formulations, but they all share the same basic concept. Many publications cite Harris (1954) as the original source of the distributional hypothesis. Sahlgren (2008) and Sahlgren (2006) points out that also Wittgenstein (1953) and Firth (1957) are also frequently attributed as early sources for the distributional hypothesis.

Rubenstein and Goodenough (1965) were among the first to present empirical evidence that supports the distributional hypothesis. First, the authors chose a selection of several word pairs and let humans rate the synonymy of the word pairs. Second, the authors collected co-occurrence statistics of the context of words from a corpus. The authors found a correlation among word co-occurrence statistics and perceived synonymy of word pairs. Different types of contexts were investigated: fixed windows, sentences but also more complex context types, for example, based on the grammatical structure.

The distributional semantics theory has been the basis for a plethora of distributional semantic models (DSMs), see Baroni and Lenci (2010). Literature reviews and selected examples for DSMs include Grefenstette (1994), Hyperspace Analogue to Language (HAL) Lund (1995) and Lund and Burgess (1996), Word-Space Model Sahlgren (2006) and the almost equally named Word Space model by Schütze (1993). Extensive hyper-parameter studies for DSMs have been presented in Bullinaria and Levy (2007), Bullinaria and Levy (2012) and Turney and Pantel (2010). Early on, distributional semantics have been used to exploit the distributional hypothesis, for example, to detect semantically similar words. Later, Baroni et al. (2014) call this type of models *count-based* DSMs in order to differentiate such models from *predictive* DSMs. Today, predictive DMSs constitute word embeddings.

Two selected count-based DMSs are worth a few more words. Schütze explicitly notes the connection between distributional semantics and distributed representations. One of the two models is presented in Schütze (1992), where distributed representations vectors for words are obtained by the application of a Singular-Value Decomposition (SVD) on the token-token co-occurrence matrix. The other model is Schütze (1993)'s Word Space. Word Space uses sub-word-level information, in particular, character four-grams. An approach that will be later on picked up by the FastText algorithm (Bojanowski et al.

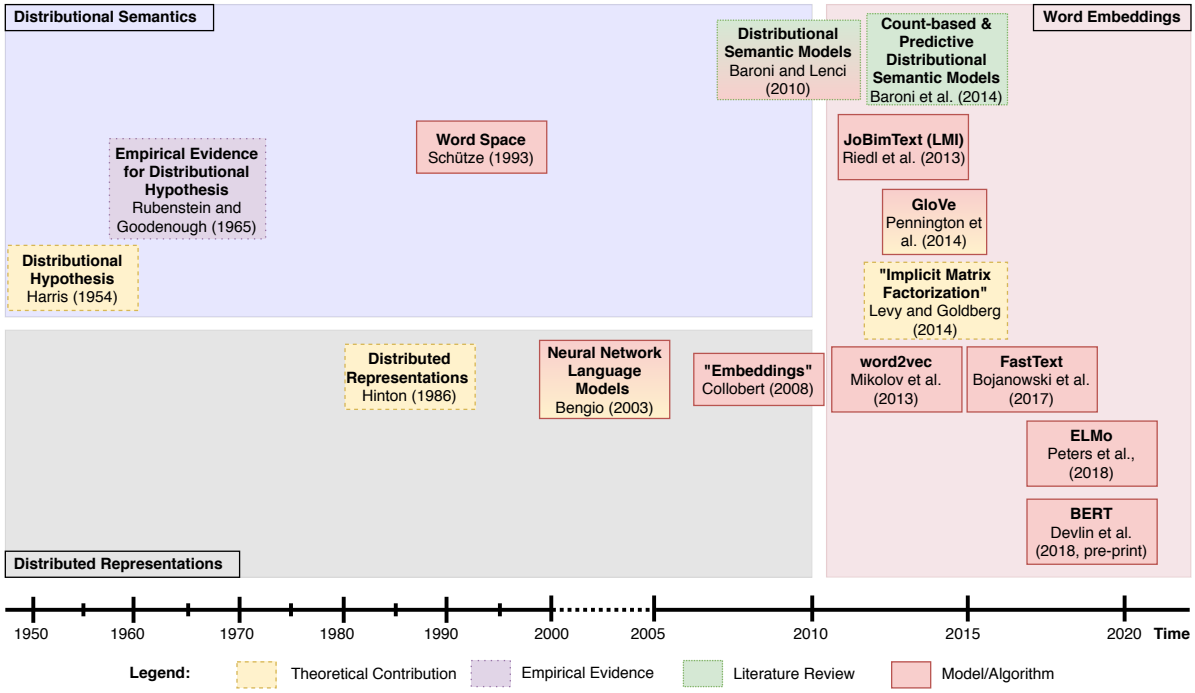

Figure 2.2.: Historical milestones in the development of word embeddings algorithms, distributional semantics and distributed representations.

(2017)), see Section 2.9. The Word Space model calculates character four-gram co-occurrence statistics and stores the information in a matrix. Afterward, a SVD is conducted on the matrix to calculate distributed character n-gram representations. A word is represented as the weighted sum of its composing character four-gram vectors. The word representations serve as feature vectors for tokens in other NLP tasks such as the detection of similar words.

2.2. Distributed Representations

In the 1980s, Geoffrey Hinton published a series of publications on distributed representations: Hinton et al. (1986a), Hinton et al. (1986b) and Hinton et al. (1986c). Distributed representations laid the foundation for artificial neural networks in the form they are used and understood nowadays. According to Hinton et al. (1986a), the basic idea of distributed representations is that "each entity is represented by a pattern of activity distributed over many computing elements, and each computing element is involved in representing many different entities". Bengio (2008) describes this basic idea more technically as "A distributed representation of a symbol is a tuple (or vector) of features which characterize the meaning of the symbol, and are not mutually exclusive". From a data structure perspective, a distributed representation is a set of "microfeatures" that represent an entity (Hinton et al. (1986a)). Today, it is common to collect such a set of microfeatures in a vector. Word embeddings vectors are such a type of distributed representation. A natural language example is used to illustrate the idea of distributed representations: a group of words activates units representing the meaning of the word, cf. Hinton

et al. (1986a), Figure 6b. The error backpropagation algorithm could already be used to train such networks, see Rumelhart et al. (1988) and Hinton et al. (1986c). However, there was no automated way to learn the semantics of words, yet. A mathematical derivation of artificial neural networks, error backpropagation and typical hyper-parameter configurations can be found in Landthaler (2011).

Neural network language models (NNLM, Bengio et al. (2003) and Bengio (2008)) model language statistically through artificial neural networks. The standard goal is to predict a word from preceding words or n-grams. These models arose naturally from the inherent nature of artificial neural networks to generalize from given samples, i.e., the functionality of artificial neural networks to predict. Bengio et al. (2003) use distributed representations of words to construct an NNLM. According to Bengio et al. (2003), "a very large set of possible meanings can be represented compactly".

2.3. Word Embeddings

Later on, the primary goal for developing unsupervised NNLM changed from creating models for word prediction to explicitly calculating high quality distributed word representations. Examples for word representations oriented models include Miller et al. (2004), Collobert and Weston (2008) and Mnih and Hinton (2009). Collobert and Weston (2008) is among the first to use the term *embeddings* to denote distributed representations of words because word embeddings vectors are embedded in a lower-dimensional vector space. At the beginning of his research, Mikolov investigated NNLMs that use Recurrent Neural Network (RNN), see Mikolov (2012), Mikolov et al. (2010), Mikolov et al. (2011) and Mikolov and Zweig (2012). Afterward, the focus of Mikolov shifted to investigating shallow neural networks and presented the word2vec algorithm in a series of publications: Mikolov et al. (2013a), Mikolov et al. (2013c) and Mikolov et al. (2013b). The presentation of the word2vec algorithm can be seen as a foundation stone for a strong interest in word embeddings in academia in recent years. Distributional semantics and distributed representations obtained through neural networks are closely related. Levy and Goldberg (2014) showed that the word2vec algorithm implicitly factorizes a special token-token co-occurrence matrix. I.e., word embeddings algorithms and count-based DSMs are intimately related. More details on the word2vec algorithm are discussed in the next section.

A follow-up algorithm of the word2vec algorithm is the FastText algorithm presented by Bojanowski et al. (2017) that picks up the idea of Schütze's Word Space. Another follow-up algorithm is the GloVe algorithm. The GloVe algorithm can be seen as one of the latest representatives of count-based models. However, in a large-scale study, Baroni et al. (2014) empirically found that predictive DSMs are superior to count-based DMSs in many cases. Most recent research attempts to shift from shallow neural networks again to deep learning technologies. ELMo (Peters et al. (2017)) and BERT (Devlin et al. (2018)), are examples of this current trend in word embeddings research.

Word embeddings are related to topic models like Latent Semantic Indexing (LSI) (Hofmann (1999)) and Latent Dirichlet Allocation (LDA) (Blei et al. (2003)). Topic models can be used to calculate distributed representations of words. However, while topic models are probabilistic models and therefore can be considered predictive, they primarily model topicality and operate on token-document statistics. Token-document statistics means that one part of the statistical model are token-specific statistics and the other part are document-specific statistics. An example of token-token statistics is the token co-occurrence matrix. An example for token-document statistics is the TF-IDF vector space model.

TF-IDF combines the global token occurrence frequency with the Inverse document frequency (IDF). In contrast to topic models, word embeddings algorithms operate on token-token statistics. Thus, topic models work on different data than current word embeddings algorithms that work solely on token-token co-occurrence. Topic models are also closely related to vector space models, see Section 3.3.

The concept of calculating word embeddings vectors as dense representation vectors for words has been generalized to other types of discrete entities, for example, graphs. Grover and Leskovec (2016)'s node2vec algorithm paved the way for much successful research in graph mining using graph embeddings.

2.4. Properties of Distributional Semantic Models

Sahlgren (2008) and Sahlgren (2006) elaborate on the history of the distributional hypothesis and, more importantly, on the effectiveness of DSMs. Sahlgren's reflection is helpful to deeper understand the potential of current word embeddings algorithms, too. Sahlgren reissues Saussure's analogy to chess, cf. Saussure (1916) and Saussure (1983). Chess can be described as a system with defined tokens and rules assigned to the tokens. For example, kings are allowed to move one field in any direction per draw while a rook can move in any direction straightforward for an arbitrary number of fields. The so-called "functional differences" among tokens can be used to identify the tokens. Similarly, natural language can be described as a system. DSMs statistically analyze the context/co-occurrence of words to infer the semantics of words. In the picture of the chess game, this translates to analyzing the moves of (many) draws to infer the rules of chess. The implications for DSMs are important. An existing corpus can be analyzed by a DSM, however, a DSM can only pick up the semantics inherently encoded by the use of the language in that particular corpus. Moreover, a DSM (without explicit model extension) cannot leverage external knowledge on semantics that humans will use in understanding a particular corpus. External knowledge can be, for example, other text corpora, training or life-experience.

Sahlgren further reminds of Saussure's work on language in Sahlgren (2008) and Sahlgren (2006). In essence, the meaning of words emerges from the interplay with other words. From a slightly different point of view, the differences in the semantics of words arise from their usage. This even might apply for subtle differences, for example, "truck" and "motor vehicle" could be considered synonyms, but a "motor vehicle" could also be considered a hyperonym of "truck". Sahlgren picks up on Saussure's functional differentiation of *syntagmatic* and *paradigmatic* relations among language entities. Syntagmatic relations describe language entities that occur in the context of each other, for example, the words "the" and "car". In contrast to that, paradigmatic language entities are used interchangeably, for example, the words "car" and "automobile". Contemporary word embeddings algorithms are trained on syntagmatic relations, but mostly paradigmatic relations are exploited later on. Hence, contemporary word embeddings algorithms seem to encode both types of relations.

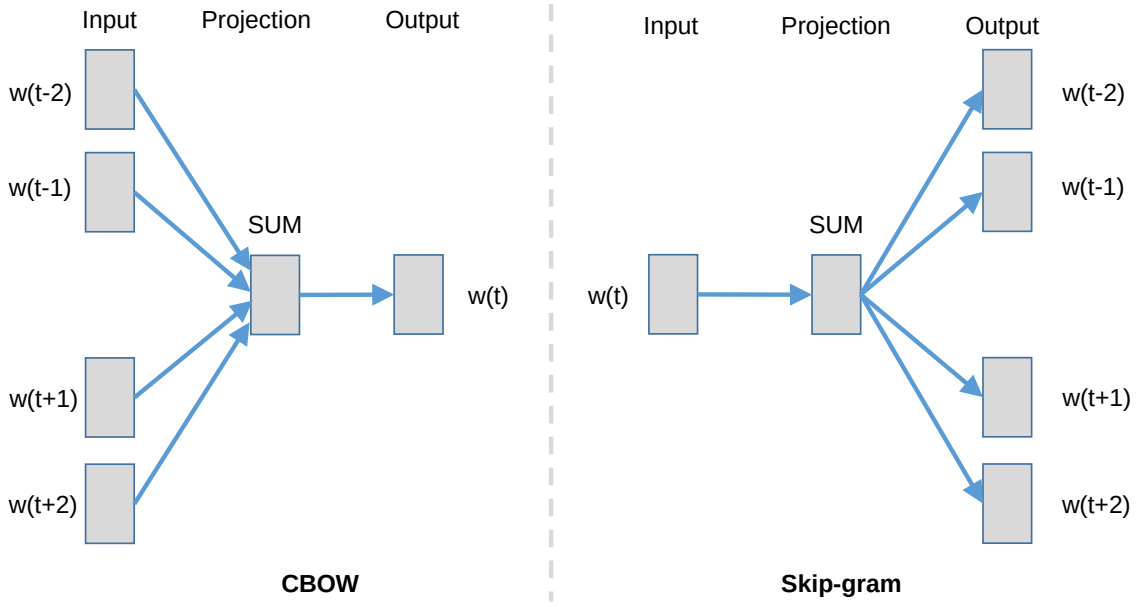

Figure 2.3.: Model architectures of the word2vec algorithm (Mikolov et al. (2013a)).

2.5. word2vec Algorithm

Starting in 2013, Tomáš Mikolov published a series of publications on an algorithm that efficiently calculates word embeddings models. The word2vec toolkit comprises the algorithm as software and is the eponym for the name of the algorithm. Mikolov's publications received much attention in AI, NLP and applied NLP research areas. The main reasons are the efficient calculation that enables the training of word embeddings models on large corpora and the intriguing properties of word embeddings models. The primary goal of this section is not to introduce all details on the word2vec algorithm, especially not the mathematical details, but to provide an easy to understand intuition of the algorithm. A mathematically more detailed introduction to the word2vec algorithm is presented in Goldberg and Levy (2014).

The word2vec algorithm trains a word embeddings model on a given training corpus. The basic idea of the word2vec algorithm is to train a shallow artificial neural network. Shallow artificial neural networks use one or a few hidden layers in contrast to deep learning where artificial neural networks with many hidden layers are used. Two different model architectures have been proposed. In Figure 2.3 the model architectures Continuous Bags of Words (CBOW), left and Skip-gram (SG), right are illustrated. Figure 2.3 has been adapted from Mikolov et al. (2013a). The CBOW model architecture attempts to predict a pivot word $w(t)$ from its context. The complementary model architecture, Skip-gram, predicts the context of a given pivot word $w(t)$. For both models, samples are obtained by a sliding window approach, i.e., a window of fixed size that is shifted sequentially over a training corpus. The artificial neural networks are trained with error backpropagation and stochastic gradient descent. The weight matrices[2] of the artificial neural network comprise the trained word embeddings model

[2] A recommended resource that explains how neural network weight matrices are transformed into word embeddings vectors in greater detail is https://iksinc.online/tag/continuous-bag-of-words-cbow/, last accessed April 24, 2019

after training. Remember the standard goal of NNLMs to predict words from preceding words. It is possible to use a word2vec word embeddings model to predict subsequent words to a given sequence of words. According to Baroni et al. (2014)'s classification, the word2vec algorithm is a predictive DSM. However, the main goal of the word2vec algorithm is to calculate word embeddings models, i.e., an inference step is not necessary.

When existing text is used as labeled data, large amounts of labeled data are available. Thus, a supervised artificial neural network can be used in an unsupervised fashion (weak supervision). Training artificial neural networks with positive examples only will result in a highly imbalanced model. The word2vec toolkit offers a solution called negative sampling. Randomly selected pairs of tokens from the corpus are used as negative samples in addition to the positive samples of token-token co-occurrence given by the windows that slide over the corpus. An early version of the word2vec algorithm uses a hierarchical softmax approach (an efficient approximation of the softmax function) rather than the negative sampling approach. Mikolov et al. (2013c), as well as succeeding studies, showed that the negative sampling approach is superior to hierarchical softmax approach in terms of performance and quality.

Another successful addition of the word2vec algorithm is the down-sampling of tokens that occur very frequently in the training corpus. Tokens that occur very frequently are usually stopwords like articles and disturb the training of less frequent tokens because they co-occur with a large fraction of terms. The word2vec algorithm uses a dynamic window size approach Mikolov et al. (2013c) and Goldberg and Levy (2014), i.e., the window size is chosen dynamically during training. In the word2vec algorithm, the effective window size is sampled from a uniform distribution from $[1; WindowSize]$. This results in giving higher weight to tokens that are closer to the pivot token. Unlike previous NNLMs that predict future words from past words, the word2vec algorithm uses a symmetric window, i.e., the word2vec algorithm incorporates past and future of a word in a given text sequence. Given the goal of producing high-quality word embeddings models, rather than predicting words from preceding words, this is a useful modification. Pennington et al. (2014) compares asymmetric and symmetric windows and showed that symmetric windows are superior to asymmetric windows.

2.6. Properties of Word Embeddings Models (word2vec)

Word embeddings models have several intriguing properties. The semantic similarity of word embeddings vectors in word embeddings model's vector space is usually calculated with cosine similarity. Word embeddings vectors of related words tend to have a smaller angle than word embeddings vectors of unrelated terms, i.e., the effects of DSMs are encoded in word embeddings models. This applies mostly to synonym relations. In addition to that, Mikolov et al. (2013d) noticed that linear vector operations, like addition and subtraction correspond to semantic relationships. For example, the prominent gender relationship example is that vec("King") - vec("Man") + vec("Woman") yields a vector that is close to vec("Queen"). Here, vec() denotes the word embeddings vector of the token given in parentheses[3]. While the word embeddings vectors encode such semantic relationships among tokens, they also encode syntactic relationships. For example, singular/plural relationships: vec("cars") - vec("car")

[3] According to Mikolov et al. (2013a) word embeddings vectors have been normalized to unit length before carrying out the linear translation operations

+ vec("apple") yields a vector close to vec("apples"). Other syntactic relationship types are encoded in word embeddings models, too.

The linear translations properties have been unaware of in the distributional semantics research before. Mikolov argues that the linear properties present in word embeddings model's vector space arise from the fact that word embeddings vector's "values are related logarithmically to the probabilities computed by the output layer" (Mikolov et al. (2013c)). Word embeddings vectors can be seen as a representation of the distribution of the context of a token. The frequency of the context tokens is modeled by the softmax function that is an exponential function. Because a product in log space translates to an additive, linear function in non-log-space[4], the product of two context distributions translates to additive, linear operations in word embeddings model's vector space.

Mikolov et al. (2013c) notes that the linear translation properties of word embeddings models are also present in word embeddings models trained with non-linear algorithms such as RNN. However, in contrast to RNNs, the simplicity of the word2vec algorithm makes it feasible to calculate word embeddings models from very large text corpora.

2.7. Word Embeddings and Distributional Semantic Models

To discuss the properties of word embeddings models in more depth, the introduction of further concepts of DSMs is required. Mutual Information (MI) is a measure of the mutual dependence of two random variables and is well-known in information theory. Point-wise Mutual Information (PMI) is a discrete variant of MI. PMI is often used as a statistical model to describe the association among words (Church and Hanks (1990)). In the context of DSMs, PMI is typically used to model word co-occurrences. Positive Point-wise Mutual Information (P-PMI) presented by Niwa and Nitta (1994) is a computationally more efficient variation of PMI. Negative entries in the PMI matrix are replaced with zeros so that only positive and zero values are retained. In a large-scale study, Bullinaria and Levy (2007) and Bullinaria and Levy (2012) empirically showed that P-PMI performs best in comparison to PMI and other token co-occurrence measures for count-based DSMs on several down-stream NLP tasks.

Levy and Goldberg (2014) showed that the word2vec algorithm is intimately related to count-based DSMs. The word embeddings models obtained by the word2vec algorithm's Skip-gram model architecture are the outcome of an iterative approximation of a process that implicitly factorizes a PMI matrix that is shifted by an additive constant.

The SVD factorizes a matrix M into matrices of singular values and eigenvectors: $M = U\Sigma V$. A SVD is often used to carry out a Principal Component Analysis (PCA). The goal of a PCA is to conduct a dimensionality reduction on a matrix. The PCA represents the data in terms of another basis of the vector space. The transformation of the (orthonormal) basis of the vector space is conducted in a way so that the data encoded by the different dimensions (variables) in the vector space is (maximally) linearly uncorrelated among the dimensions. The new basis of the vector space is derived from the SVD of the matrix. The dimensions (variables) that belong to the largest singular values (principal components) are retained, while the manually chosen number of least important dimensions are discarded. The

[4] $log(a * b) = log(a) + log(b)$

word2vec algorithm's Skip-gram model architecture iteratively approximates a SVD of the PMI matrix. However, Österlund et al. (2015) empirically showed that for word representation vectors obtained from DSMs, the opposite is most effective and that the principal components of the largest singular values should be discarded.

Another interesting property of word representation vectors obtained from count-based DSMs is that empirical investigations by Schütze (1992) suggest that most or all dimensions contribute (more or less) equally to the cosine similarity of two word representation vectors. I.e., individual dimensions in word representation vectors are probably not tied to specific human-understandable features. Schütze (1992) further notes that vectors obtained from DSMs lead to similar results in terms of quality for both, raw count vectors as well as transformed word representation vectors where the transformation is a dimensionality reduction with PCA. The results of Schütze have been obtained on a word sense disambiguation task.

There is no consensus on the interpretability of the dimensions of word embeddings models. Turian et al. (2010) argues that "each dimension of word embeddings model's vector space represents a latent feature of the word, hopefully capturing useful syntactic and semantic properties", while Jameel and Schockaert (2016) argue that the dimensions are "essentially meaningless". Other approaches attempt to clamp dimensions of word embeddings models by design to semantic aspects. For example, Qureshi and Greene (2018) link the dimensions of word embeddings models to Wikipedia topics.

2.8. word2vec Toolkit

The word2vec algorithm is available as an open-source toolkit, see Appendix A, and comes with several hyper-parameters that affect the quality of the resulting word embeddings models, default values are denoted in parentheses:

- **Model Architecture (CBOW):** The implementation of the word2vec algorithm ships with two basic model architectures: Continuous bag-of-words (CBOW) and Skip-gram; CBOW predicts a pivot word from its context while the Skip-gram model architecture predicts the context of a pivot word. On the one hand, Mikolov et al. (2013a) reports that CBOW word embeddings models are trained significantly faster and perform slightly better on syntactic similarity tasks. On the other hand, Mikolov states that Skip-gram word embeddings models perform much better on semantic similarity tasks.
- **Min-Count (5):** The Min-Count hyper-parameter defines the minimum occurrence frequency of tokens in the training corpus to be considered. Tokens that occur less than the Min-Count many times are excluded from the training process. As a consequence, word embeddings vectors are only returned for tokens that occur frequently enough. Setting the Min-Count hyper-parameter to 1 retains all tokens.
- **Iterations (5)[5]:** The Iterations hyper-parameter controls the number of training iterations over the training corpus for the gradient descent algorithm. One iteration consumes once all (positive) training samples from the training corpus. For smaller corpora, a larger number of iterations is recommended to mitigate lower numbers of training samples.

[5]Note that the Iterations hyper-parameter is introduced in the word2vec toolkit version 0.1c.

- **Vector Size (100):** The Vector Size is a manually chosen hyper-parameter that defines the number of dimensions of the resulting word embeddings vectors. The vector size is subject to a trade-off between computational efficiency, the amount of information that can be encoded in the vectors (information theory entropy) ,and subsequently, the performance that can be achieved on down-stream tasks. A finite training corpus has a limited amount of information that can be exploited. Mikolov et al. (2013a) reports that there exists a maximum size of the vectors so that additional dimensions will lead to worse results. Pennington et al. (2014) reports that the quality of word embeddings models starts to decrease for vector sizes larger than 300.
- **Window Size (5):** The Window Size hyper-parameter controls the size of the context window, i.e., the maximal number of tokens taken into account for the training step for one pivot token. The context window is symmetric, i.e., as many tokens as specified by the Window Size hyper-parameter before and after the pivot token are included. Remember that the word2vec algorithm uses a dynamic window size approach, i.e., the window size is sampled from a uniform distribution: $[1; WindowSize]$. Goldberg (2016) reports that larger windows result in more topical similarities, while smaller windows result in more syntactic similarities.
- **Negative Samples (5):** The Negative Samples hyper-parameter defines the number of negative examples used for one positive example during the stochastic gradient descent training. Mikolov et al. (2013c) reports that useful ranges for small datasets are values in the range $[5; 20]$ while for larger datasets values in the range $[2; 5]$ might suffice.
- **Sample Threshold (0.001):** Frequently occurring words in the training corpus are randomly down-sampled in the word2vec algorithm. The Sample Threshold hyper-parameter controls the probability that a frequently occurring word will be excluded from training. The documentation of the command-line tool of the word2vec toolkit recommends a useful range of values of $[0; 0.00001]$. However, the default value is 0.001. Thus, at least a range of $[0; 0.001]$ should be considered.
- **Alpha (0.025 for Skip-gram and 0.05 for CBOW):** The Alpha hyper-parameter defines the learning rate of the gradient descent algorithm. The Alpha hyper-parameter weighs the update to the weight matrices of gradient descent steps. In general, for training artificial neural networks with gradient descent, a too-large learning rate will prevent convergence of the gradient descent algorithm. A too-small learning rate will result in very slow convergence and requires a larger number of iterations to converge.
- **Hierarchical Softmax (False):** The Hierarchical Softmax approach can be used as an alternative to the negative sampling approach. Due to the increased performance and better-quality results of negative sampling, the Hierarchical Softmax option is obsolete.

The resulting word embeddings models are stored in either a binary or text-file format. In the first line of the text-file, the vocabulary size and the vector size are listed. The remainder of the text-file contains the string and the vectors as white-space separated values.

2.9. FastText Algorithm and Toolkit

The FastText algorithm is presented in a series of publications: Bojanowski et al. (2017), Joulin et al. (2016) and Joulin et al. (2017). The FastText algorithm picks up on the idea of the Word Space model presented by Schütze (1993). The Word Space model splits tokens into character four-grams. The idea of the FastText algorithm is to calculate character n-gram embeddings using the word2vec algorithm. Afterward, a word embeddings vector for a token is obtained by accumulating all character n-gram embeddings that are part of the token. To some degree, the character n-grams mimic syllables. The authors argue that word morphology can be leveraged by this approach. In contrast to the Word Space model that splits tokens into character four-grams, the FastText algorithm splits tokens into character n-grams where the range of the gram sizes is defined as a hyper-parameter. An example configuration could be to set the minimum and maximum value of "n" to two and three. For this configuration, the token *vehicle* would be split into a set of ve, eh, hi, ic, cl, le, veh, ehi, hic, icl, cle character two- and three-grams.

In Bojanowski et al. (2017), the authors argue that the out-of-vocabulary problem and infrequent tokens problem can be mitigated to some degree by this approach because the composing character n-grams occur much more frequently and word embeddings vectors for tokens can be calculated even if the token does not occur in the training corpus.

The FastText algorithm is implemented and available as an open-source toolkit in the gensim library, see Appendix A. The algorithm comes with all hyper-parameters of the word2vec algorithm except the Hierarchical Softmax option. Additionally, the FastText algorithm comes with the following hyper-parameters that affect the quality of the resulting word embeddings models:

- **MinN (3), MaxN (6):** The range of sizes of n-gram characters words are split into is defined by the minimum and maximum N.
- **LrUpdateRate (100):** The LrUpdateRate is a decay factor that modifies the learning rate. The idea is that the learning rate allows for large updates to the gradient descent in the beginning to speed up training and to fine-tune results with smaller updates at the end of the training procedure.

The default hyper-parameter values are indicated in parentheses. The default values for the shared hyper-parameters with the word2vec toolkit are equal except that the Alpha hyper-parameter is set to 0.05 for both model architectures (CBOW and Skip-gram) and the default Sampling Threshold hyper-parameter is set to 0.0001. The FastText algorithm shares the file-formats for storing word embeddings models with the word2vec toolkit.

2.10. GloVe Algorithm and Toolkit

The GloVe algorithm (Pennington et al. (2014)) is another algorithm to calculate word representation vectors from a training corpus. In contrast to the word2vec algorithm, a representative of predictive DSMs, it can be categorized as one of the latest representatives of count-based DSMs. The resulting models might be better-called word representations, but the term word embeddings is commonly used. The GloVe algorithm calculates a global token co-occurrence matrix and poses the calculation of word

representations as a least-squares problem. To make the calculation computationally feasible, the least-squares problem is solved with a sampling method (AdaGrad Duchi et al. (2011)), i.e., an iterative approximation algorithm.

The starting points for the GloVe algorithm are two probability distributions. One probability distribution models the ratio of token co-occurrences. The second probability distribution models the relation among word representation vectors. Least-squares regression is used to minimize the distance among the two probability distributions according to a distance measure for probability distributions (Kullback-Leibler divergence). The word representation vectors can be seen as slack variables that are calculated during the application of the least-squares regression. While the GloVe algorithm uses probability distributions to model token co-occurrences, the resulting word embeddings models cannot be used for a prediction task using the GloVe algorithm. Therefore, according to Baroni et al. (2014)'s classification, the GloVe algorithm should be categorized as a count-based DSM rather than a predictive DSM.

The GloVe algorithm is implemented and available as an open-source toolkit, see Appendix A. The implementation of the GloVe algorithm is decomposed into four command-line tools that calculate a vocabulary (vocab), the global token-token co-occurrences (cooccur), shuffle the global token-token co-occurrences statistics and calculate the word embeddings models (glove). The decomposition enables the re-use of shared intermediate results. The GloVe toolkit shares the Vector Size (50), Min-Count (-), Window Size (15) and Iterations (25) hyper-parameters with the word2vec toolkit. In addition to the hyper-parameters shared with the word2vec toolkit, the GloVe toolkit comes with several additional hyper-parameters that affect the quality of the resulting word embeddings models:

- **Max-Vocab (-):** The Max-Vocab hyper-parameter controls the vocabulary size. It is an alternative to the Min-Count hyper-parameter of the word2vec toolkit to exclude infrequently occurring tokens from training. The idea of setting a Max-Vocab hyper-parameter is to set a boundary on the vocabulary size and retain only the most frequent words up to a vocabulary size of the value of the Max-Vocab hyper-parameter.
- **Symmetric (Left and Right Context):** In contrast to the word2vec algorithm, the context window can be configured to use the left (before token) context only or to use as many tokens to the left and right (before and after the pivot token) as the value of the Window Size hyper-parameter is set to. Pennington et al. (2014) reports that the symmetric context window (Left and Right Context option) performs better than a single-sided context window.
- **Distance Weighting (1):** In contrast to the word2vec algorithm that uses a dynamic window size approach, for the GloVe algorithm, the distance between two tokens that co-occur in a context is linearly weighted (1) or not weighted at all (0).
- **Eta (0.05), Alpha (0.75) and X-Max (100.0):** Hyper-parameters that control the learning rate and learning rate modification of the AdaGrad algorithm during training.

The default values of the hyper-parameters are denoted in parentheses. The GloVe toolkit supports check-pointing, i.e., storing a word representations model after a fixed interval of iterations that can be used to significantly speed up hyper-parameter studies because the results of intermediate word embeddings models can be re-used during training. The GloVe toolkit comes by default with a custom file format for word embeddings models.

2.11. Discussion

In this chapter, word embeddings algorithms are introduced and embedded into the historical context of distributional semantics and distributed representations research. The most well known contemporary word embeddings algorithms are identified and discussed: The word2vec, FastText and GloVe algorithms. The word2vec algorithm is an algorithm that unifies the two research fields of distributional semantics and distributed representations. As a consequence, the word2vec word embeddings models inherit the synonymy aspects encoded in DSMs.

Saussure's investigations on natural language suggest that semantics encoded in DSMs is limited to the use of the natural language in a training corpus. Syntactic relationships are used for the training of word embeddings models, while, for the most part, paradigmatic relationships among tokens are exploited from word embeddings models. In the vector space of word2vec word embeddings models, additional linear structures are present. The linear structures suggest that multiple word embeddings vectors can be accumulated to represent phrases, sentences, paragraphs or even documents.

The FastText algorithm picks up on the idea of splitting tokens into character n-grams and to calculate character n-gram embeddings. The FastText algorithm mimics the calculation of syllables embeddings. In contrast to the word2vec and FastText algorithms that are true predictive DSMs, the GloVe algorithm is categorized as a count-based DSM. The hyper-parameters of the word2vec algorithm, FastText and GloVe algorithms are discussed in detail. The hyper-parameters of the different word embeddings algorithms can have a substantial impact on the quality of the resulting word embeddings models. In the literature, the Model Architecture, Vector Size, Window Size and Sample Threshold are reported as the most influential hyper-parameters.

If I have seen further it is by standing on the shoulders of Giants.

Isaac Newton

CHAPTER 3

Foundations and Related Work

This chapter introduces foundations that are used in the following chapters and reviews related work. Related work covers previous work in the areas of AI&Law with a particular focus on legal information retrieval. The related work also encompasses approaches for thesaurus creation and reconstruction, semantic search and query expansion. The foundations include specifics of the German legal domain, variants of text segment and document representations and label propagation algorithms. Further evaluation measures for information retrieval, inter-rater-reliability and system usability are discussed.

3.1. AI&Law

Legal research is an integral part of a lawyer's daily work. Legal information retrieval denotes tool support for legal research. Legal information retrieval is an important part of AI&Law research ever since (Van Opijnen and Santos (2017)). Legal data is subject to several specific characteristics in comparison to general information retrieval. Van Opijnen and Santos (2017) provides a collection of characteristics that are specific to legal data, see Figure 3.1. Most research in legal information retrieval addresses challenges that arise from specific characteristics of legal data or attempt to exploit the properties of legal data, see, for example, Hafner (1978), Bing (1987) or Dick (1991). Ashley (2017) provides an overview of the field.

In contrast to, for example, Twitter tweets, legal documents are often long and of high quality. However, the semantics of the legal language is often very complex. Legal documents are often hierarchically and highly structured. Many different legal document types exist, for example, provisions, court decisions or contracts. Legal systems, including the German legal system, tend to accumulate large amounts of textual data over time. Often, new documents replace existing textual data or have a due date upfront.

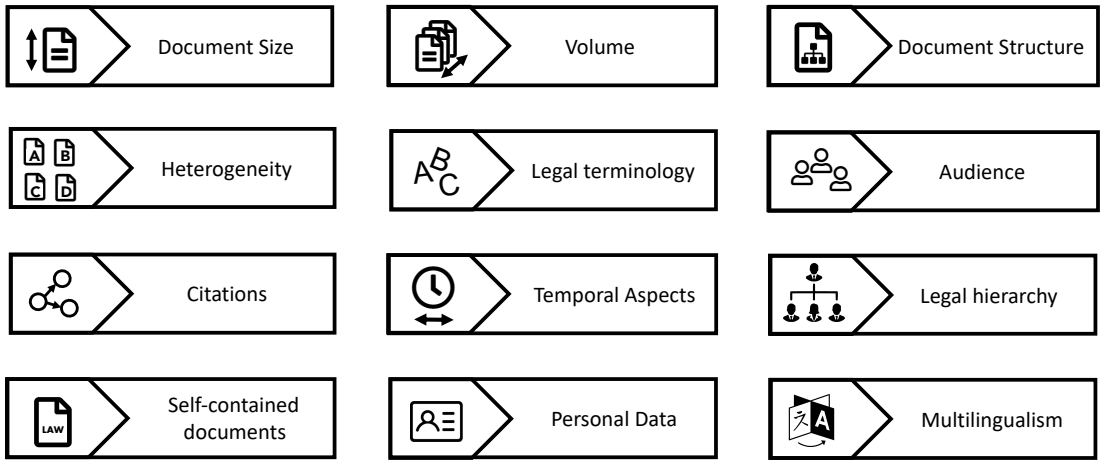

Figure 3.1.: Legal data characteristics according to Van Opijnen and Santos (2017).

The most important aspect of legal data, which addressed in this thesis, is the legal terminology, in particular, synonymy. For example, formulations in contracts need to comply with certain provisions. Legal comments often contain examples of contract clauses with slightly different formulations. As a consequence, the legal terminology not only contains synonymy and semantic relatedness on word level but also on clause, sentence and phrase levels. Furthermore, several additional specifics of legal data are addressed, including different document types, document structure, large document sizes, large volumes of legal data and diverse audiences.

A major trend is to dig deeper into the semantics of legal documents by leveraging NLP technologies. For example, Grabmair et al. (2015) uses an ontology on the subject of intellectual property law that is combined with NLP and machine learning technologies to improve conceptual legal information retrieval of vaccine injury decisions. Waltl (2018) uses the UIMA type system, NLP and machine learning based approaches to engineer a sophisticated tool for legal experts that includes basic and advanced legal information retrieval tools. As part of a master's thesis, Pickel (2016) compares the WE-DF vector space model and Elasticsearch's *More like this* functionality for the retrieval of related legal norms of the German Civil Code (GCC). Schweighofer et al. (2007) uses ontologies and relevance feedback methods to improve legal information retrieval.

Lastres (2015) carried out a survey with 190 associate-level participants on different aspects of legal research and legal information retrieval. According to the survey, up to 15 hours a week are spent on conducting legal research by law firm associates. Peoples (2005) conducted a user study with 56 law students and compares keyword search and a more enhanced version of a keyword search that provides additional operators that can be applied to the keywords. Wiggers et al. (2018) conducts a user study to assess relevance criteria such as the title, the document type or the length of documents

for legal information retrieval systems. The information retrieval system of Wiggers et al. (2018) uses the Okapi BM25 vector space model.

The extraction of citations and implicit references from the GCC is addressed in Landthaler et al. (2015). The conceptual classification of explicit and implicit citations in legal documents is elaborated in Waltl et al. (2016) and Waltl et al. (2017a)

In argumentation mining, premises need to be matched against claims (Palau and Moens (2009)). This task is often called textual entailment. Textual entailment is close to Semantic Text Matching but not equal. Textual entailment puts the focus on logical dependencies while Semantic Text Matching puts the focus on text similarity. Rinott et al. (2015) use the WE-DF vector space model for small text segments to match evidences and claims extracted from debates of performance-enhancing drugs. The results of the WE-DF vector space model are compared to the Okapi BM25 vector space model as a baseline. Naderi and Hirst (2016) conduct argumentation mining for parliamentary debates. Naderi and Hirst (2016) use pre-trained word embeddings models and skip-thought vectors to encode small text segments. Smywiński-Pohl et al. (2019) use word embeddings to construct a legal dictionary for Polish law. Falakmasir and Ashley (2017) extract legal factors from case law documents for argumentation mining. Different vector space models are compared: TF, TF-IDF and variants of the former that only use verbs.

Sadeghian et al. (2016) use word embeddings to classify legal citations. Vo et al. (2017) uses word embeddings to suggest synonyms of terms to experts that carry out a technology-assisted review. Sugathadasa et al. (2019) compare the TF-IDF vector space model and several deep learning approaches using Paragraph Vector (doc2vec) for a legal information retrieval task. The information retrieval task is evaluated with a gold standard evaluation set. As part of a master's thesis, Erl (2018) applies the WE-DF vector space model, doc2vec and the TF-IDF vector space model to a Semantic Text Matching problem for a compliance use case. Adebayo et al. (2016) conducts experiments in information retrieval and textual entailment on the COLIEE 2015 training and test sets using word embeddings. Text segments are represented with a normalized word embeddings vectors of the WE-DF vector space model. Savelka et al. (2019) retrieves sentences from statutory texts.

Waltl and Vogel (2018) reasons on the importance of explainable AI approaches for the legal domain. In Waltl et al. (2018), the LIME approach (Ribeiro et al. (2016)), is applied to a classification of legal norms.

3.2. German Legal Domain

From a computer science point of view, the legal domain is an application domain. From a legal perspective, this thesis focuses on the German legal sphere and the German legal system[1]. In contrast to US law that is dominated by case law and other legal systems in the world, German law has strong roots in Roman law and is dominated by statutory law. German law is embedded in European law. The General Data Protection Regulation (GDPR) is an example of a European law that applies to the German legal sphere. Similar to US law, German and European law are further bound by international

[1] http://ieg-ego.eu/de/threads/crossroads/rechtsraeume-rechtskreise/elisabeth-berger-deutscher-rechtskreis, last accessed July 15, 2019

law. International and European are complex subjects on their own and are not explicitly addressed in this thesis.

The specific characteristics of legal data identified by Van Opijnen and Santos (2017) apply to German legal data, too. Hierarchical relationships constitute a general specific characteristic of legal data. Many different hierarchical relationships occur in German legal data and the German legal system. For example, legal documents such as statutes or contracts are structured hierarchically, legal actors are organized in hierarchical structures such as court instances, but also legislators such as the federation and federal states are organized in hierarchical structures[2]. The German law is composed of general laws such as the GCC and regulations that are specific to a legal field, for example, public procurement law. General rules attempt to cover as many cases as possible. Specific rules are exceptions to general rules, explanations of norms or references to other norms (Larenz and Canaris (1995)). Orthogonal to that German law is governed by general principles such as *proportionality* or the *principle of confidence*. German legal theory provides different methods how the law is applied to a particular case, for example, by the application of subsumption[3] or the analogy method, cf. Larenz and Canaris (1995).

German legal data and the German legal system are also subject to additional specific characteristics. Many legal document types are specific to German-speaking regions; For example, legal comments condense statutory provisions and court decisions. An essential aspect of the German legal domain is the use of the German language that has several specific characteristics. As an example, consider the heavy use of open compound words in the German language. Besides a few international legal publishers, several German legal publishers are specialized in the German market. The German legal system can be considered complex. While the German tax law is probably not the most complex tax law in the world, it can be classified as a very *complex* law[4]. German lawyers are highly trained experts. The German lawyer market is split into a few international law firms and many smaller attorney's offices[5]. Additionally, many other specialized legal actors are important in the German legal system, for example, legislators, judges, lawyers and notaries.

The multitude of different laws, principles, methods and actors as well as the complexity constitute a huge challenge for German AI&Law and German legal information retrieval but sometimes also for legal experts. For example, in extreme cases, two laws can build a contradiction. Prioritization, argumentation and experience are required to resolve such contradictions, cf. Haft and Hilgendorf (2009) and Larenz and Canaris (1995).

In general, more research on German AI&Law is required (Waltl (2018)). In this thesis, research is conducted on the German legal fields of tax law and tenancy law. Several specifics of legal data that have been identified by Van Opijnen and Santos (2017) are addressed in this thesis. In tax law, the dataset contains many different document types of varying sizes and structures. In tenancy law, tenancy contracts and legal comments are considered. In the tenancy law dataset, the hierarchical document structure is exploited. The main actors that can leverage the results of this thesis are legal publishers and their clients that are, for the most part, lawyers.

[2]However, the federation and the federal states also have different competencies

[3]In law, the subsumption denotes the application of abstract rules to a concrete situation.

[4]https://www.spiegel.de/wirtschaft/service/steuern-kommt-die-mehrheit-der-weltweiten-steuerliteratur-aus-deutschland-a-1111192.html, last accessed August 10, 2019

[5]https://www.deutscheranwaltspiegel.de/auf-die-positionierung-kommt-es-an/, last accessed August 10, 2019

3.3. Text Segment Representations and Vector Space Models

Vector space models for information retrieval have a long tradition. The TF-IDF vector space model is used as a baseline and introduced more in-depth. Word embeddings based vector space models are of great interest nowadays but less well researched so far. Several variants to encode text segments or documents with word embeddings exist.

3.3.1. TF-IDF

TF-IDF is a traditional term-frequency based vector space model (Salton and Buckley (1988)) and can be considered as the backbone of nowadays information retrieval systems. The basic assumption of TF-IDF is that tokens that occur only in a few documents are more informative about the topics of a document than tokens that frequently occur in many documents, such as stopwords. TF-IDF uses a combination of global (TF) and local (IDF) term frequency statistics. The weight of a particular token depends on its global occurrence frequency but also on the number of occurrences in a particular document. Note that the weight of a particular token is document-specific.

Mathematically, the TF-IDF weight for a token t is calculated according to a calculation scheme that weighs a term's term-frequency and it's IDF through multiplication: TF-IDF$_t := TF_t * log\frac{N}{DF_t}$. The term-frequency TF_t is calculated as the occurrence frequency of term t in a document. The IDF is the ratio of the total number of documents in the corpus N and the number of documents that contain the token DF_t. The IDF is often weighted by the logarithmic function to mitigate the effect of exploding IDF values for infrequently occurring tokens

A document is represented as a vector of size of the vocabulary, where every dimension of the vector represents a token in the vocabulary and each dimension is assigned the TF-IDF weight for the respective token. An input query of a user is considered as a so-called pseudo-document and also encoded as a document in the TF-IDF vector space. The ranking of documents is usually carried out by cosine similarity.

Several variants and alternatives to the TF-IDF vector space model have been proposed, for example, Okapi BM25 presented by Robertson et al. (2009) and derivatives that integrate term-frequency based methods with probabilistic frameworks. However, only the TF-IDF vector space model will be considered in the research presented in this thesis. With respect to the classification scheme of Baroni et al. (2014), TF-IDF can be considered as a count-based DSM to some degree. However, TF-IDF, equivalently to topic models, operates on token-document frequency statistics rather than token-token frequency statistics, see also Section 6.3.

3.3.2. Word Embeddings Based Vector Space Models

Word embeddings vectors represent individual tokens in the vocabulary of a training corpus. Many tasks, for example, information retrieval, require a representation of text segments such as phrases, sentences, paragraphs or documents, i.e., elements that consist of multiple words and are usually of variable length. The most basic approach that uses word embeddings vectors is to calculate a text

segment embedding by accumulating all word embeddings vectors of the composing tokens, i.e., the WE-DF text representation.

The Bag-of-Words (BOW) assumption is that the order of words in documents and the context of words can be ignored. A word embeddings based vector space model like WE-DF is still an approach that is based on the BOW assumption because the accumulation of vectors ignores the word order. However, information about frequent compositions of words is encoded in word embeddings models and has the potential to incorporate additional semantic aspects.

Many other approaches exist that generalize the idea of word embeddings to text segment embeddings. This section presents a selection of more sophisticated approaches to calculate text segment embeddings than the accumulation of word embeddings vectors.

Several approaches have been proposed that attempt to accumulate word embeddings vectors in a weighted fashion. Ferrero et al. (2017) uses the WE-DF text representation and a part-of-speech (POS)-based variant of the WE-DF text representation for plagiarism detection. Nagoudi et al. (2017) calculates sentence similarity using WE-DF, Word Embeddings - Inverse Document Frequency (WE-IDF) and a POS-variant of the WE-DF text representations and a combination of all methods to calculate the text similarity among Arabic sentences. Júnior et al. (2017) compares WE-DF and WE-IDF text representations for sentiment analysis of Twitter tweets.

An extension of the word2vec algorithm to train text segment embeddings is the doc2vec algorithm (Le and Mikolov (2014)) that is also called doc2vec. The main idea of the doc2vec algorithm is to introduce additional vectors that represent text segments. The additional vectors are trained at the same time with word embeddings. The Distributed Memory Model of Paragraph Vectors (PV-DM) and Distributed Bag Of Words Model of Paragraph Vectors (PV-DBOW) model architectures extend the CBOW and Skip-gram model architectures of word2vec. The word embeddings vectors are shared among all text segments, while the text segment vectors are unique among the text segments. The authors propose two ways to integrate the text segment vectors, either by concatenation of vectors or accumulation. However, in their experiments, they only used the concatenation of vectors.

Technically, the PV-DBOW model architecture samples a training sample of the size of the window size hyper-parameter, but then randomly samples one word from the training sample to form the classification task. Thus, the word order is lost in the PV-DBOW model architecture. For the PV-DM model architecture, the word order is incorporated because complete context windows serve as input to the classification task.

Encoding a new text segment as a text segment embedding, i.e., an inference step, requires a slightly more complex procedure. The word embeddings vectors, as well as the hyper-parameters, are fixed, and the new text segment embedding is calculated via stochastic gradient descent and error backpropagation.

The doc2vec algorithm has been used to tackle Semantic Text Matching problems by Erl (2018). The doc2vec algorithm is reported to work slightly better than the WE-DF vector space model.

Other approaches generalize the idea of word embeddings to phrase, sentence, paragraph or document level. Kiros et al. (2015) introduced the Skip-Thought Vector algorithm. Sentences are considered as the atomic units. In the word2vec algorithm, tokens are considered as the atomic units. Skip-thought vectors can be imagined (in a simplified way) as feeding the word2vec algorithm with sentences rather

than words. Another approach to calculate sentence embeddings is the sent2vec algorithm presented by Pagliardini et al. (2018). The sent2vec algorithm can be thought of as a combination of the idea of the CBOW model architecture and the FastText algorithm.

Last but not least, other artificial neural network based approaches have been proposed. For example, Salakhutdinov and Hinton (2009)'s Semantic Hashing is a deep autoencoder that clamps documents in term frequency representation as input and output. The autoencoder maps similar texts to similar vectors in the hidden layers.

3.4. Label Propagation

Label propagation is a family of semi-supervised graph algorithms that assign labels to unlabeled nodes from (a small number of) labeled nodes (Bengio et al. (2006)). Zhu and Ghahramani (2002) initially presented the idea for label propagation. In contrast to the k-nearest neighbor algorithms, label propagation algorithms take into account not only locally close neighbors but also the global, overall structures in vector spaces. Figure 3.2 shows an illustration of the intended effect of label propagation algorithms on an artificial toy problem. On the left, a minimally labeled dataset is depicted. In the middle, the outcome of a k-nearest-neighbors algorithm shows that the dataset is labeled wrong. On the right, the correct labeling is shown. The goal of label propagation algorithms is to spread labels from a few labeled datapoints to unlabeled datapoints by taking into account the global structure of the data. The intuition of label propagation is that labels from labeled nodes are propagated to unlabeled nodes along the potentially weighted edges of a graph by means of an iterative algorithm. Mathematically, this is achieved through iterative application of matrix-vector operations. In this thesis, the label spreading algorithm Zhou et al. (2004) is used.

The standard label propagation and the label spreading algorithm work on a graph $G = (V, E)$ with nodes V and edges $E = w_{ij}$ represented as an affinity or weight matrix W. Non-existing edges are encoded with zero values, existing edges with one or in the case of weighted edges with the edge weight (unequal to zero). Furthermore, a label distribution matrix $Y^{(0)}$ assigns labels to the nodes using one-hot encoding. New label distribution matrices are iteratively calculated by the multiplication with a transformation matrix T: $Y^{(t+1)} := TY^{(t)}$. In the case of the standard label propagation algorithm, the transformation matrix is calculated as the inverse diagonal matrix $T := D^{-1}$ where the diagonal entries are calculated from the weight matrix $D_{ii} := \sum_j W_{ij}$. In order to retain the initial labels, the labels are reset to the original values after each iteration (clamping). For the label spreading algorithm, the transition matrix S is calculated as $S := D^{-\frac{1}{2}} W D^{-\frac{1}{2}}$ and used by the calculation rule $Y^{(t+1)} := \alpha S Y^{(t-1)}$, where α is a hyper-parameter that controls the smoothness of the transformations.

The graphs that are input to the label propagation algorithms can have different characteristics. Edges can be directed, or undirected, weighted or binary and self-references can be enforced, allowed or removed. The hyper-parameters of the label propagation algorithms include the number of iterations, which optimally runs until convergence, but is not guaranteed to converge. For the label spreading algorithm, also the α smoothness hyper-parameter needs to be chosen.

A more detailed introduction to label propagation, including comprehensive examples, can be found in Mueller (2018). Promising results for knowledge graph extension have been reported, for example,

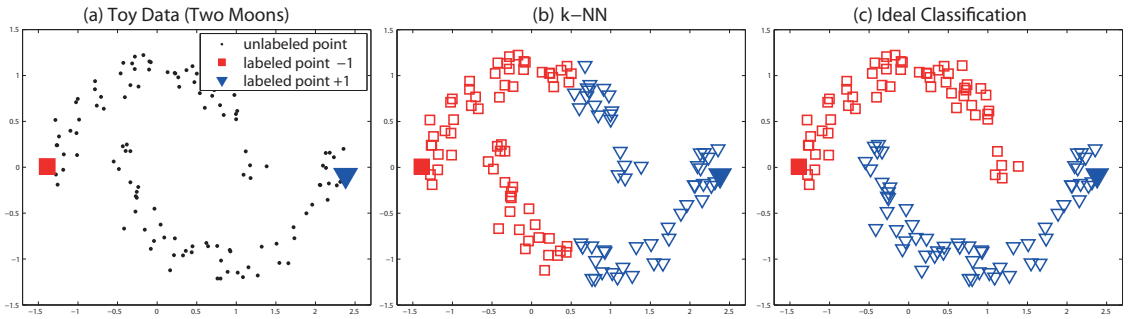

Figure 3.2.: Illustration of the effects of label propagation algorithms (Zhou et al. (2004)).

by Ravi and Diao (2016). Several label propagation algorithms are available as open-source implementations in the scikit-learn library, see Appendix A.

3.5. Thesaurus (Re-)Construction and Thesaurus Extension

Early approaches to automatically discovering relationships among words have been investigated by Jones (1964). Jones (1964) further investigated the linguistic concept of synonymy. A huge bunch of approaches attempts to exploit the distributional hypothesis for thesaurus reconstruction, for example, Takenobu et al. (1997), Uramoto (1996) andMeusel et al. (2010). Grefenstette (1994) gives an overview of the field.

The Sketch Engine (Kilgarriff et al. (2004) andRychlý and Kilgarriff (2007)) and the JoBimText tool suite (Riedl and Biemann (2013), Biemann and Riedl (2013)) are two of the latest approaches that use count-based DSMs. Both constitute large software systems build around count-based DSMs. The most recently used significance measure is Lexicographer's Mutual Information (LMI), cf. Riedl and Biemann (2013). LMI is a variant of the PMI significance measure.

The JoBimText and word2vec algorithms are compared on their capability to reconstruct thesauri by Ramrakhiyani et al. (2015). The comparison of the JoBimText and word2vec algorithms in this thesis shares many characteristics with Ramrakhiyani et al. (2015), however, no qualitative evaluation is conducted by Ramrakhiyani et al. (2015).

As part of her master's thesis, Altamirano Sainz (2015) integrates an existing thesaurus in a legal information retrieval system and conducts a qualitative evaluation of the system. The reconstruction of legal thesauri is addressed as part of a master's thesis by Vos (2017). Mueller (2018) investigates the extension of German legal thesauri with label propagation approaches. The results of Mueller (2018) are re-used for comparison to other approaches in this thesis.

Some authors call the task of thesaurus reconstruction or thesaurus extension lexical expansion. Another important task that DSMs can be used for is word sense disambiguation, i.e., the detection of polysemic relationships among words. Word sense disambiguation is not addressed in this thesis in detail.

AutoExtend has been presented by Rothe and Schütze (2017). AutoExtend is an approach that uses deep

auto-encoders to calculate synset embeddings. The deep auto-encoder can take into account additional resources such as synonym, antonym or hyponym relations from WordNet as additional constraints. The synset embeddings are embedded in the same vector space as the input word embeddings vectors and can be be used for thesaurus extension as an alternative to the Synset Vector Approach. Rothe and Schütze (2017) uses the Synset Vector Approach as a baseline for the comparison with synsets calculated by AutoExtend.

3.6. Semantic Search and Query Expansion

Semantic search is not a precisely defined term. A very general definition is "to search with meaning" (Bast et al. (2016)). This definition separates semantic search from lexical search. Lexical search matches exact literals. Examples for lexical search are vector space models that are based on the BOW assumption. Building semantic search systems is challenging. Consider a lawyer that enters the keywords *cosmetic repairs shifting*. From the given keywords, it is difficult to determine whether the lawyer is seeking for a template formulation of a tenancy contract, the explanation of the concept of "shifting" or specific legal details regarding the *shifting of cosmetic repairs*. Many different aspects can be considered when talking about semantic search, for example, understanding the user intention or incorporating the contextual meaning of words, for example, to conduct query expansion.

There exists a large bunch of approaches to perform semantic search. Background knowledge basis can be used to support the understanding of semantic aspects of query terms. In contrast to keyword search, natural language queries can be processed.

Bast et al. (2016) gives an overview and classification of the field of semantic search. Bast et al. (2016) differentiates three types of queries: keyword, structured and natural language search and three types of data: text, knowledge bases and combined data. Text is considered as unstructured data and knowledge bases are considered as structured data. Combined data is a combination of structured and unstructured data. Query expansion is considered as an extension of keyword search by Bast et al. (2016). In this thesis, thesaurus extension is considered as an important life-cycle activity to support (keyword) search. Moreover, natural language search approaches are surveyed by Bast et al. (2016).

Note that, sometimes, semantic search is restricted to systems that retrieve data from structured data sources like XML or ontologies, cf. Dong et al. (2008).

Query expansion is an attractive research area. A survey of query expansion methods prior to the rise of word embeddings is presented by Carpineto and Romano (2012). Word embeddings constitute a strong focus of query expansion research in recent years. A more recent survey is provided by Azad and Deepak (2019). Using the WE-DF vector space model to encode query and documents can be considered as a very basic approach, see Section 3.3.2.

Many query expansion approaches are based on language models for information retrieval. Hiemstra (2001) provides an overview of influential language models such as vector space models or the probabilistic language model. Approaches that use word embeddings for query expansion are, for example, Ganguly et al. (2015) or Zamani and Croft (2016). Ganguly et al. (2015) extends the probabilistic language model of Ponte and Croft (1998). The Probabilistic language model uses Bayes' theorem to model posterior probabilities. The posterior probabilities are used to rank documents in the corpus. The basic

assumption is that by using word embeddings, the BOW assumption of traditional language models can be overcome to some degree. In the Generalized Language Model, query terms and corpus terms are individually sampled from a neighborhood in word embeddings model's vector space.

The Generalized Language Model can be considered as a relaxed version of the Word Mover's Distance. The Word Mover's Distance attempts to find for each word in a query the closest term in a document according to the word embeddings model's vector space, i.e., the minimal traveling distance among two text segments is calculated (Kusner et al. (2015)). The Word Mover's Distance, as the name suggests, can be seen as a more complex text similarity measure. Calculating the Word Mover's Distance is computationally very costly. More relaxed versions of calculating Word Mover's Distance have been proposed by the authors, too. Similar to Ganguly et al. (2015), Zamani and Croft (2016) proposes an extension of probabilistic language models where posterior probabilities are calculated by leveraging properties of word embeddings model's vector space.

Word embeddings models encode semantic aspects of words. Semantic search in this thesis means to leverage the semantic information encoded in word embeddings models to improve (legal) information retrieval. This translates, for the most part, to exploit the synonymy encoded in word embeddings models, i.e., to conduct or improve query expansion.

3.7. Evaluation Measures

In this section, the evaluation measures used in this thesis are introduced. Manning et al. (2008) provides an overview of evaluation measures used in information retrieval research. Quantitative relevance measures are the most widely used evaluation measures. Quantitative evaluations of information retrieval systems require a *gold standard* - also called *ground truth* evaluation set. For the most part, precision/recall and Mean Average Precision (MAP) are used in this thesis. Additional measures are used in this thesis. Cohen's kappa is an inter-rater reliability measure. The Spearman's rank correlation coefficient is a statistical measure to assess the similarity of the rankings of two ranked lists. The SUS is a qualitative evaluation measure that can be used to evaluate information retrieval systems. The SUS is an established measure to measure a system's usability with human users. The measures are discussed in more detail in the following.

3.7.1. Quantitative Information Retrieval Relevance Measures

Often, a fixed number of top-ranked results recommended by an information retrieval system is evaluated. As a consequence, two types of errors can occur: false positives (FP) and false negatives (FN). Likewise, positive results can be true positives (TP) and true negatives (TN). True negatives are often not considered in information retrieval because all top-ranked results are optimally correct results. True negatives are not considered in this thesis. In the following, $|Q|$ queries, $q_1, \ldots, q_Q$, and for each query $k = 1, \ldots, N$ results are considered.

A widely used relevance measures is precision/recall (Perry et al. (1955)). For one query, precision (P_q) and recall (R_q) are defined as follows:

$$P_q := \frac{TP_q}{TP_q + FP_q} \tag{3.1}$$

$$R_q := \frac{TP_q}{TP_q + FN_q} \tag{3.2}$$

Calculating the precision (P) and recall (R) for several queries is straightforward:

$$P := \frac{\sum_{q=1}^{Q} TP_q}{\sum_{q=1}^{Q} TP_q + \sum_{q=1}^{Q} FP_q} \tag{3.3}$$

$$R := \frac{\sum_{q=1}^{Q} TP_q}{\sum_{q=1}^{Q} TP_q + \sum_{q=1}^{Q} FN_q} \tag{3.4}$$

where

$$TP_q := \sum_{k=1}^{N} r \begin{cases} r = 1 & , k^{th}\, result\, correct \\ r = 0 & , otherwise \end{cases} \tag{3.5}$$

An intuition for precision is that precision measures if a high fraction of correct results of retrieved results is retrieved. Recall measures the fraction of retrieved relevant results from all relevant results. Precision/recall is typically plotted as precision/recall curves. Precision/recall curves are less suited for hyper-parameter studies due to the possibly large number of precision/recall curves to plot. More aggregated evaluation measures are Average Precision (AP) and MAP.

One possibility to obtain a more aggregated relevance measure based on precision/recall is the well known F_1 score (Van Rijsbergen (1974)). F_1 score is the harmonic mean of precision and recall. Thus, a precision/recall curve can be reduced to a single F_1 score curve. F_1 score is defined as follows:

$$F_1 := \frac{2PR}{P + R} \tag{3.6}$$

F_1 score weighs precision and recall equally. F_β score can be used to favor either precision over recall or vice versa, see Manning et al. (2008). An even more aggregated relevance measure for one precision/recall curve is AP. AP is calculated as follows:

$$AP_q := \frac{1}{N} \sum_{K=1}^{N} P_q(K) \tag{3.7}$$

where $P_q(K)$ is the precision for the first K results: $P_q(k) = \sum_{k=1}^{K} P_q$. AP can be considered as a measure of the area under the precision/recall curve. MAP is the average of AP over several queries (an average over averages):

$$MAP := \frac{1}{Q}\sum_{u=1} AP_q = \frac{1}{Q}\sum_{u=1}\frac{1}{N}\sum_{K=1}^{N} P_q(K) \tag{3.8}$$

For all relevance measures so far, $@N$ variants can be calculated, i.e., only up to N top-ranked results are considered. For the MAP, frequently used alternatives are to weigh the sum of the AP for each query with either the total number of correct results M (irrespective of the number of retrieved results) or the minimum of M and N:

$$MAP := \frac{1}{Q}\sum_{u=1}\frac{1}{M}\sum_{K=1}^{N} P_q(K) \tag{3.9}$$

$$MAP := \frac{1}{Q}\sum_{u=1}\frac{1}{min(N,M)}\sum_{K=1}^{N} P_q(K) \tag{3.10}$$

MAP is currently one of the most used aggregated relevance measures in information retrieval. An important property of the MAP relevance measure is that a special weight is put on the top-ranked results. For example, if the top-ranked results are $[1; 0; 1]$, where 1 is a correct and 0 is an incorrect result, then for this single query (using equation 3.8 and $N = 3$) the MAP amounts to:

$$MAP = \frac{1}{3} * (1 * 1 + 0 * 1 + \frac{1}{3} * 1) = \frac{1}{3} * (1 + \frac{1}{3}) = \frac{1}{3} + \frac{1}{9} \approx 0.44 \tag{3.11}$$

3.7.2. Cohen's Kappa Inter-rater Reliability Score

Cohen's kappa is a measure to assess agreement among multiple raters (Cohen (1960)). Cohen's inter-rater reliability measure is defined as:

$$\kappa := \frac{p_0 - p_c}{1 - p_c} \tag{3.12}$$

where p_0 is the relative observed agreement among raters and p_c is the hypothetical probability of chance agreement. Cohen's kappa usually is a real number in the range [-1;1]. According to Landis and Koch (1977), kappa scores in the range of 0.61 to 0.80 can be considered as *substantial consensus* and values larger than 0.8 can be considered as *(almost) perfect consensus*. Note that for some special conditions values outside the range of [-1;1] are possible.

3.7.3. Spearman's Rank Correlation Coefficient

Spearman's rank correlation coefficient (ρ) is a statistical measure to gauge the statistical dependence between the rankings of two variables (Spearman (1904)). Given two rankings rg_X, rg_Y of size N, then Spearman's rank correlation coefficient is calculated as the Pearson correlation coefficient:

$$\rho := \frac{cov(rg_X, rg_Y)}{\sigma_{rg_X} * \sigma_{rg_Y}} \tag{3.13}$$

where $cov(rg_X, rg_Y)$ is the covariance of the rank variables and $\sigma_{rg_X}, \sigma_{rg_Y}$ are the standard deviations of the rank variables. Intuitively, it can be used to assess the similarity of two equally sized ranked lists given that the elements are shared among the variables.

3.7.4. System Usability Scale

User utility encompasses approaches to measure *user happiness*. According to Manning et al. (2008) *user happiness* can be measured with user studies, for example, by letting users engage in tasks and rate specific aspects of an information retrieval system or measuring metrics like *time-to-answer*. However, Manning et al. (2008) also notes that user studies are time-consuming and difficult to conduct.

The SUS is an established usability measure (Brooke (1996)). It consists of a question battery of ten Likert-scale type questions that include factor analysis and results in a single score that can be used to qualify the usability of a software system. For the user study, the SUS questions are translated into german[6]. The German SUS questions are modified to include the improvements suggested by Bangor et al. (2009). The term system is replaced with the specific system name (Selection Search, Keyword Search) and an additional question is added. The additional question is measured as a single, interpretable score that serves as a verification of the SUS score. The SUS questions are listed in the appendix (**Q08**-**Q17** and **Q19**-**Q28**).

3.8. Discussion

In this chapter, foundations that are used later on and related work are described. The review of related work underlines that legal research is an important part of lawyers' daily work. Thus, legal information retrieval is an important topic. Synonymy is an important semantic aspect of legal information retrieval. Query expansion and natural language search are a sub-fields of semantic search and highly relevant for legal information retrieval. Query expansion deals with synonymy aspects of natural language. Standard keyword search requires users to think about reasonable keywords carefully. Natural language search is different to standard keyword search because users enter natural language as a search query. The word embeddings technology has been used more and more in AI&Law research recently but are not understood thoroughly in research. Several research gaps are addressed in this thesis:

1. **German legal information retrieval** is a rarely investigated topic in research.
2. The usage of **word embeddings** in AI&Law research accelerates, but word embeddings algorithms are rarely investigated in depth in AI&Law research.
3. The majority of NLP-related research in AI&Law uses gold standard datasets for evaluations. **Qualitative evaluations** are difficult and expensive to conduct and therefore less frequently used in AI&Law research.

[6] https://experience.sap.com/skillup/system-usability-scale-jetzt-auch-auf-deutsch/, last accessed April 20, 2019

4. **Thesaurus extension** is an important task for legal publishers, but due to the requirement of a qualitative evaluation, thesaurus extension is not addressed in (AI&Law) research extensively.
5. Word embeddings have been rarely investigated for **query expansion in legal information retrieval**.
6. The **WE-DF vector space model** has not been explored for its capability to conduct an **implicit query expansion** extensively.
7. Little research conducts an in-depth comparison of **TF-IDF and WE-DF vector space models, in particular in the context of legal information retrieval**.

In the next chapter, the word embeddings technology is investigated to support the use case of thesaurus extension, i.e., to support an explicit form of query expansion.

The really important thing is learning how to skeptically question and rely on empirical evidence.

Lawrence M. Krauss

CHAPTER 4

Explicit Query Expansion: Thesaurus Extension

For many legal research tasks, finding a large fraction of relevant documents from a database of documents is essential. For information retrieval systems, this translates to achieving a high recall. Due to the use of synonyms in the legal language, query expansion is a widely used approach to increase the recall in legal information retrieval systems. It is common for legal publishers to use query expansion approaches that use a legal thesaurus. The maintenance of thesauri is a difficult, manual and expensive task. In this chapter, different approaches are investigated that leverage word embeddings algorithms to support humans at the task of extending an existing thesaurus. The investigations are carried out using a dataset provided by the Datev eG. The dataset consists of a German tax law corpus and an existing thesaurus. The thesaurus is manually and specifically maintained for the tax law corpus.

From a more technical point of view, the investigated use case comprises finding candidate terms for existing synonym sets (synsets). The use case is very challenging for automation. Several challenges that occurred during the research are collected and summarized. Word embeddings algorithms are particularly well suited to identify synonym relations among tokens in the vocabulary of a suitable training corpus. Different contemporary word embeddings algorithms are examined in this chapter. The Word2vec and FastText algorithms are state-of-the-art algorithms in the class of predictive DSMs. The GloVe algorithm is considered as a recent algorithm in the class of count-based DSMs. The JoBimText algorithm is selected as a representative of traditional count-based DSMs.

The goal is to calculate candidate terms to extend existing synsets. Different approaches are considered to calculate candidate term lists that can be presented to human experts for review. The JoBimText Approach uses traditional count-based DSMs that operate on token co-occurrence statistics. To compare such approaches to word embeddings based approaches, the Single Word Approach is introduced. The Single Word Approach uses single terms of a given synset to calculate a candidate term list. The Single Word Approach is further used to justify the Synset Vector Approach. The Synset Vector Approach calculates the mean of all terms of a given synset - or more precisely - the mean of the word

embeddings vectors of the synset terms. The Intersection Approach combines several candidate term lists obtained by the Synset Vector Approach to improve the ratio of retrieved synonyms. The Label Propagation Approach is investigated due to promising results for ontology construction reported in the literature. The Label Propagation Approach uses graphs calculated from word embeddings models and applies label propagation algorithms to the graphs to calculate a partition of all tokens in the word embeddings model's vocabulary into synsets. All approaches use a pipes & filter architecture, see, for example, Buschmann (1998). Pipes & filter architectures subsequently process data in different algorithmic steps.

All considered word embeddings algorithms can be combined with all thesaurus extension approaches. All considered word embeddings algorithms come with a large bunch of hyper-parameters that need to be adjusted for a specific task. The existing thesaurus is used as a "gold standard" thesaurus to identify suitable hyper-parameter configurations. The RP-Score is introduced as a straightforward quality measure to assess the quality of word embeddings models with the gold standard thesaurus to investigate the hyper-parameters of the word embeddings algorithms. To assure the validity of the results, the RP-Scoree is compared to the MAP relevance measure.

The investigated use case is the extension of thesauri and not the reconstruction of a given thesaurus because a real-world thesaurus is "never" complete. The suitability of word embeddings based approaches to extend thesauri is assessed through a qualitative evaluation of calculated candidate term lists. More than 2.500 candidate terms have been manually evaluated by at least two reviewers per candidate term. Exemplary candidate term lists are presented where adequate to explain the results of the evaluations. Several measures and experiments are taken to assure the validity of the results. The research presented in this chapter builds on research presented in Landthaler et al. (2018c). Research results presented in Mueller (2018) are re-used.

4.1. Use Case

It is common for (German) legal publishers to use thesauri for query expansion for legal information retrieval. The creation and maintenance of thesauri in the legal domain are a difficult, manual and expensive task (Dirschl (2016)). The life-cycle of a thesaurus encompasses several activities: the initial construction, the maintenance or the reduction or extension of a thesaurus. Additional use cases can be thesaurus merging, for example, with freely or commercially available thesauri or quality assurance, too. While all use cases could leverage word embeddings, the focus of this thesis is the thesaurus extension use case. The thesaurus extension use case obviously requires an existing thesaurus.

A reasonable approach to reducing the complexity of maintaining legal thesauri is to create and maintain a separate thesaurus per law domain, for example, one thesaurus for tenancy law and one thesaurus for tax law (Dirschl (2016)). This thesis follows this recommendation and focuses on the German tax law domain. However, German tax law is a broad legal field on its own. The creation, maintenance and extension of legal thesauri requires expensive domain experts. Dirschl (2016) proposes a standardized process to create such domain thesauri manually. Dirschl (2016) estimates costs of 10-20,000€ for the creation of a thesaurus per law domain. For the existing thesaurus used in this thesis, this amount of money is very likely not sufficient.

In prior research, the focus has been on the reconstruction of existing thesauri, in particular with

(count-based) DSMs. Despite the immense pile of research on automated creation of thesauri, legal thesauri reconstruction has been rarely investigated in research. The extension of legal thesauri has, to the best knowledge of the author, not been investigated previously. One reason is a large effort of conducting qualitative evaluations that are required because gold standard thesauri are "never" complete for real-world datasets. Nevertheless, the extension of existing thesauri is an important use case for legal publishers because often a thesaurus that is used for query expansion already exists. However, legal corpora change over time and thesauri need to be readjusted. The creation and extension of (German) legal thesauri are subject to several challenges that are discussed in the next section.

4.2. Use Case Challenges

The creation and extension of German legal thesauri are subject to several challenges due to the nature of thesauri, characteristics of the German language and (German) legal language specifics. The challenges that arose while conducting this research are summarized in the following list:

1. **Relation types:** Thesauri are lightweight ontologies. While ontologies can capture almost any type of relationships among terms, thesauri usually cover a limited number of relationship types. The most common relationship is the synonym relationship. However, thesauri can cover other semantic relationship types such as antonyms, hyponyms, hypernyms, hyperonyms or more technical relationships like abbreviations. The identification of different relationship types often requires different technological solutions.
2. **Scope of an individual synset:** It is a difficult task to determine the scope of a particular synset. As a simple example, a synset of the terms "car" and "truck" is used here. Both terms characterize vehicles but the terms characterize different types of vehicles. In tax law, it can be a necessity to consider all types of vehicles for a tax-related inquiry. However, it can also be required to search only for synonyms of a particular vehicle subclass. For example, cars can be classified in a different tax class than trucks. Even this simple example illustrates the potential complexity of the determination of a synset's scope that often depends on particular inquiries to a search engine and often is a difficult decision even for human experts. Eventually, hierarchical relations can help, but it is not possible to model hierarchical relations with the synsets concept.
3. **Neighborhood of synsets:** The scope of individual synsets affects the scopes of two or more synsets that are semantically close. To continue the ["car", "truck"] synset example, two separate synsets that contain "cars" and "truck" synonyms might be either combined as one synset or not. It can be difficult to set the semantic boundary between such two synsets. The more semantically close synsets are involved, the more difficult it can become to set semantic boundaries for synsets.
4. **Coincidence of polysemy & synonymy:** A particular challenge of natural language is that synonyms can be polyseme at the same time. For example, the two words "financial institute" and "bank" but also the words "bank" and "bench" can form two separate synsets. The identification of polysemes requires additional technological solutions, for example, word sense disambiguation technologies.
5. **Variations of word forms:** Words in natural language come with many variants such as abbreviations or flections due to grammar rules, different spellings, spelling errors or ways to form

open compound words. For the given dataset and tax law use case, many abbreviations of court names occur and are highly relevant. In the (rare) case of spelling errors in judgments, the spelling errors cannot be changed in the original documents and therefore should be included in a thesaurus. Words are spelled differently. For example, numbers can be written with numeric characters or as words. Many synsets in the given thesaurus reflect such variations of word forms.

6. **Open compound words:** Open compound words, or legal terms such as "adequate rest period" that are a composition of multiple tokens, are heavily used in the German language. Open compound words are formed by either direct concatenation, or whitespaces or hyphens separation. In the investigated corpus, often all possible combinations of forming open compound words are present.

7. **Defining parts of words:** In some cases, few characters can change the semantics of a single term to a strong degree. For example, abbreviations, where a single added character changes the semantics of the abbreviation completely. Single characters can change a word type, for example, "state" (noun) vs. "stated" (adjective, past participle). This constitutes especially a challenge for DSMs that use sub-word level statistics like the FastText algorithm.

8. **Definition of legal terms:** A specific characteristic of legal language is that legal terms are defined as vague and precise at the same time, cf. Van Opijnen and Santos (2017). However, the exact definition or fine-grained semantical differences of legal terms are often decisive for a particular user inquiry in legal information retrieval, for example, the slightly different but highly relevant semantics of the terms "net salary" and "gross salary". It is very challenging for DSMs to capture such semantical differences.

9. **Tax law specifics:** Tax law regulates a very broad law domain. Many different objects of daily use are subject to taxes. As a consequence, the natural language present in the documents is a mixture of tax-law-specific terms and terms occurring in general natural language. However, terms occurring in general natural language can have a sharper definition in the context of tax law.

10. **NLP pre-processing:** Pre-processing is typically one of the most challenging parts of any NLP application, cf. Nogueira et al. (2008). This applies particularly to large datasets that have been created over a long period of time. Such corpora typically include many special cases that require manual detection and exceptional treatment. For example, different legal document types frequently use different symbols for the hierarchical structuring of documents. Another example is that language rules can change. For example, due to the German spelling reform in 1996, in many cases the "ß"-symbol is now replaced with "ss".

11. **Infrequently occurring terms:** Infrequently occurring terms constitute a technical challenge for DSMs because DSMs are based on statistical analysis. Few example usages of a term lead to statistically vague values. However, less frequently occurring words are often not very important, except when a high recall is crucial.

12. **Evaluation measures:** A special challenge to the use case at hand is the appropriate evaluation of technical solutions that support thesauri life-cycle activities. For example, it is questionable if an existing thesaurus can be used as a so-called "gold standard" for particular NLP tasks. In

the case of thesaurus extension, a gold-standard can not be used because real-world thesauri are "never" complete.

13. **Corpus updates:** Another challenge to maintaining thesauri is that corpora change over time. For example, documents are added, deleted or modified. It is difficult to relay such changes to the respective thesauri because of the constant nature of change.
14. **Complexity management:** Thesauri can grow to a large extent over time. The complexity of the general management of a thesaurus but also individual challenges of synsets, for example, determining an appropriate scope of a synset, become even more difficult for large thesauri (curse of dimensionality).

The challenges are not always independent and trade-offs have to be made. For example, open compound words can be detected to some degree by different technological solutions at the expense of fewer occurrences of individual open compound words.

4.3. Technical Approaches

A word embeddings model consists of a set of word embeddings vectors, one word embeddings vector for one token in the corpus that a embeddings model has been trained upon. Depending on the choice of the Min-Count hyper-parameter, all tokens in the corpus or only a subset of tokens in the corpus is represented by a word embeddings vector. The synonymy relations encoded in word embeddings models can be accessed by calculating the cosine similarity among the word embeddings vectors of a word embeddings model.

The cosine similarity calculates the angle between vectors in (high-dimensional) vector space. The commonly used heuristic is that smaller angles between tokens correspond to a high probability of a synonymy relation among the tokens. Different approaches that leverage word embeddings models are considered in the following. Moreover, the JoBimText Approach uses traditional count-based DSMs.

The Single Word Approach calculates a candidate term list for a single query term using word embeddings models and cosine similarity. Three approaches are explored that calculate synset embeddings. Synset embeddings represent a synset with a synset vector that lives in the same vector space as the word embeddings vectors. The Synset Vector Approach directly uses synset embeddings to calculate candidate term lists. The Intersection Approach uses the intersection of several candidate term lists calculated with the Synset Vector Approach. The Label Propagation Approach uses label propagation algorithms to calculate a partition of all terms in a word embeddings model's vocabulary into synsets.

The existing thesaurus is called an input thesaurus when a clarification is helpful. The synsets of the existing thesaurus are called input synsets when necessary. The terms of the input thesaurus are removed from a word embeddings model's vocabulary before calculating candidate term lists. For the JoBimText Approach terms of the input thesaurus are removed from the calculated candidate term lists.

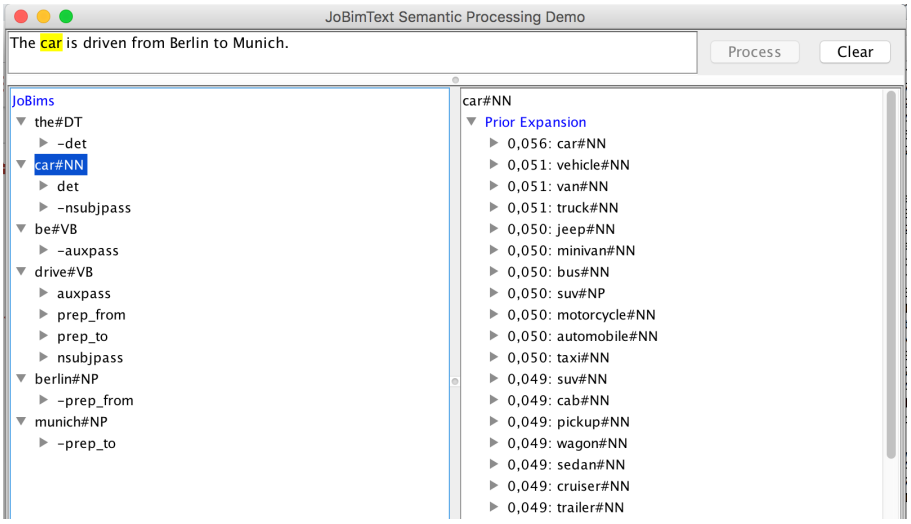

Figure 4.1.: Screenshot of the JoBimText tool suite's GUI.

4.3.1. JoBimText Approach

The JoBimText Approach uses the JoBimText algorithm that is a representative of traditional count-based DSMs. Count-based DSMs have a long tradition in thesaurus construction. Besides thesaurus construction, count-based DSMs can be used for thesaurus extension, too. Two well-known implementations for count-based DSMs are SketchEngine (Kilgarriff et al. (2004)) and the JoBimText algorithm (Biemann and Riedl (2013)). The JoBimText algorithm is selected because the implementation is published as an open-source tool suite, comes with a ready-to-use virtual machine and implements the most important significance measures: LMI, PMI and log-likelihood. The LMI significance measure shows the best results, cf. Riedl and Biemann (2013).

JoBimText is an extensive tool suite that includes the calculation of distributional thesauri from text corpora. The JoBimText tool suite offers an API to access its functionality. A distributional thesaurus is calculated by applying a significance measures to token-token co-occurrence statistics. The resulting distributional thesaurus can be seen as a graph of tokens and relations among the tokens. The JoBimText Approach uses the neighbors of query terms. The neighboring tokens can be ordered according to the significance measure and used as suggestions for thesaurus extension. The JoBimText tool suite offers a GUI, see Figure 4.1. The screenshot shows the neighboring words for the term "car". Note that in the screenshot POS-tags are included in the calculation. For the JoBimText Approach, a variant is used that does not use POS-tags. The JoBimText algorithm can only present the ordered list of neighboring words for a single query term, i.e., it is not possible to give several query terms as a combined input to the distributional thesaurus to find synonymous terms. The JoBimText Approach is compared to the Single Word Approach because it finds synonyms for a single query term, too.

Figure 4.2 illustrates the processing pipeline of the JoBimText Approach. The pre-processing is different than for word embeddings based approaches because the JoBimText tool suite requires a specific input format: One case sensitive sentence per line rather than one single line of lowercased tokens without punctuation. From the pre-processed corpus, a distributional thesaurus is calculated. The implementation uses the Hadoop framework that allows a distributed calculation. The calculated dis-

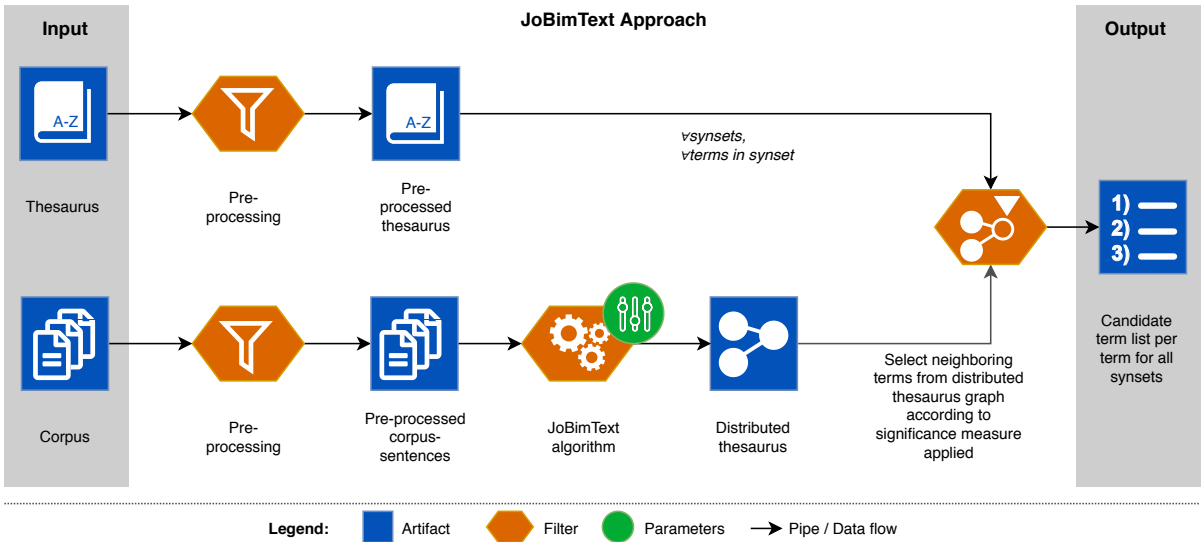

Figure 4.2.: Processing pipeline of the JoBimText Approach.

tributional thesaurus is internally stored in a MySQL database. Similar tokens to a query token can be accessed via the API that queries the distributional thesaurus graph with SQL statements.

4.3.2. Single Word Approach

The Single Word Approach is the most basic approach that uses word embeddings for thesaurus extension. The Single Word Approach is used in this thesis as a vehicle approach to compare the JoBimText and Synset Vector Approaches and to justify the Synset Vector Approach later on.

The Single Word Approach successively takes single terms from existing synsets and calculates ranked candidate term lists from the word embeddings model's vocabulary. For all query terms, candidate terms are calculated by ranking all tokens in the vocabulary of a word embeddings model using cosine similarity. Terms in the existing thesaurus are removed from word embeddings models before the calculation of the candidate terms and stored separately. Figure 4.3 illustrates the processing pipeline of the Single Word Approach. After pre-processing of both, thesaurus and corpus, a single word embeddings model is calculated.

For all terms in an existing synset, a fixed-length candidate term list is obtained. From a ranked list of tokens in the word embeddings model's vocabulary, it is difficult to decide how many of the top-ranked tokens should be retained. Either a fixed number of candidate terms is used or a threshold is used to limit the number of candidate terms. Selecting a threshold based on cosine similarity is difficult because the cosine similarity of even the top candidate term differs profoundly for different synsets. A fixed result set length is chosen for the remainder of this thesis. Another downside of the Single Word Approach is that due to the potentially large number of terms in an input synset, a candidate term list of fixed size is calculated for each term in an input synset. This potentially results in a large number of candidate terms that would require review by human experts.

In summary, the Single Word Approach is a simple baseline and vehicle approach to compare the

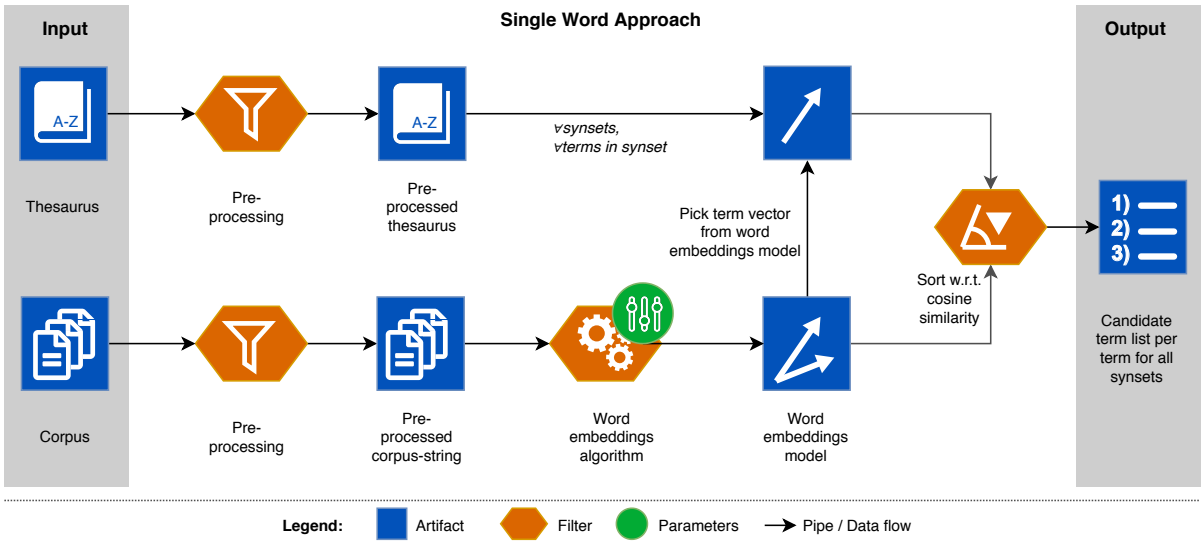

Figure 4.3.: Processing pipeline of the Single Word Approach.

JoBimText Approach to word embeddings based approaches. Nevertheless, the Single Word Approach provides additional insights into word embeddings on its own.

4.3.3. Synset Vector Approach

The Synset Vector Approach is a slightly more advanced approach than the Single Word Approach. A single vector, the so-called synset vector, is calculated from the word embeddings vectors of all terms in an existing synset by taking the mean of the word embeddings vectors. The synset vector represents the existing synset. The synset vector is used similar to individual query terms' word embeddings vectors in the Single Word Approach to calculate candidate term lists from the word embeddings model's vocabulary using cosine similarity. The Synset Vector Approach has been used as a baseline approach by Rothe and Schütze (2017). The motivation to use a synset embedding vector to represent a synset is that synset embeddings should entail the semantics of a synset *better* than the individual terms of a synset on their own. One rationale is that multiple terms should better represent the scope of a synset. For example, *car* and *truck* set the scope of a synset broader than the scope of the term *car* alone.

Figure 4.4 illustrates the processing pipeline for the Synset Vector approach. Similar to the Single Word Approach, after pre-processing of both, corpus and existing thesaurus, a single word embeddings model is trained on the corpus. In contrast to the Single Word Approach, the mean of the word embeddings vectors of all terms of a synset in the existing thesaurus is calculated. Again, similar to the Single Word Approach, the cosine similarity is used to calculate a candidate term list with all tokens in the word embeddings model's vocabulary except for the tokens of the input synsets.

Equivalent to the Single Word Approach, the candidate term list needs to be limited by either a manually chosen fixed length of the result sets or a threshold. Again, the fixed-length option is chosen for the remainder of this thesis. For both, the Single Word Approach and the Synset Vector Approach, word embeddings models can be trained with any word embeddings algorithm. The Synset Vector Ap-

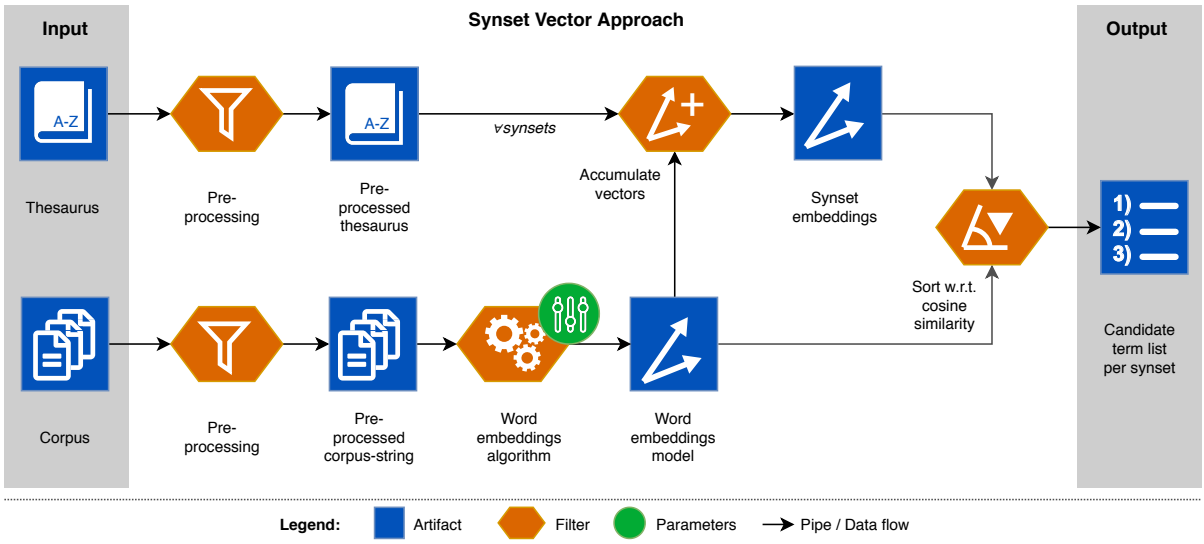

Figure 4.4.: Processing pipeline of the Synset Vector Approach.

proach can be used to improve the quality of suggested terms and to reduce the number of candidate terms presented to human reviewers.

4.3.4. Intersection Approach

The Synset Vector Approach delivers a fair amount of suggestions of good quality. However, word embeddings algorithms calculate a robust word embeddings model only for tokens that frequently occur in the corpus. There is no automated way to automatically determine a suitable amount of results to human experts. To some degree this can be seen as to automatically determine the scope of a synset. It can be observed that the top-ranked suggestions calculated from word embeddings models that include infrequently occurring tokens vary a lot even for small changes to the hyper-parameters, such as one additional training iteration. Both problems serve as a motivation for the Intersection Approach.

The main idea of the Intersection Approach is to calculate a stable core of candidate terms for a synset from more than one word embeddings model. The top-ranked tokens in candidate term lists calculated with the Synset Vector Approach tend to differ a lot. An intersection of the candidate term lists retains high-quality candidate terms and filters noisy candidate terms. A side-effect of the Intersection Approach is that the length of the returned candidate term list is determined automatically.

From another point of view, the Intersection Approach can be seen as a form of extrapolation. Extrapolation was presented in Romberg (1955) and is also called Romberg's method. In the context of numerical calculations, Romberg's method is an approach for the numerical calculation of integrals. The values of several numerical integrals over differently sized grids are combined in a way so that results with higher accuracy than the accuracy of the individual integrals can be calculated. Similar to that, the Intersection Approach takes candidate term lists from multiple word embeddings models and combines them to obtain candidate term lists of higher quality. The Intersection Approach has

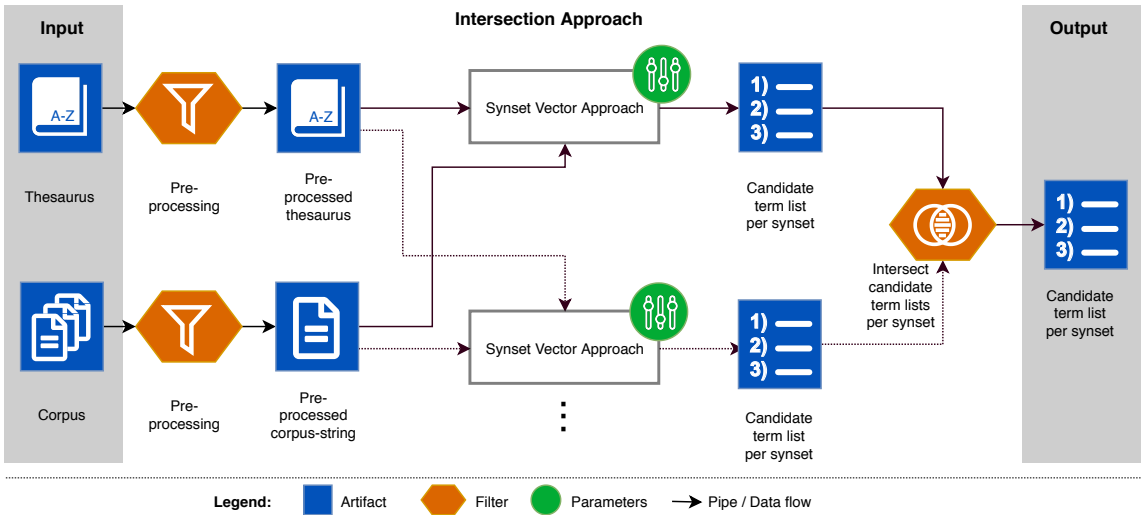

Figure 4.5.: Processing pipeline of the Intersection Approach.

been previously presented in Landthaler et al. (2018c). In addition to Landthaler et al. (2018c), the Intersection Approach is investigated in more depth in this thesis and compared to other approaches for thesaurus extension.

Figure 4.5 illustrates the processing pipeline of the Intersection Approach. Several candidate term lists are calculated for each synset in the input thesaurus with the Synset Vector Approach with slightly varied hyper-parameters for the word embeddings algorithm, for example, with variations of the Iterations hyper-parameter. The intersection of multiple candidate term lists is calculated for each synset separately. The Intersection Approach introduces additional hyper-parameters. The intersection of the candidate term lists of all tokens in word embeddings models vocabularies does not lead to a reduction of the result lists. A subset of the top-ranked results per synset needs to be selected. The maximum candidate term lists size sets a maximum to the final candidate term list length. The space of possible hyper-parameter variations and the number of considered word embeddings models can be varied. For example, several hyper-parameters can be changed at the same time for the calculation of two word embeddings models.

4.3.5. Label Propagation Approach

Label propagation denotes a class of algorithms that conduct semi-supervised learning on graphs. The core idea of label propagation algorithms is to propagate labels from a small set of labeled nodes to (a usually larger set of) unlabeled nodes of a graph. Label propagation algorithms have been used for large-scale ontology extension, for example, by Ravi and Diao (2016). The hypothesis and primary motivation for the Label Propagation Approach are that, in addition to direct neighbors in word embeddings model's vector space, also chains of synonyms might be determined. For example, a similar word to "car" is "truck" and "truck" is close to "bus", while "car" and "bus" are not necessarily semantically that close.

When label propagation algorithms shall be used for thesaurus extension, the thesaurus extension

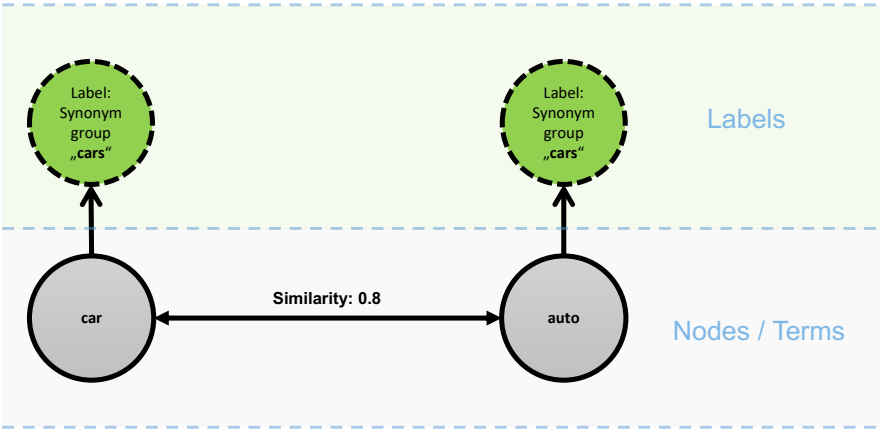

Figure 4.6.: A model for thesaurus extension with label propagation algorithms.

task needs to be modeled as a graph labeling task. Figure 4.6 illustrates the selected model of the thesaurus extension task as a graph labeling task. The graph can be constructed as follows: Each node represents a token of a word embeddings model. The edges represent the neighbors of a node. The nearest neighbors can be calculated using cosine similarity among the word embeddings vectors, for example, with k-nearest-neighbor algorithms or the neighbors are calculated as the neighbors in the ϵ-neighborhood of a token in word embeddings model's vector space. The weights can be binary, i.e., one (or zero) represent present (or absent) neighborhood relations. Alternatively, edges can be weighted with cosine similarity values. The graph can optionally contain self-references. Afterward, the nodes that represent terms in the input thesaurus are assigned a label. A reasonable label is an integer that acts as a unique identifier of a particular synset.

Label propagation algorithms take a graph with a small number of labeled nodes as input and *propagate* the labels to unlabeled nodes and hence assigning labels to unlabeled nodes. This corresponds to assigning a synset label id to unlabeled nodes. To retain labels of the input thesaurus, a reassignment of the label nodes might be necessary after each iteration of the label propagation algorithm. The result is typically a partition of all tokens in a word embeddings model's vocabulary, i.e., all tokens get assigned one label (single-label classification). Thus, the application of label propagation algorithms should also automatically solve the problem of identifying the correct number of assignments for a particular synset.

The Label Propagation Approach is comparably complex and additional hyper-parameters are involved. Reasonable hyper-parameter values need to be determined. For example, k and ϵ for k-nearest neighbor and ϵ-neighborhood algorithms for graph construction but also the Iterations hyper-parameter for the label propagation algorithm are hyper-parameters that need to be set. Label spreading is a label propagation algorithm that comes with an additional α hyper-parameter that controls the propagation "speed". Many more label propagation algorithms have been proposed in the literature that possibly come with additional hyper-parameters.

Figure 4.7 shows the incorporation of the label propagation algorithm into a thesaurus extension processing pipeline. Word embeddings models are trained equally to the Synset Vector Approach but form the input to the graph construction part of the pipeline, together with the input thesaurus. The labeled

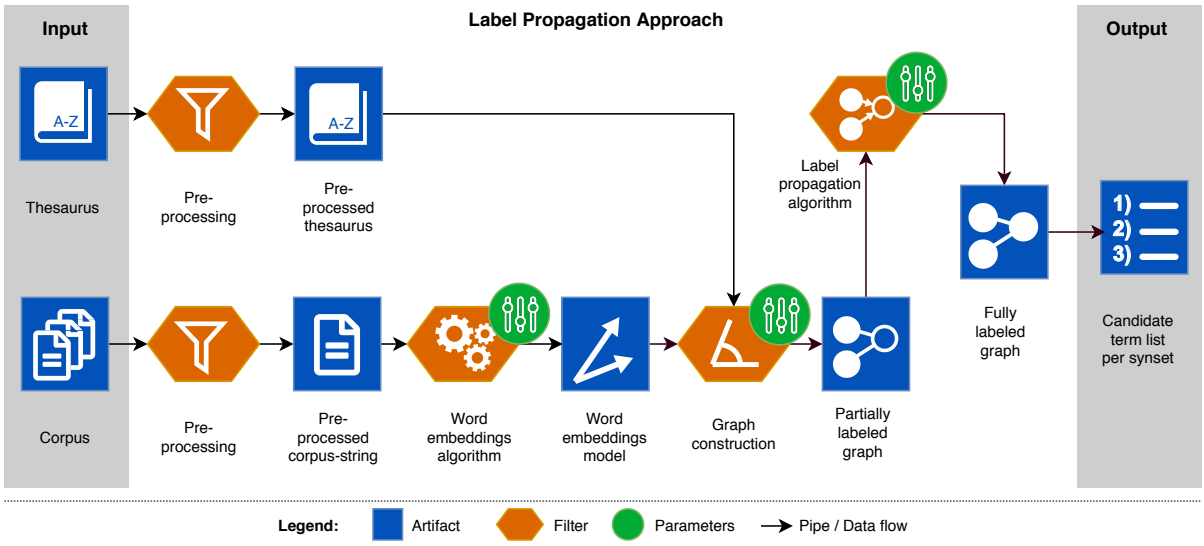

Figure 4.7.: Processing pipeline of the Label Propagation Approach.

graph is the input to the label propagation algorithm. The output of the label propagation algorithm is a partition of all words in the word embeddings model's vocabulary into synsets.

4.4. Dataset

All experiments in this chapter are conducted using a dataset that is provided by the industry partner Datev eG. The Datev eG is a registered cooperative society that provides information technology services to tax accounts and lawyers. The German tax law dataset consists of a text corpus and a thesaurus for German tax law. The thesaurus is specifically maintained for the text corpus and used mainly for query expansion for German legal information retrieval. The corpus encompasses different document types, cf. Figure 4.8. The majority of documents are court decisions of different German courts. The dataset has also been used by Waltl (2018) and Waltl et al. (2017b) to extract semantically complex information such as legal definitions or the year of the dispute of judgments with rule-based approaches.

For this thesis, the text corpus serves as a training corpus to train word embeddings models. The corpus consists of more than 130.000 documents. The documents are of different lengths and in parts date back to the early 20th century. The corpus provides meta-information for each document. However, for all experiments, the raw full-text of the documents is used. The corpus comprises roughly 150 million tokens. Table 4.1 shows the resulting vocabulary sizes depending on the minimum occurrence frequency of terms in the training corpus. The minimum occurrence frequency of tokens in the training corpus has a substantial impact on the quality of resulting word embeddings vectors. A minimum occurrence frequency of five is the default value in the toolkits of all investigated word embeddings algorithms. In general, a minimum occurrence frequency of five leads to stable word embeddings vectors. The downside of setting a minimum occurrence frequency larger than one is the out-of-vocabulary prob-

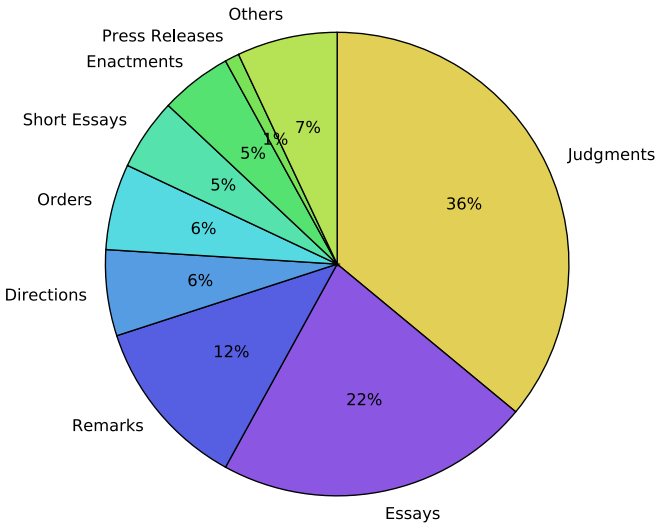

Figure 4.8.: Document type distribution of the German tax law corpus (Landthaler et al. (2018c)).

	Min-Count 1	Min-Count 5
Vocabulary size	ca. 609000	ca. 188000

Table 4.1.: Vocabulary sizes of two Min-Count hyper-parameter configurations.

lem. Even small values of minimum occurrence frequency result in a drastically reduced vocabulary size.

The accompanying thesaurus encompasses several concepts, such as synonym and antonym sets. The unprocessed thesaurus contains around 12.000 synsets. Exemplary synsets are presented in Appendix D. More detailed statistics of the dataset are provided in the next section.

4.5. Pre-processing

The dataset comprises a collection of documents and a thesaurus. To train word embeddings models and to run the subsequent parts of the processing pipelines, both, the documents and thesaurus are pre-processed. The word embeddings algorithms require as input a single string of whitespace-separated tokens. Tokens represent pre-processed words and are considered as the atomic units of the pre-processed dataset. The text corpus comprises the raw full-text of all documents in the dataset. The raw full-text is pre-processed with the following steps in the given order:

1. **Newline replacement:** Single or compound newline characters are replaced with a single blank whitespace character.
2. **Whitespace tokenization:** The raw full-text strings are split into tokens by splitting the strings at whitespace characters.

	Full thesaurus		Train/test splits of thesaurus			
			Min-Count 1		Min-Count 5	
			Train	Test	Train	Test
#synsets	2552		2552	2552	2552	2552
#terms	6164		6508	5827	3277	2887
#relations	10894		-	-	-	-
synset size	2.41		1.31	1.17	1.28	1.13

Table 4.2.: Evaluation thesauri statistics for two Min-Count hyper-parameter configurations.

3. **"§"-symbol replacement:** The paragraph symbol plays an important role in German legal documents. The paragraph symbol indicates internal and external references. At a later step, all one-character tokens will be removed. To retain the information, the paragraph symbol is replaced with a unique token consisting of a larger amount of characters that are unlikely to be used otherwise: *PARAGRAPHSYMBOL*.
4. **Special character removal (except hyphen):** All punctuation such as periods, commas, semicolons, quotation marks and similar characters need to be removed. The easiest way to accomplish this is to retain only alphanumeric characters. Hyphens between alphanumeric characters are retained because in the German language, many compound words are written out with hyphens.
5. **One- and two-character token removal:** Despite the previous pre-processing steps, the pre-processed corpus contains many tokens of one or two characters. One or two characters tokens result from text organization elements and elements specific to the legal domain in the raw text. Examples for such elements are hierarchical text segment identifiers and list element identifiers. Thus, all tokens with less than three characters are removed. However, with this simple heuristic, not all undesired elements are removed, for example, the list item identifier "iii" that is part of the sequence "i", "ii", "iii".
6. **Lowercasing:** In German language, sentences start with upper case characters (with very few exceptions). Nouns, for the most part, also start with upper case characters. Other word types start with a lowercase character most of the time. To maximize the number of example usages for tokens, all tokens are lowercased. It might be useful to retain uppercase forms of nouns. However, the differentiation of nouns and non-nouns at the beginning of sentences is difficult.

The resulting list of tokens is concatenated with single whitespace characters. After the pre-processing, a corpus of approximately 144 million tokens is obtained. Other typical pre-processing steps in NLP such as stemming or lemmatization are counterproductive for the use case at hand because the goal is to identify different word forms such as flections, abbreviations and different spellings of terms.

The existing thesaurus is pre-processed as follows:

1. **Multi-token terms removal:** All terms that contain whitespaces are removed. This procedure removes many terms, but the identification of multi-token terms (open compound words) in the corpus is a difficult challenge and not investigated in this thesis.

2. **Lowercasing:** To comply with the pre-processing of the corpus, all characters in the thesaurus are lowercased.
3. **Single Synset Assignment:** Several terms occur in more than one synset. To apply the label propagation algorithm, occurrences of terms in multiple synsets are removed. The largest synset retains the terms while the terms are removed from smaller synsets. Label propagation algorithms calculate a partition of all tokens in a word embeddings model vocabulary. Assignments of terms to more than one synset would require more complex multi-label label propagation algorithms that are not considered in this thesis.
4. **Out-of-vocabulary removal:** Infrequently occurring tokens are difficult for word embeddings algorithms and therefore a minimum occurrence frequency for terms is used in several cases to obtain stable word embedding vectors. As a consequence, not all terms in the thesaurus are present in the word embeddings models. All terms not present in the word embeddings models are removed from the thesaurus.
5. **Single terms synset removal:** After the duplicate removal and the out-of-vocabulary removal steps, some synsets are left with a single term. These synsets are removed from the thesaurus because single term synsets cannot be used for the evaluations.

The thesaurus does not contain "§"-symbols, single-character tokens or special, non-alphanumeric characters. A more detailed analysis of pre-processing alternatives and their impact on the Datev eG dataset is discussed in Mueller (2018).

Statistics of the pre-processed thesaurus are displayed in Table 4.2. The pre-processed thesaurus has a total of 2552 synsets with 6164 terms. A total of 10894 relations can be used for the RP-Score evaluations. The largest decimation of terms and synsets is due to the out-of-vocabulary where only around 50% of synsets remain. Due to the single terms synsets removal, another large fraction of synsets is removed. After the pre-processing, the majority of synsets is left with only two terms, cf. the histogram displayed in Figure 4.9. The histogram displayed in Figure 4.9 shows the amount of different synset sizes for the thesaurus pre-processsed with Min-Count 5.

To maximize the comparability and to reduce the computational effort, the corpus and the thesaurus are pre-processed once. This pre-processed thesaurus is used for all experiments where possible. For the quantitative evaluations that use the RP-Score and the qualitative evaluation, the full pre-processed thesaurus is used. For the quantitative evaluations that use precision/recall or MAP, the pre-processed thesaurus is split into one training- and test-set split. One experiment investigates if several folds of train-/test-splits lead to more robust results, but this is not the case as will be demonstrated. Table 4.2 additionally shows average statistics of the training- and test-sets. The sampling strategy is to split all synsets into 50% training and test subset terms. For an uneven number of terms, the majority of terms is assigned to the training subset. In data science, it is common to have a smaller test set and a larger training set. However the splitting of the synsets into more typical shares of train- and test-subsets is difficult because many synsets have only few terms.

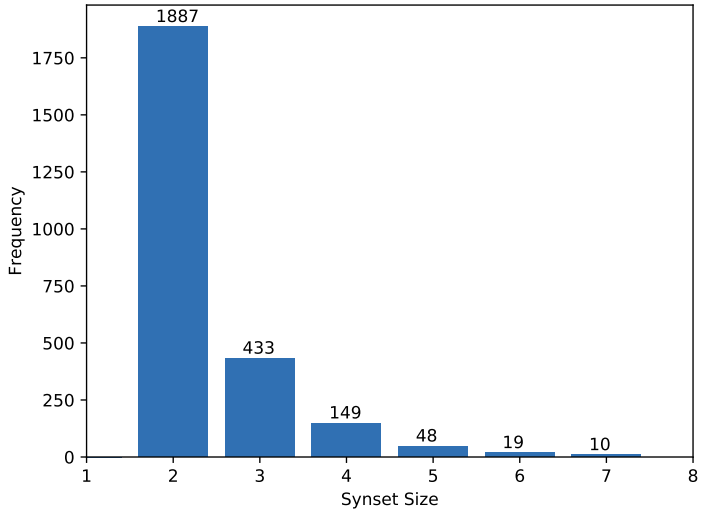

Figure 4.9.: Histogram of the synset sizes of the pre-processed tax law thesaurus.

4.6. Evaluation

The primary goal of this section is to evaluate different contemporary word embeddings algorithms and different approaches that utilize word embeddings algorithms and count-based DSMs to extend legal thesauri. The evaluation can be split into a quantitative and a qualitative part.

At first, evaluation measures are identified and compared. Secondly, hyper-parameter studies are conducted to investigate the impact of hyper-parameters of word embeddings algorithms and to identify suitable hyper-parameter configurations for the successive experiments. Afterward, the different approaches and word embeddings algorithms are evaluated quantitatively and qualitatively.

4.6.1. Evaluation Setup and Presentation Form

4.6.1.1. Quantitative Evaluation Setup

In the subsequent sections of this chapter, RP-Score, MAP and precision/recall curves relevance measures are used. For the RP-Score, smaller values are better, while for the MAP, larger values are better. For the precision/recall curves, a curve that is closer to the top right of a diagram is best. The detailed hyper-parameter configurations used in the subsequent chapters are listed in Appendix B.

Figure 4.10 shows an example for a quantitative evaluation using precision/recall relevance measure using three different evaluation thesaurus splits. The three evaluation thesauri are equal in their structure. The number of terms in the train- and test-subsets is unchanged due to the train/test splitting strategy. The three evaluation thesauri differ in the assignment of terms to train and test subsets. The very subtle changes of the curves in Figure 4.10 lead to the conclusion that multiple evaluation thesauri

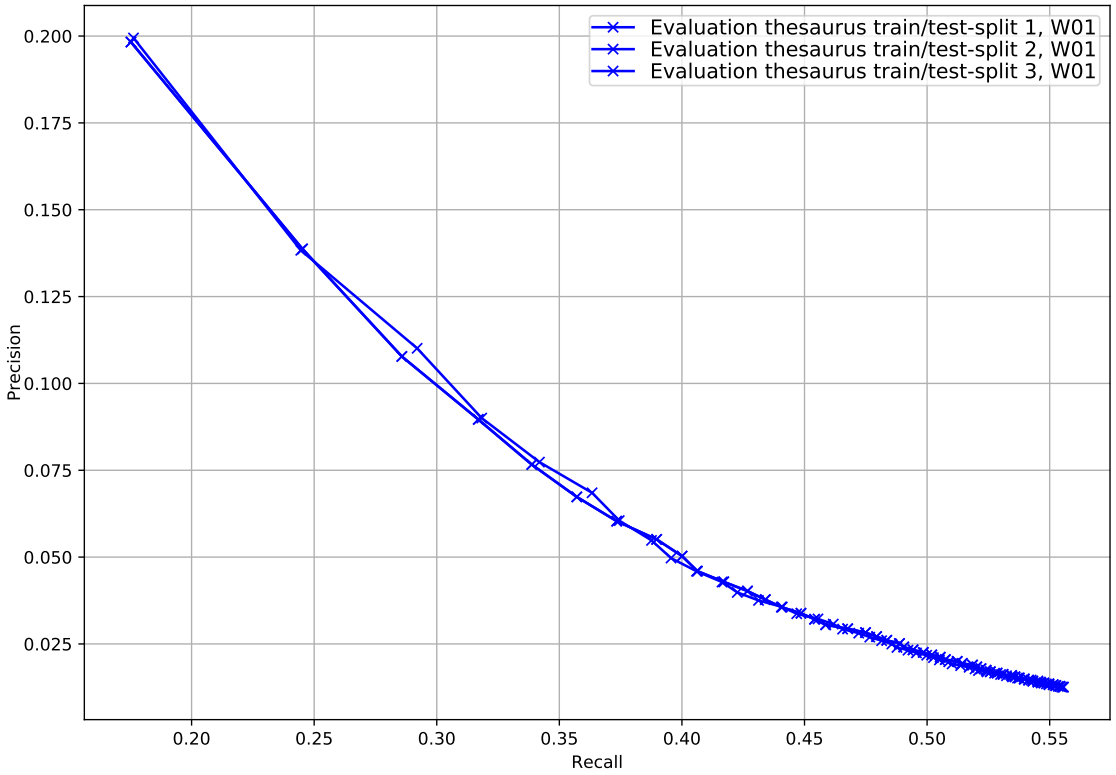

Figure 4.10.: Precision/recall curves for multiple evaluation thesaurus train/test splits.

splits are not necessary for precision/recall and the MAP relevance measures for the selected sampling strategy. Thus, only a single train-/test subset split is used in the remainder.

For the diagrams displayed in Sections 4.6.2 Evaluation Measures and 4.6.3 Hyper-parameter Study, the legend entries are ordered so that the order resembles the visual order from top to bottom where possible. The different word embeddings algorithms GloVe, word2vec and FastText are indicated with colors blue, orange and green. Different markers and line styles are consistently used for the Min-Count and Model Architecture hyper-parameters.

Terms of the input thesaurus are removed from word embeddings models (or candidate term lists), except for the RP-Score evaluations, because terms are assigned to at most one synset (single-label problem).

4.6.1.2. Qualitative Evaluation Setup

The goal of the qualitative evaluation is to assess the quality of candidate terms suggested by the different thesaurus extension approaches and considered word embeddings algorithms. A qualitative evaluation is more complicated but required where a quantitative evaluation is not possible. Remember that a real-world thesaurus is never complete. For the qualitative evaluations, the full innput thesaurus synsets are used as input for the thesaurus extension approaches. Terms of the input thesaurus are

removed either before or at latest after the calculation of candidate term lists. For one qualitative evaluation, 30 synsets are selected randomly from the training corpus. The selected synsets are the input to the thesaurus extension approaches. To assure the best possible comparability of the results, the same randomly selected synsets from the given thesaurus are used where possible. The randomly selected synsets have been reviewed and adjusted to resemble the synset sizes histogram, see Figure 4.9. Further, examples for a selection of the use case challenges presented in Section 4.2 are included. By default, up to ten candidate terms for each selected synset are calculated. A fixed number of up to ten candidate terms has been chosen as a trade-off. For some synsets almost no useful candidates can be detected while for other synsets several dozen reasonable candidate terms exist. An empirical convergence analysis shows that 30 synsets and up to ten suggested terms lead to stable results.

Two human experts rated the candidate terms according to three categories:

1. **synonyms**: Synonyms are semantically highly related to the input synset and very good candidates for adding such terms to the existing thesaurus. In borderline cases the semantic scope given by the existing synset and the rater's judgment is decisive.
2. **semantically related**: Semantically related candidate terms still cover some semantic aspect of the existing synset but are too far from the scope of the synset according to the opinion of the rater.
3. **semantically unrelated**: All approaches also suggest candidate terms that make no sense at all or are semantically much too far from the scope of the given synset in the opinion of the rater.

To assure the validity of the qualitative evaluation, the stability of the results is evaluated upfront. Therefore, the percentage of the three categories, "synonyms", "semantically related" and "semantically unrelated" are plotted against an increasing number of evaluation synsets. As indicated by the results

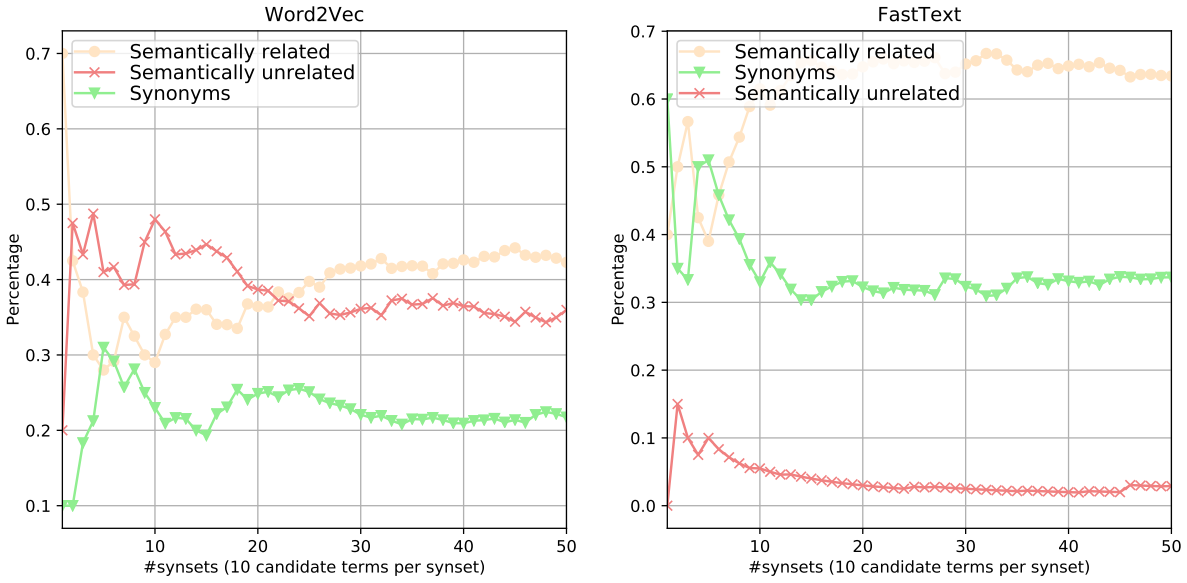

Figure 4.11.: Qualitative evaluation convergence analysis.

in Figure 4.11, the percentages assigned for the three different categories converge after around 25 synsets. Hyper-parameter configurations W02 and F01 are used. By default, 30 synsets are chosen as the default number of evaluation synsets for the qualitative evaluation. Some approaches suggest less than ten terms per evaluation synset, but the results are still representative because results are stable for 25 evaluation synsets.

The Inter-rater reliability is measured with Cohen's kappa, see Section 3.7.2. Note that the Kappa value for the inter-rater reliability for the qualitative evaluation displayed in Figure 4.11 is 0.5. The comparably low number serves as a lower bound for the expected convergence rate. For qualitative evaluations with larger Kappa values, even faster convergence of the percentage values can be expected.

The results of the qualitative evaluations are presented as stacked bar diagrams where the bars indicate the percentage of ratings per category. The plotted results include the aggregated ratings of both raters. The diagrams further display Cohen's kappa score κ and the total number of rated candidate terms N[1].

Where possible, the evaluation synsets are shared among the qualitative evaluations of the different approaches for better comparison. However, the results of the sensitivity analysis in Section 4.6.8 suggest that the setup of the qualitative evaluations allows a comparison despite different evaluation synsets and smaller changes of the hyper-parameters of the word embeddings algorithms. Shared evaluation synsets further enable an analysis of the number of shared results among the different approaches and word embeddings algorithms.

Where appropriate, selected example candidate term lists are presented to provide an intuition of the different types and the quality of the results. The example terms further serve as another step to explain the differences in the results of the qualitative evaluation. The example terms are collected in Appendix D. The presentation of the example terms is designed to show the input and output of the different approaches. The average rating of the perceived synonymy per candidate term indicates the perceived synonymy by two raters. The average rating of perceived synonymy ranges between one (synonyms) and zero (semantically unrelated).

4.6.2. Evaluation Measures

To date, the quality of word embeddings models can only be assessed indirectly via downstream NLP tasks. The NLP downstream task at hand is the extension of legal thesauri given a training corpus and an existing thesaurus. In contrast to other downstream NLP tasks like paraphrase detection or sentiment analysis, thesaurus extension can be achieved by directly accessing word embeddings models. For other downstream NLP tasks, word embeddings are often one ingredient among several other NLP technologies. The thesaurus extension task can be seen as an information retrieval task. For a given synset, one or several candidate term lists are calculated. Standard relevance measures used for evaluations in information retrieval are precision/recall and derivatives. For the hyper-parameter study, precision/recall curves are not suitable because for every word embeddings model, a precision/recall curve is plotted. This becomes confusing for hyper-parameter studies where many word embeddings models need to be investigated. More aggregated relevance measures need to be considered. The F_1

[1] By default, $N = 300$ = 30 (synsets) * 10 (candidate terms). For two reviewers this amounts to 600 ratings for one qualitative evaluation.

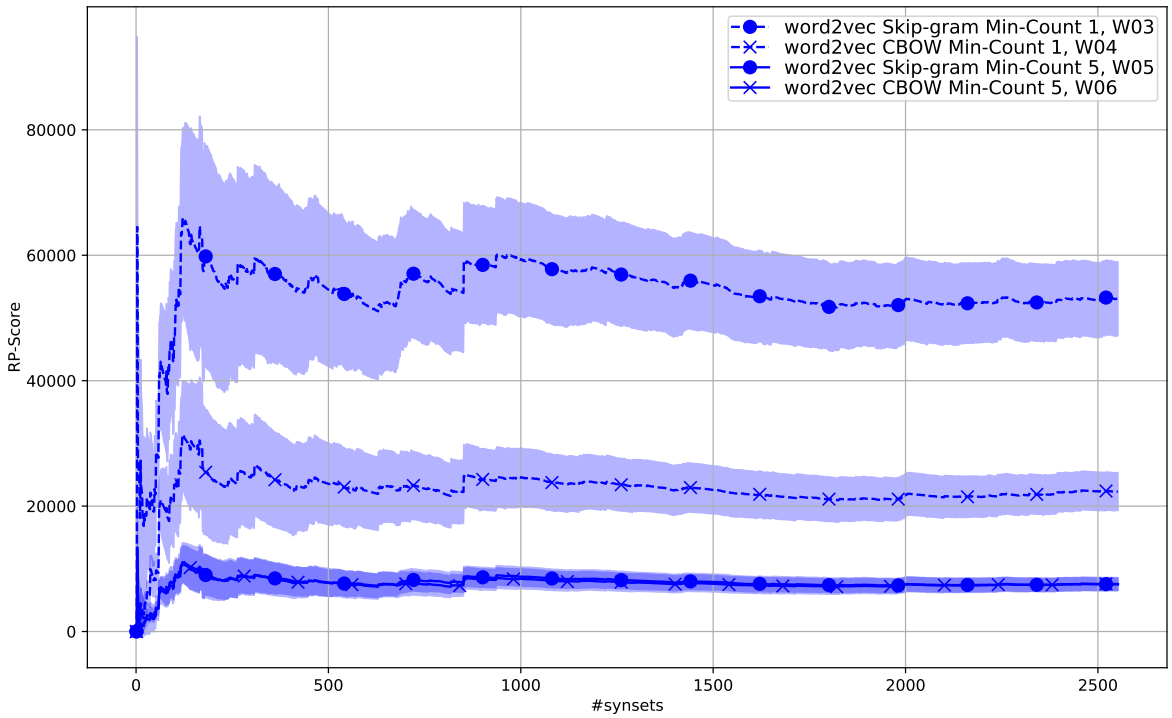

Figure 4.12.: RP-Score relevance measure convergence analysis.

score would also lead to a curve for every word embeddings model. A more suitable relevance measure is MAP where a single value scores each word embeddings model. In this section, the RP-Score is introduced that serves as an alternative or complementary relevance measure to the MAP and is particularly well suited for the use case at hand. The investigations show that RP-Score and MAP show qualitatively similar results.

The RP-Score is a straightforward, easy to use and indicative measure to conduct hyper-parameter studies for word embeddings based thesaurus extension. Both MAP and RP-Score require an existing thesaurus. While the RP-Score is especially useful for the use case at hand, it can be used for the evaluation of other information retrieval tasks, too. The intuition of the RP-Score is that better word embeddings models should result in smaller angles among synonymous terms in word embeddings model's vector space. For a query term or synset vector, all other tokens of the vocabulary of the word embeddings model are ranked using cosine similarity. Most synonymous words are expected to have a small angle and hence appear in the top-ranks of a ranked list of tokens. The RP-Score measures the ranking position distance among two terms that are known to be synonyms from the existing thesaurus.

Mathematically, the RP-Score is defined as follows:

$$RP - Score := \sum_{s \in synsets} \frac{\sum_{w_1, w_2 \in s, w_1 \neq w_2} RP_{w_1}(w_2)}{|s|(|s|-1)}$$

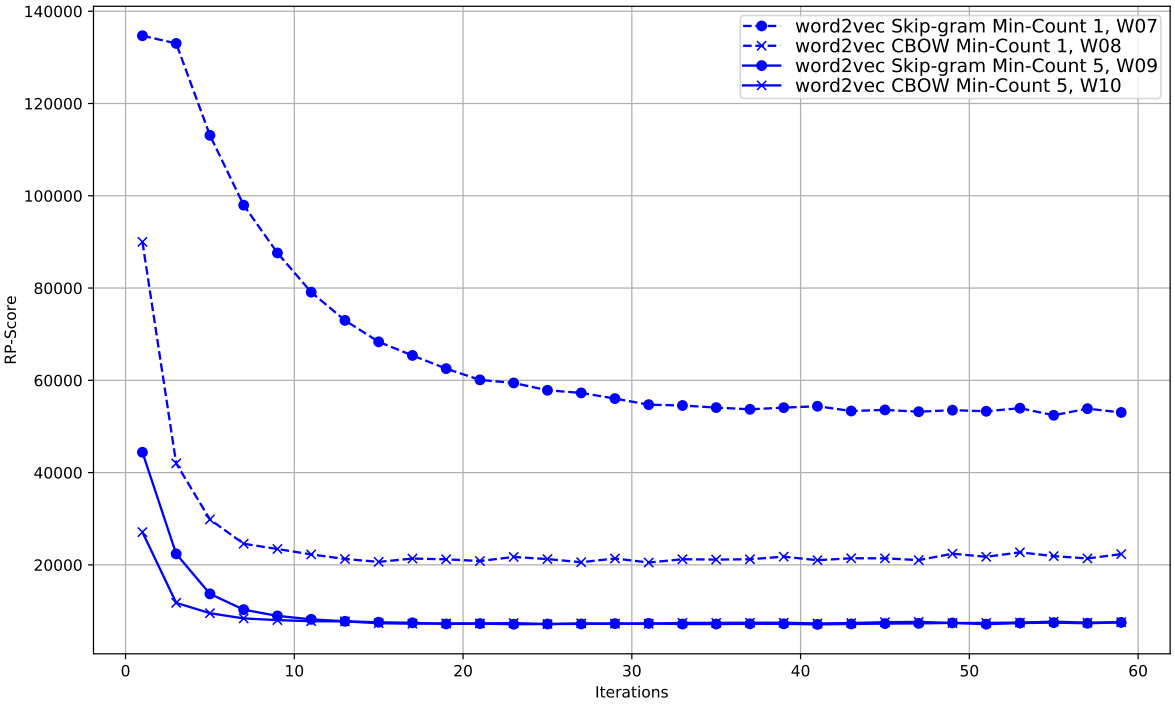

Figure 4.13.: RP-Score hyper-parameter study for the word2vec hyper-parameters Iterations, Min-Count and Model Architecture.

where $RP_{w_1}(w_2)$ is the ranking position distance between the query word w_1 and the related word w_2. Query and related word are in a synonym relationship according to an existing synset in a thesaurus. Note that the ranking position distance is not symmetric, i.e., in general, $RP_{w_1}(w_2) \neq RP_{w_2}(w_1)$[2]. The RP-Score is averaged by the number of possible relations between any two words in a synset. The resulting synsets RP-Scores can be averaged among a chosen number of synsets, for example, all synsets in a given thesaurus.

When two word embeddings models are compared, a smaller RP-Score is better. The RP-Score is not suited to compare results among different thesauri but gives a human-understandable indicator of how far the average ranking position distance among related tokens is. This helps to estimate an appropriate size of fixed length candidate term lists.

The RP-Score and MAP relevance measures are different but still related. Precision/Recall and derived relevance measures like MAP require a train/test split of the thesaurus. A thesaurus cannot be split into groups of synsets, but each synset needs to be split into training and test subsets. Figure 4.9 shows that the largest fraction of synsets consists of only two terms. Most synsets would be split into training and test subsets consisting of single terms. The RP-Score does not require splits into train/test subsets but investigates all possible relations of any two terms in a synset. Many synsets are composed of two terms. In this case, two relations are evaluated. The MAP would typically be carried out with the Synset Vector Approach. The MAP could be set up with a Single Word Approach so that for each term

[2]The cosine similarity among two terms is equal but the ranking position distances are not symmetric.

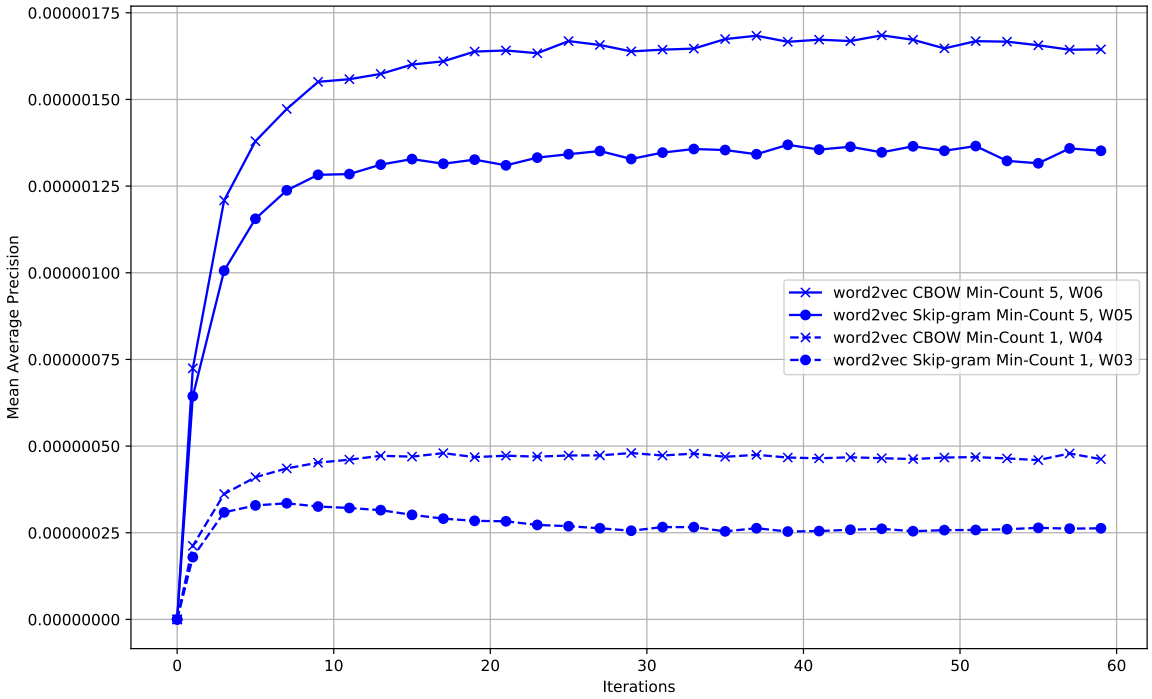

Figure 4.14.: MAP hyper-parameter study for the word2vec hyper-parameters Iterations, Min-Count and Model Architecture.

in a input synset, the ranking positions of all terms in the test synset are considered. If all possible combinations of train/test splits (once for each synset) are considered, then the MAP would be similar to the RP-Score in terms of the number of considered relations. There are further differences among the RP-Score and MAP relevance measures. The MAP is designed to favor correct results of the top-ranked results. The MAP applies a weighting of ranking positions that is based on a power series. In contrast to that, the RP-Score applies a linear weighting. The MAP is a relevance measure that is normalized to unit length. The normalization fosters comparability of results across, for example, different datasets. However, this is not a requirement for hyper-parameter studies. The RP-Score is not normalized by default and provides a more human-understandable intuition of the distribution of the expected terms in the ranked result lists.

An important aspect for the hyper-parameter studies is the robustness of the results obtained through relevance measures. In Figure 4.12, the RP-Scores and the standard deviation of the RP-Scores are plotted against an increasing number of relations of the existing thesaurus. The relations are appended per synset, i.e., all relations of one synset are appended per increment on the x-axis. For the first few synsets, the standard deviation starts to grow fast but converges for a larger number of synsets. The word embeddings models used here are comparably stable (40 iterations). While large synsets (with exponentially more relations) can still shift the RP-Score by almost 10.000 for less than 1.000 synsets, the RP-Score is robust when more than 2.000 synsets are used. In the following, evaluation thesauri with 2500 synsets are used, to assure as reliable results as possible for the hyper-parameter studies.

	MAP	RP-Score
word2vec CBOW Min-Count 1 (W04)	310 minutes	4011 minutes
word2vec CBOW Min-Count 5 (W06)	108 minutes	1209 minutes

Table 4.3.: Runtime of evaluation measures.

A direct comparison of the convergence of RP-Score and MAP with an equal number of considered relations is difficult to set up because of the different constructions of the relevance measures.

A more in-depth inspection of the results presented in Figure 4.12 reveals that even for word embeddings models that reached convergence (CBOW model architecture and Min-Count 5), several relations of the evaluation thesaurus have a ranking position distance larger than 150.000, i.e., ranking position distances of almost the size of the vocabulary. For example, the terms "fehlerberichtigung" ("error correction") and "fehlerbehebung" ("error rectification") have a ranking position difference of 170272 and 185865 respectively. This already indicates that full automation using word embeddings based approaches is difficult to achieve.

Figure 4.13 shows the results of a hyper-parameter study for the word2vec word embeddings algorithm for different hyper-parameter configurations. In the beginning, the RP-Score drastically drops with every additional iteration of the word2vec algorithm. At around 40 iterations, the RP-Score seems to reach a convergence state.

A qualitative comparison of the RP-Score and MAP relevance measures is possible by comparing Figures 4.13 and 4.14. For both Figures, the word embeddings models are calculated with the Synset Vector Approach and the same hyper-parameter configurations for the word2vec algorithm. A visual comparison of the curves displayed in the two figures shows that RP-Score and MAP show a very similar qualitative behaviour[3][4].

The MAP curves of the two model architectures CBOW and Skip-gram (Min-Count hyper-parameter set to five) differ much more than the RP-Score curves for the two model architectures. An explanation for this "contradiction" is that the MAP gives a significantly higher weight to the very first ranking positions. If the first few ranking positions are considered, then the CBOW model architecture is better than the Skip-gram model architecture.

Table 4.3 reflects the computational effort required for the calculation of MAP and RP-Score. The most expensive operation is to calculate the cosine similarity among word embeddings vectors. The RP-Score is not computationally more efficient than the MAP relevance measure. However, because no train-/test-set splitting is required, more relations can be considered[5]. This is beneficial for small evaluation thesauri.

The main takeaway of this section is that the RP-Score is an alternative or complementary relevance measure to MAP. The RP-Score is well suited for the use case of thesaurus extension and shows qualitatively similar behavior to the MAP relevance measure. The RP-Score is particularly useful for smaller evaluation thesauri because more relations than in standard MAP setups can be exploited. For the

[3]Remember that smaller RP-Score values indicate better results while for the MAP, larger values indicate better results. However, for a hyper-parameter study, a stronger weighting of the top-ranked results is not required.

[4]Equation 3.8 is used for the calculation of the MAP, i.e., only suggested terms are considered

[5]At least in a simpler way, i.e., when using only one train-/test subsets split to calculate MAP.

GloVe	**word2vec**	**FastText**
Symmetric	Model Architecture*	Model Architecture*
Iterations*	Iterations*	Iterations*
Min-Count*	Min-Count*	Min-Count
Window Size*	Window Size*	Window Size*
Vector Size*	Vector Size*	Vector Size*
Max-Vocab	Negative Samples*	Negative Samples*
Distance Weighting	Sample Threshold*	Sample Threshold*
Alpha	Alpha	Alpha
Eta	Hierarchical Softmax	MinN*
X-Max		MaxN*
		LrUpdateRate

Table 4.4.: Hyper-parameters of word embeddings algorithms that affect the quality of word embeddings models.

given dataset, less than 2000 synsets lead to stable RP-Score evaluations. While not required, for the subsequent evaluations, still all 2500 synsets of the evaluation thesaurus are used to assure the best possible quality of the results. The RP-Score analysis further already shows that word embeddings based approaches to thesaurus extension are too noisy to reproduce a gold standard thesaurus. Even for good word embeddings models, many synonymous terms according to the evaluation thesaurus have a large ranking position distance in the ranked result lists of all tokens.

4.6.3. Hyper-parameter Study

The word embeddings algorithms considered in this thesis come with many hyper-parameters. The primary goal of the hyper-parameter studies is to get insights on the impact of the different hyper-parameters on the quality of the results that can be achieved for legal thesaurus extension. The different approaches for thesaurus extension are subject to additional hyper-parameters that are investigated later. A secondary goal of this section is to identify proper hyper-parameter configurations to compare the word embeddings algorithms and the different approaches later on.

The primary goal is not to identify optimal hyper-parameter configurations. Identifying optimal hyper-parameter configurations is challenging. First, there are many hyper-parameters and due to the curse of dimensionality, a brute-force approach is not feasible. Several hyper-parameter optimization strategies are reported in the literature. The brute-force approach, also known as grid search, tests all combinations of hyper-parameter values. The down-side of grid search is the computational effort required to carry out grid search. A good balance between the quality of results and computational effort is often provided by random search. Several variants of random search have been reported in the literature, see, for example, Bergstra and Bengio (2012). However, all strategies that test for combinations of hyper-parameters require a validation set. The evaluation of a single hyper-parameter configuration is expensive because several hundreds of synsets are required to obtain robust estimates of the quality of the results, cf. Section 4.6.2. Hence, the random search is still computationally costly. Moreover, many hyper-parameters of the word embeddings algorithms show linear or convex behav-

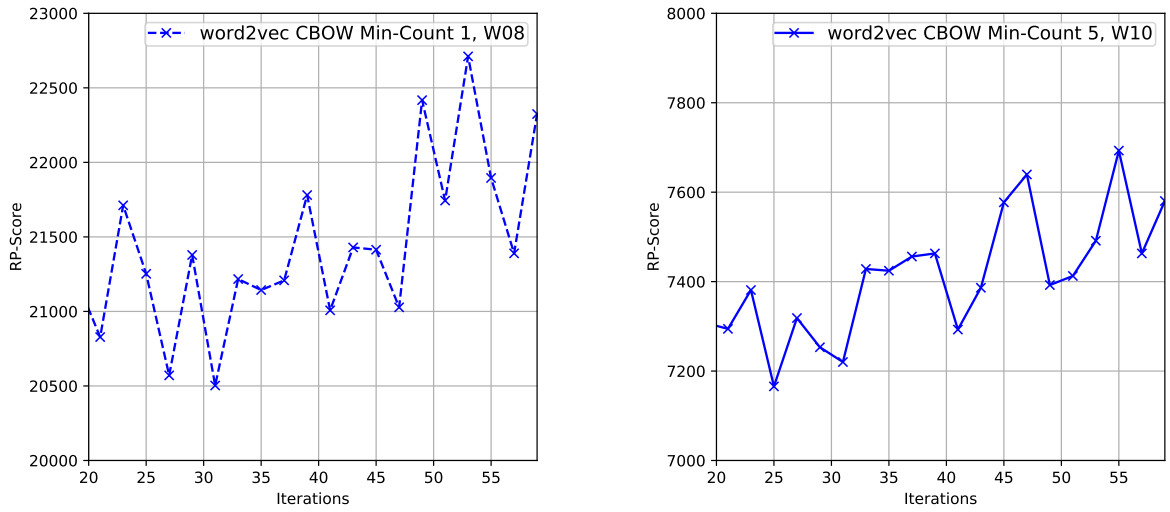

Figure 4.15.: Zoomed hyper-parameter study for the word2vec algorithm's Iterations hyper-parameter.

ior. Thus, the hyper-parameter study concentrates on individual hyper-parameters and assumes that best individual hyper-parameter selections do not show a disproportionate impact on combinations of hyper-parameters.

The hyper-parameters for the word embeddings algorithms GloVe, word2vec and FastText are introduced in Chapter 3. Table 4.4 shows an overview of the different hyper-parameters. Investigated hyper-parameters in this chapter are marked with an asterisk. Shared hyper-parameters among all word embeddings algorithms are indicted with a dark grey cell background. Hyper-parameters shared only among the word2vec and FastText toolkits are indicated with a light grey cell background.

Not all hyper-parameters that potentially affect the quality of the resulting word embeddings models are investigated. Mikolov et al. (2013c) notes that hyper-parameter selection is a task-specific decision. Mikolov et al. (2013c) identifies the Model Architecture, Vector Size, Window Size and Sample Threshold as the most crucial hyper-parameters. The Iterations hyper-parameter was introduced at a later stage of the published toolkit: word2vec version 0.1c. In contrast to Mikolov et al. (2013c)[6], the Iterations hyper-parameter is also highly relevant, especially for training corpora with less than billions of tokens such as the dataset used here. Additionally, the Min-Count hyper-parameter has a strong effect on the resulting word embeddings vectors. In general, the Hierarchical Softmax alternative to Negative Sampling is inferior, as shown by Mikolov and others. The Alpha hyper-parameter of the word2vec algorithm is a standard hyper-parameter of artificial neural networks and should not substantially affect the quality of word embeddings models unless chosen in extreme ranges. For the GloVe algorithm, similar to the word2vec algorithm, the Iterations, Min-Count, Window Size and Vector Size hyper-parameters can be expected to be most sensitive to a particular dataset and task. Therefore, the Distance Weighting, Alpha, Eta and X-Max hyper-parameters are not investigated in the hyper-parameter study. The FastText algorithm shares most hyper-parameters with the word2vec algorithm.

[6]The Iterations hyper-parameter was not included in the word2vec toolkit at the time of publication.

The ranges of the character n-grams, MinN/MaxN, are the most important additional hyper-parameters of the FastText algorithm.

A more detailed analysis of the Min-Count hyper-parameter is complicated because of the out-of-vocabulary problem. Either a single evaluation thesaurus with a high value of the Min-Count hyper-parameter is used for every Min-Count value or an individual evaluation thesaurus needs to be calculated. Both options reduce the comparability and generalizability of the results. Thus, only Min-Count hyper-parameter values of one and five are investigated.

4.6.3.1. Hyper-parameter Study for the word2vec Algorithm

The word2vec algorithm comes with a bunch of hyper-parameters and the Model Architecture, Vector Size, Window Size and Sample Threshold hyper-parameters have been identified as the most important hyper-parameters by the authors of the word2vec algorithm (Mikolov et al. (2013c)). Figure 4.13 displays the results of the study of the Iterations, Min-Count and Model Architecture hyper-parameters for the word2vec algorithm. Especially for smaller training corpora, the Iterations hyper-parameter has a strong impact on the quality of the resulting word embeddings models. All investigated config-

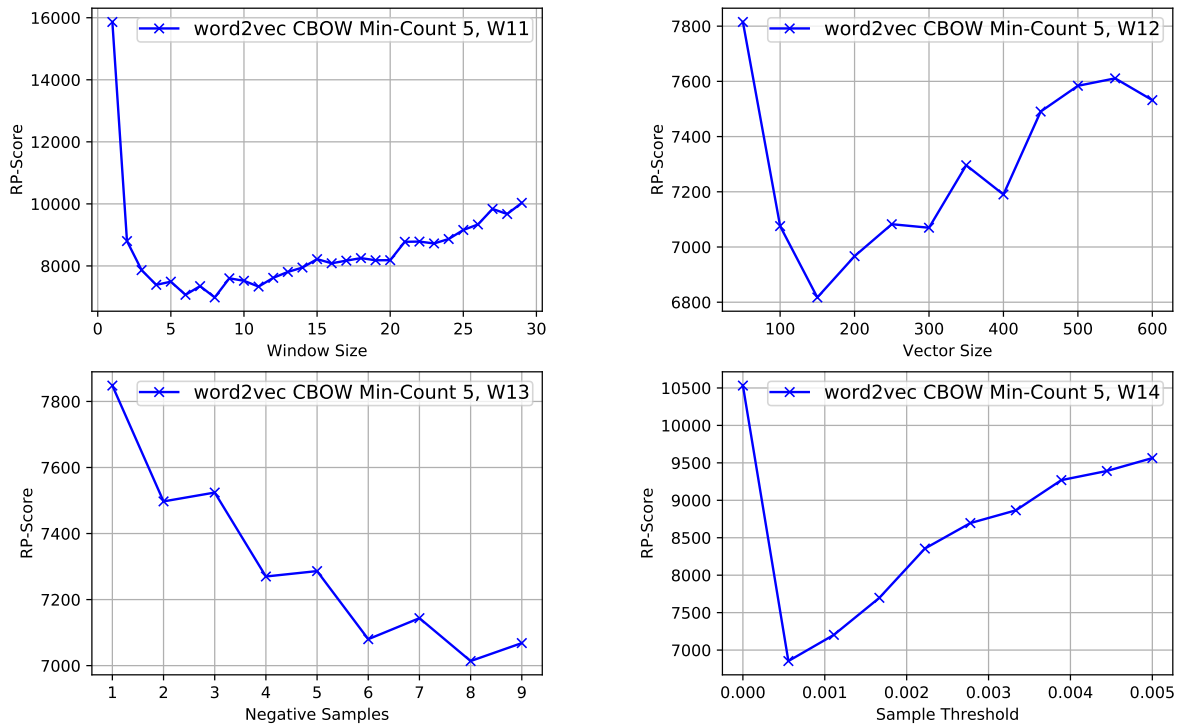

Figure 4.16.: Hyper-parameter study for the word2vec algorithm's hyper-parameters Window Size, Vector Size, Negative Samples and Sample Threshold.

urations displayed in Figure 4.13 seem to reach convergence. The Model Architecture and Min-Count hyper-parameters can have a strong effect, too. The law of great numbers states that a larger number of experiments increases the significance of results. Likewise, the quality of models based on the distributional hypothesis depends on the number of training samples for each term. More precise, the quality of a word embeddings vector of a single token depends on the number of training samples for that token.

The prevailing hypothesis for word embeddings is that larger training corpora lead to a higher quality of word embeddings models. The Iterations hyper-parameter mitigates the requirement of sufficient training samples to some degree. For smaller training corpora, increasing the Iterations hyper-parameter can lead to higher quality word embeddings models because it increases the number of training samples per token. Another option is to increase the Min-Count hyper-parameter that increases the minimum number of different training samples per token. The results of Figure 4.13 show that the RP-Score is better when the Min-Count hyper-parameter is set to five. This is not surprising.

Note that also for the hyper-parameter study of the Min-Count hyper-parameter set to one, the evaluation thesaurus is pre-processed with Min-Count hyper-parameter set to five. For the training corpus pre-processed with Min-Count set to one, the word embeddings model's vocabulary size would be increased. Consequently, the number of potential candidate terms would be significantly larger. It was shown in Landthaler et al. (2018c), that setting the Min-Count hyper-parameter of both, the word2vec algorithm and the pre-processing of the evaluation thesaurus leads to better results in terms of RP-Score. The more different training examples are available for tokens, the higher is the quality of the respective word embeddings vectors.

The CBOW model architecture leads to significantly better results for the Min-Count hyper-parameter set to one than the Skip-gram model architecture. When setting the Min-Count hyper-parameter to five, Skip-gram and CBOW model architectures result in similar RP-Scores, except for the faster convergence of the CBOW model architecture.

After approximately 40 training iterations, the word embeddings models trained with the word2vec algorithm can be expected to reach convergence.

The RP-Score curves (as well as the MAP curves) show properties of fractal dimensions. Fractal dimensions have been defined in the context of research on fractals, see Mandelbrot (1988) and Hausdorff (1918). On a global scale, the RP-Score curves appear smooth and convergence can be easily detected in Figure 4.13. A more detailed, local scale is depicted in Figure 4.15. The RP-Scores appear to vary in terms of multiples of hundred in areas where the curves appear smooth in Figure 4.13. When the Min-Count hyper-parameter is set to five, the variations in RP-Score are much smaller in the range of plus/minus 250. When the Min-Count hyper-parameter is set to one, the variations of the RP-Scores are significantly larger by a factor of roughly four times. Remember that also the vocabulary size is around four times larger for word2vec algorithm's Min-Count hyper-parameter set to one than for the Min-Count hyper-parameter set to five. These variations are, however, much smaller than the variations in terms of thousands visible in Figure 4.13. In Section 4.6.8, a sensitivity analysis takes a closer look at the impact of (smaller) hyper-parameter variations to the qualitative evaluations.

The results of the remaining hyper-parameter studies of the word2vec algorithm are depicted in Figure 4.16. On a local scale, all hyper-parameters appear to have a local optimum. The discrete hyper-parameter Window Size shows an optimum around six and eight tokens, which is close to the default

hyper-parameter value of five. The vector size shows a local optimum around 150, which is in contrast to reasonable values reported in the literature. In the literature, a decrease of the quality of word embeddings models starting from a vector size of 300 and larger is reported, cf. Pennington et al. (2014). This is likely due to the comparably small size of the training corpus. Remember the trade-off between information entropy and computational efficiency: when a certain vector size is reached, no further information is encoded in the word embeddings vectors.

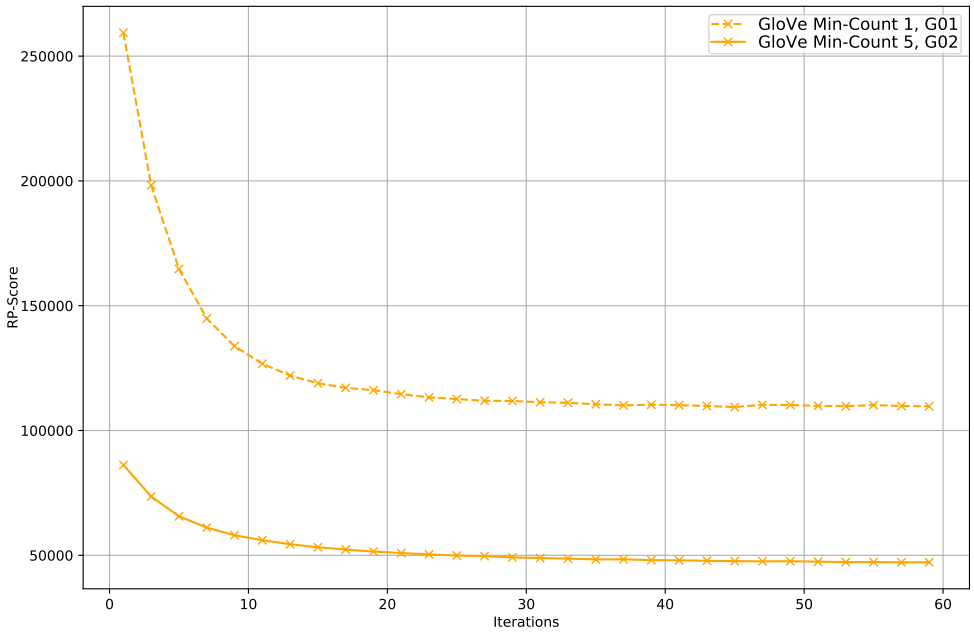

Figure 4.17.: Hyper-parameter study for the GloVe algorithm's hyper-parameters Iterations and Min-Count.

The Negative Samples hyper-parameter shows a local minimum of around eight negative samples. The negative samples cater to a balance between positive and negative samples. The negative samples are selected randomly from the training corpus vocabulary. A concrete value that balances fixed numbers of positive and negative examples can be expected depending on the number of positive examples, i.e., the corpus size.

The Sample Threshold hyper-parameter controls the down-sampling of frequently occurring tokens and therefore depends on the occurrence frequency of such tokens in a particular training corpus. While the grid of values considered in the hyper-parameter study is somewhat arbitrary, the effects of the hyper-parameter are also limited. An optimum is reached at a value of around 5e-4. This value is smaller by a factor of ca. ten in comparison to the default value.

The RP-Scores vary differently for the different hyper-parameters. For the Vector Size and Negative Samples hyper-parameters, the RP-Scores vary around 1.000, while for the Window Size and Sample Threshold hyper-parameters, the RP-Scores vary around 2.000 and 3.500 respectively. Thus, the Neg-

ative Samples and Vector Size hyper-parameters are less critical than the Window Size and Sample Threshold hyper-parameters for the dataset and the investigated use case.

4.6.3.2. Hyper-parameter Study for the GloVe Algorithm

The GloVe algorithm can be considered as a count-based DSM. The GloVe algorithm has fewer hyper-parameters than the word2ec and FastText algorithms. Figure 4.17 depicts the results of the hyper-parameter study for the Iterations and Min-Count hyper-parameters. It can be expected for all DSMs that a restriction to more frequently occurring tokens improves the quality of the results. Similar to the word2vec algorithm, a convergence of the GloVe algorithm is reached after around 40 iterations.

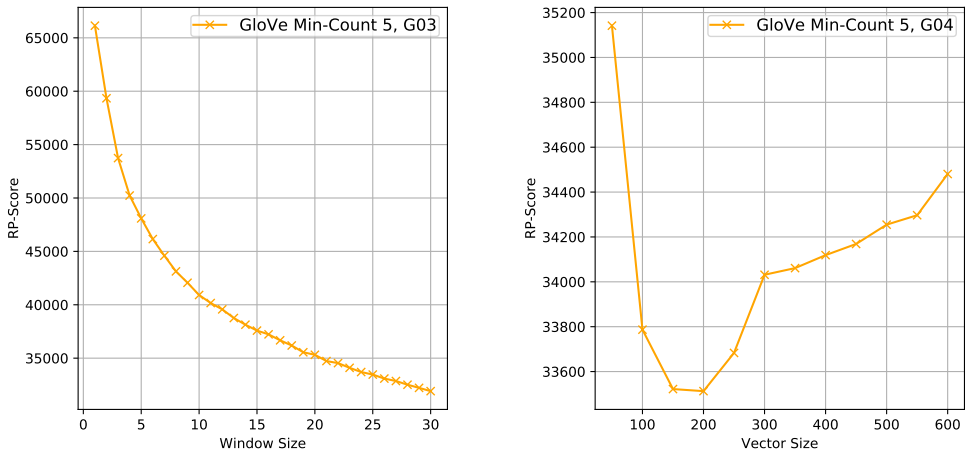

Figure 4.18.: Hyper-parameter study for the GloVe algorithm's hyper-parameters Window Size and Vector Sizes.

Figure 4.18 shows the results of the hyper-parameter studies of the Window Size and Vector Size hyper-parameters of the GloVe algorithm. The Window Size hyper-parameter study shows interesting results because the curve is almost convex, i.e., no optimum is hit in the investigated range of values. Remember that the GloVe algorithm uses a linear weight decay rather than a dynamic window size approach like the word2vec algorithm. For this hyper-parameter study, it remains unclear when an optimum will be reached. In contrast to the Window Size hyper-parameter study, the Vector Size hyper-parameter study highly resembles the word2vec algorithm's Vector Size hyper-parameter study that reaches a local optimum at around a vector size of 150 to 200, i.e., the amount of information encoded in the GloVe word embeddings vectors is roughly equal to word2vec word embeddings vectors.

4.6.3.3. Hyper-parameter Study for the FastText Algorithm

The FastText algorithm transfers the idea of the word2vec algorithm to character n-grams of tokens. The Iterations hyper-parameter study for the FastText algorithm depicted in Figure 4.19 reveals interesting results. For the CBOW model architecture, a more or less convex convergence of the RP-Score can be seen. However, for the Skip-gram model architecture, convergence is very fast, and a local

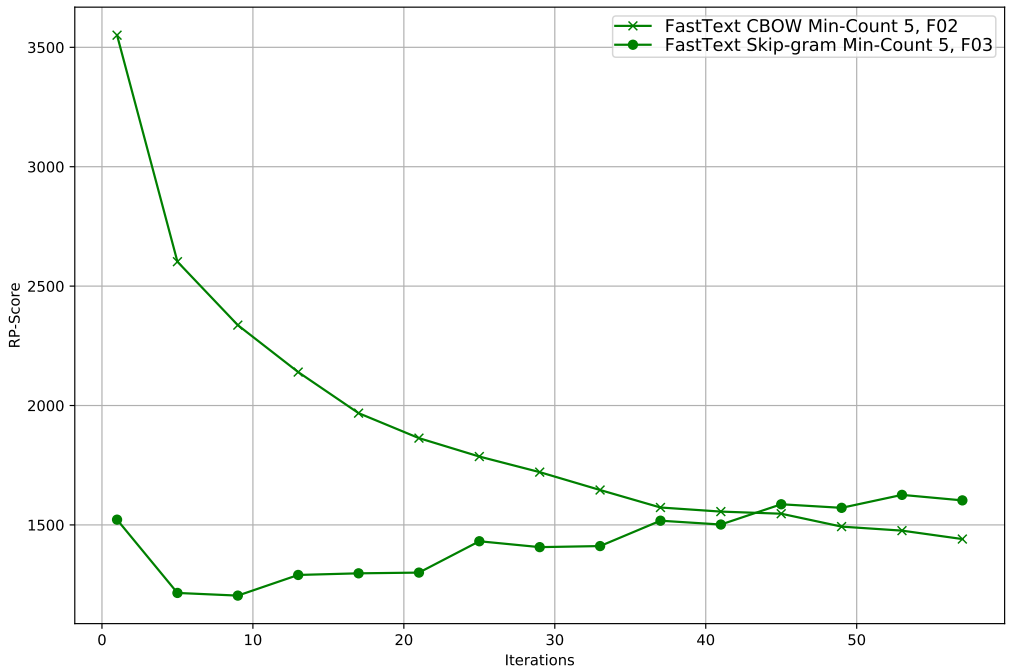

Figure 4.19.: Hyper-parameter study for the FastText algorithm's hyper-parameters Iterations, Min-Count and Model Architecture.

optimum is reached after around six iterations. Even more disturbing is that for the Skip-gram model architecture, the RP-Scores rise again after ten iterations and lead to worse RP-Scores than for the CBOW model architecture. The Skip-gram model architecture shows the best global results after few iterations. The fast convergence can be explained by the fact that much more character n-grams exist than tokens for the same training corpus. It remains unclear why the RP-Scores start to increase again but only for the Skip-gram model architecture. Eventually, the same effect occurs after even more iterations with the CBOW model architecture.

The results for the hyper-parameters Window Size, Vector Size, Negative Samples and Sample Threshold of the FastText algorithm are displayed in Figure 4.21. A small window size gives the best results. Due to the many character n-grams present in tokens, a smaller window size benefits from fewer co-occurrence samples. The FastText algorithm also encodes more information. Hence, a larger vector size in comparison to the word2vec algorithm is beneficial. The impact of the Negative Samples and Sample Threshold hyper-parameters is very small or almost not present.

Two additional hyper-parameters of the FastText algorithm are the MinN and MaxN hyper-parameters. The MinN and MaxN hyper-parameters define the minimum and the maximum of the range of the size of character n-grams. In Figure 4.21, several combinations of MinN and MaxN values are investigated. Hyper-parameter configuration F08 is used. The combinations are sorted and colored by the range size (orange: 1, green: 2, red: 3, violet: 4, brown: 5). For a range size of one, the character four-grams perform best. This coincides with Schütze (1993)'s choice. However, the results obtained with larger range sizes are better. The best combination of (3,5) is close to the default values combination of (3,6).

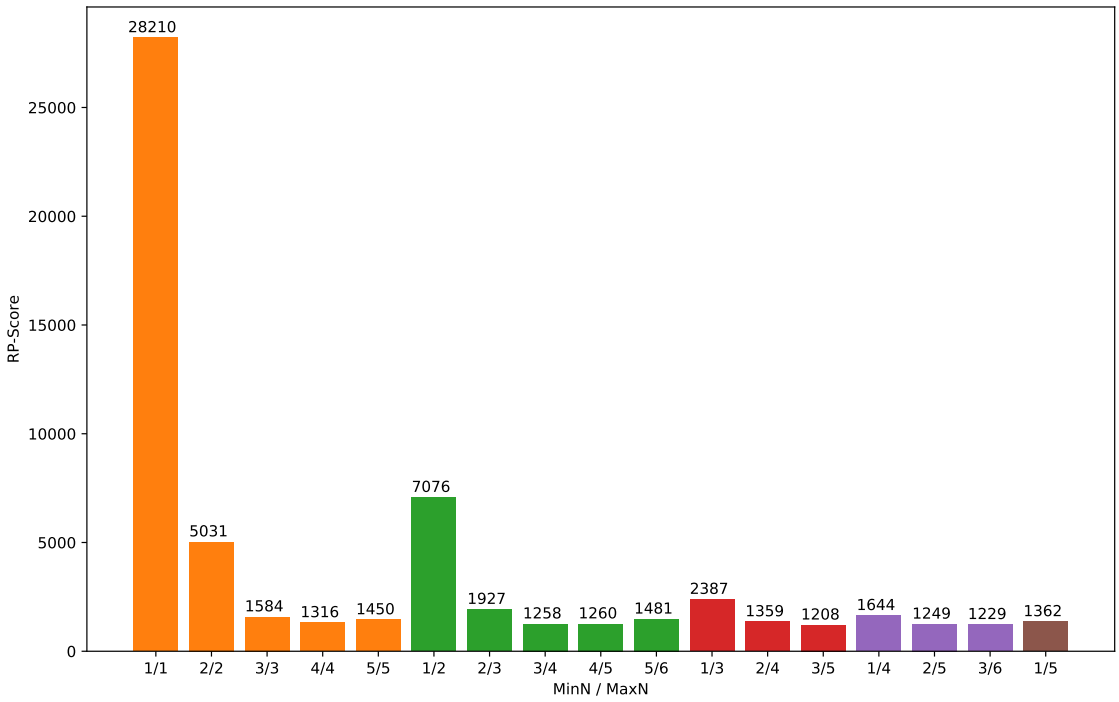

Figure 4.20.: Hyper-parameter study for the FastText algorithm's hyper-parameters MinN and MaxN.

4.6.3.4. Comparison of Hyper-parameter Studies

Next, the hyper-parameter studies of the different word embeddings algorithms are compared. Figure 4.22 sets the shared hyper-parameters Iterations and Min-Count (and Model Architecture for word2vec and FastText algorithms) of the different word embeddings algorithms into a comparison on a more global scale. In general, the GloVe algorithm leads to worse results than the word2vec algorithm. FastText algorithm's hyper-parameter configurations perform best when compared to the hyper-parameter configurations of the GloVe and word2vec algorithms. An interesting aspect is that the FastText algorithm shows very little convergence behavior in comparison to the GloVe and word2vec algorithms. The Min-Count hyper-parameter still shows a strong effect. The best GloVe word embeddings models for the Min-Count hyper-parameter set to five slightly lead to better results than for the worse performing Skip-gram model architecture of the word2vec algorithm. Nevertheless, both the CBOW and Skip-gram model architectures of the word2vec algorithm significantly outperform the GloVe algorithm when the Min-Count hyper-parameter is set to five. The results of the CBOW model architecture of the word2vec algorithm when setting the Min-Count hyper-parameter to five is close to results of the FastText algorithm.

Figure 4.23 shows that the locally strongly varying results of the different hyper-parameters of different word embeddings algorithms do not vary that much when set into a more global comparison. In fact, except for the Window Size hyper-parameter (especially for the GloVe algorithm), the hyper-parameters show an almost linear behavior. The results suggest that the GloVe algorithm heavily depends on the selection of the Window Size hyper-parameter. The other hyper-parameters Vector

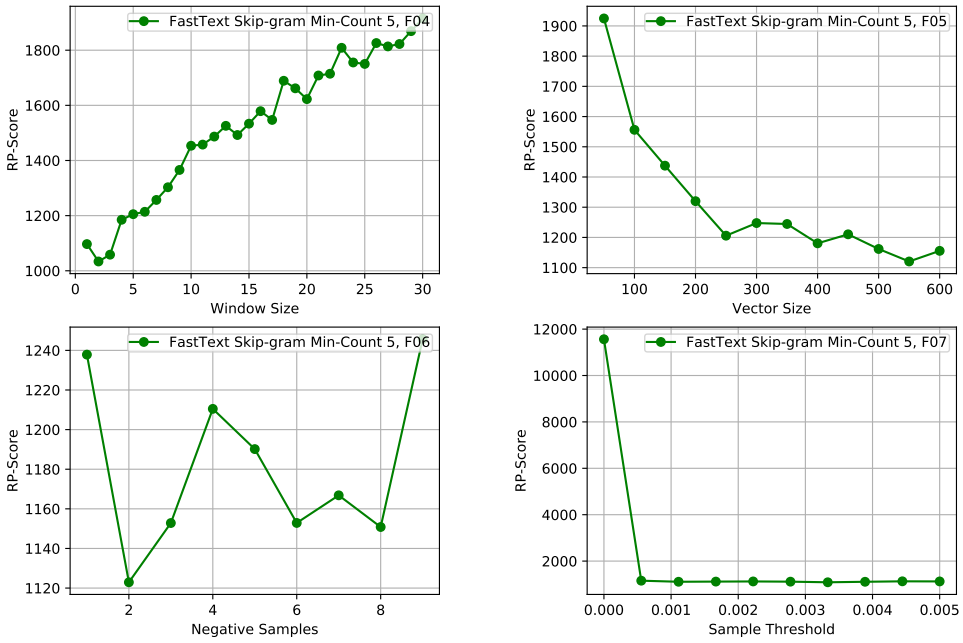

Figure 4.21.: Hyper-parameter study for the word2vec algorithm's hyper-parameters Window Size, Vector Size, Negative Samples and Sample Threshold.

Size, Negative Samples and Sample Threshold have only a small impact on the quality of the resulting word embeddings models for all word embeddings algorithms. For the FastText algorithm, on this more global scale, the impact of the hyper-parameters appears minimal. Merely the Sample Threshold hyper-parameter of the FastText algorithm should be set to a value larger than zero.

4.6.3.5. Runtime Analysis

An important aspect for the feasibility to apply word embeddings is the computational effort necessary to compute word embeddings models of high quality. The "real" timings of bash's "time" command-line tool are used to assess the computational effort to calculate word embeddings models. Figures 4.24 and 4.25 illustrate the runtime of the different hyper-parameter studies presented so far. For most hyper-parameters, a linear increase of the values of the hyper-parameter leads to a linear increase of the runtime, especially for the word2vec and FastText algorithms. A linear increase is beneficial (for example, in comparison to quadratic or exponential increases). Note that for reasons of visual presentation in Figure 4.24, only the first three to four data points for the FastText algorithm with CBOW and Skip-gram model architectures are visible.

Figure 4.24 displays the computational effort for the hyper-parameter studies for the Iterations, Min-Count and Model Architecture hyper-parameters for all three word embeddings algorithms. Again, only a few data points for the FastText algorithm are shown for better visualization. When the

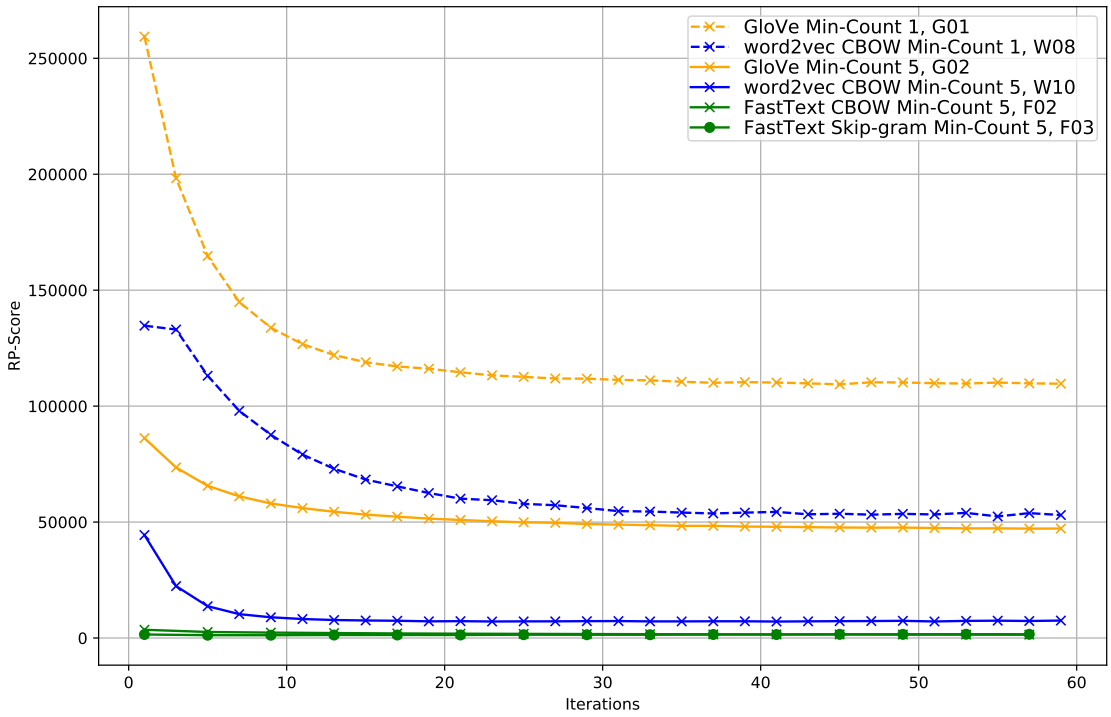

Figure 4.22.: Comparison of the hyper-parameter studies for the hyper-parameters Iterations, Min-Count and Model Architecture.

Min-Count hyper-parameters is set to five, then the word2vec algorithm's CBOW model architecture achieves better or equivalent quality with significantly less computational effort than the Skip-gram model architecture. The GloVe algorithm shows worse quality for the Min-Count hyper-parameter set to five with equivalent computational effort in comparison to the word2vec algorithm. The FastText algorithm shows the highest quality results at the price of much higher computational effort in comparison to the word2vec and GloVe algorithms.

Figure 4.25 displays the computational effort for the different hyper-parameter studies Window Size and Vector Size for the three word embeddings algorithms. Additionally, the runtime for the hyper-parameter studies for the Negative Samples and Sample Threshold hyper-parameters of the word2vec and FastText algorithms are displayed. The hyper-parameters show a more or less linear impact on the computational effort, except for GloVe algorithm's Window Size hyper-parameter that also shows extraordinary behavior in terms of quality. For the word2vec algorithm's Sample Threshold hyper-parameter, the greatest computational effort goes with the highest quality of word embeddings models. However, the computational effort induced by the Sample Threshold hyper-parameter is comparably small (around factor two in comparison to the default value). For the GloVe algorithm, the runtime of all command-line tools is aggregated per word embeddings model.

Figure 4.26 shows the full runtime plots for the FastText algorithm's hyper-parameters Window Size, Vector Size, Negative Samples and Sample Threshold. All hyper-parameters except the Sample Threshold hyper-parameter show a linear impact on runtime. Due to the comparably large computational

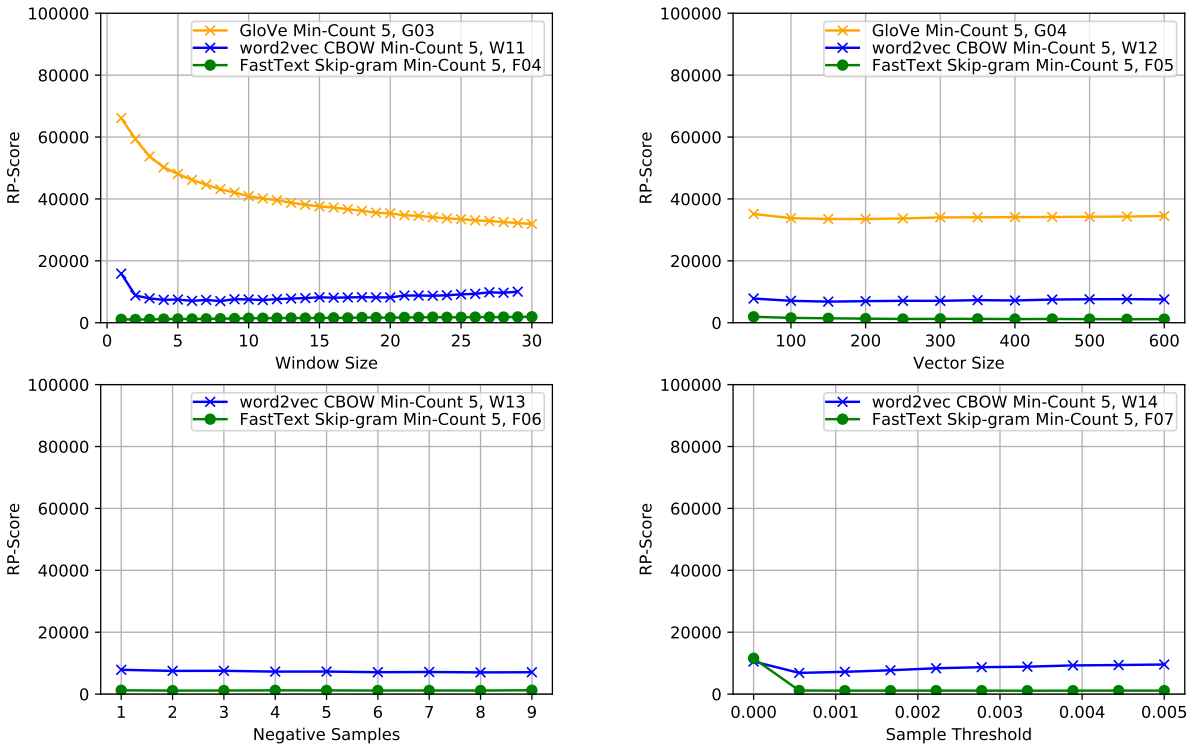

Figure 4.23.: Comparison of the hyper-parameter studies for the hyper-parameters Window Size, Vector Size, Negative Samples and Sample Threshold.

effort of the FastText algorithm, smaller values of the hyper-parameters are favorable. Despite the almost not existing impact of the Sample Threshold hyper-parameter on the quality of word embeddings models, the runtime slightly increases for larger values. Not using the Sample Threshold hyper-parameter by setting it to zero reduces runtime significantly. However, not using the Sample Threshold hyper-parameter is the only value in the examined range that reduces the quality of resulting word embeddings models. The difference in quality is significant. Thus, a small, non-zero value for the Sample Threshold hyper-parameter is recommended.

Runtime (minutes)	**Min-Count 1**		**Min-Count 5**	
vocab	0.36		0.36	
Window Size	**5**	**15**	**5**	**15**
coocur	1.54	3.61	1.46	3.46
shuffle	0.58	1.39	0.51	1.16
glove	149.16	156.25	60.33	124.48
Total	151.28	161.25	62.3	129.1

Table 4.5.: Runtime of GloVe command-line tools.

The impact on the runtime for the MinN and MaxN hyper-parameter combinations on the FastText

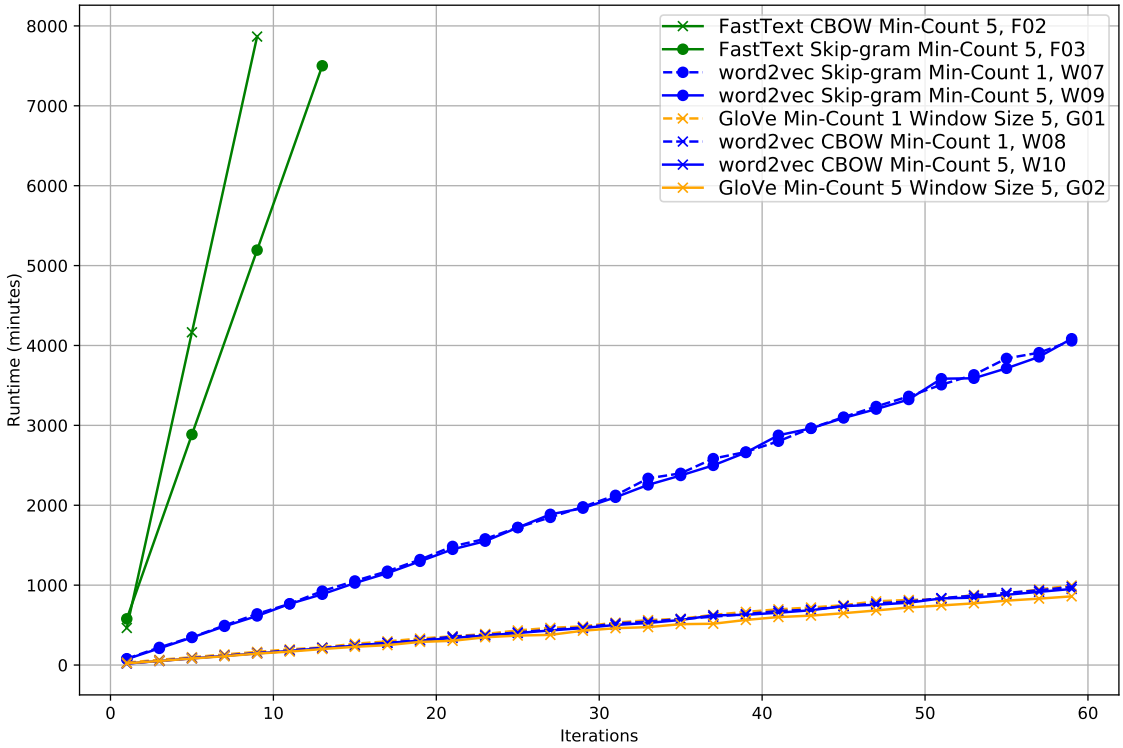

Figure 4.24.: Runtime of the hyper-parameter studies for the hyper-parameters Iterations, Min-Count and Model Architecture.

algorithm is shown in Figure 4.27. Equally to Figure 4.20, the combinations are sorted by range size (orange: 1, green: 2, red: 3, violet: 4, brown: 5). As can be expected, larger ranges of character n-grams require more runtime because more character n-grams need to be calculated. Larger character n-grams require less computational effort because the total number of larger character n-grams is smaller than the total number of smaller character n-grams. However, since the quality increases with larger range sizes, a large enough range should be selected. The default MinN/MaxN hyper-parameter combination of the FastText algorithm (3,6) is a good trade-off.

The implementation of the GloVe algorithm is split into four command-line tools. Table 4.5 displays the computational effort of each command-line tool for the Iterations hyper-parameter set to a value of 59. The "Glove" command-line tool consumes the most runtime by far. Given the very little savings of splitting the task of calculating word embeddings models, it does not make sense to re-use the results of the other command-line tools "vocab", "cooccur" and "shuffle". On the contrary, the split into four command-line tools makes hyper-parameter studies much more complicated and error-prone.

The word embeddings models have a size of around 500MB to two GB per word embeddings model. The word embeddings model size depends mostly on the vocabulary size (controlled via the Min-Count hyper-parameter) and the Vector Size hyper-parameter. The training corpus size also has an impact on the vocabulary size, but is given by the dataset.

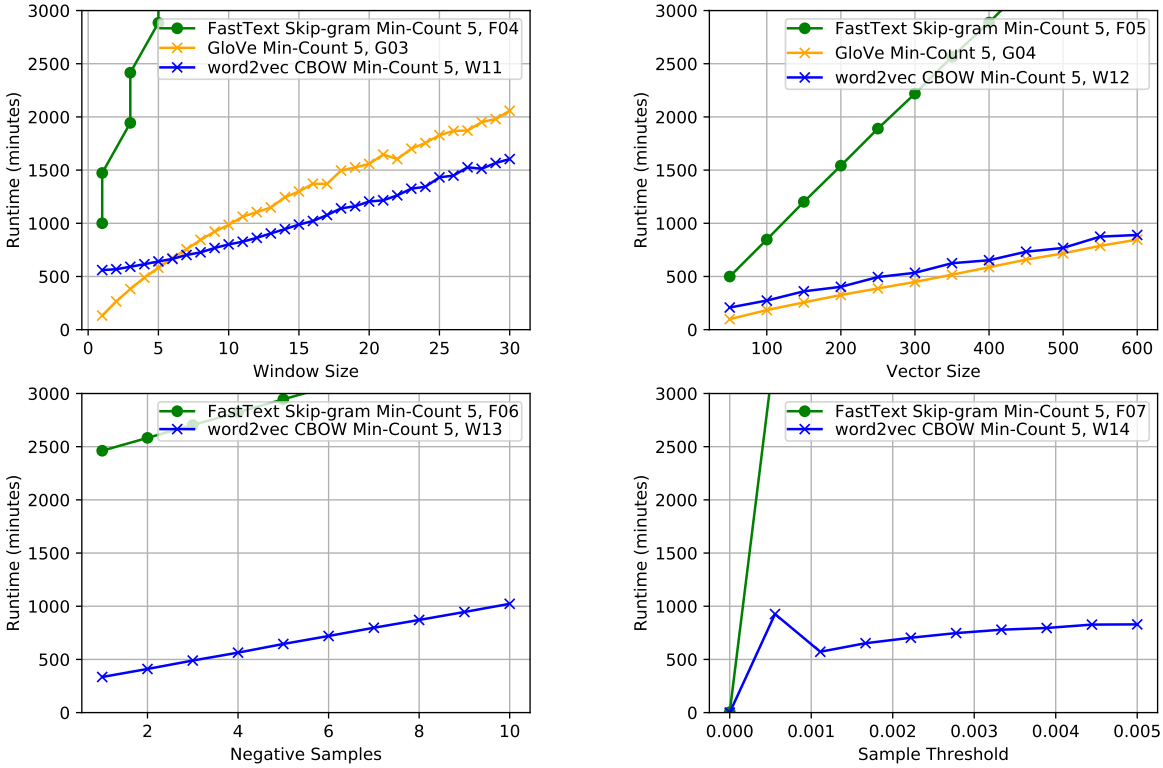

Figure 4.25.: Runtime of the hyper-parameter studies for the hyper-parameters Window Size, Vector Size, Negative Samples and Threshold.

4.6.3.6. Summary

In summary, the GloVe algorithm requires much more hyper-parameter tuning than the word2vec algorithm, i.e., the word2vec algorithm appears to be more robust to hyper-parameter changes. The computational effort for the GloVe and word2vec algorithms is comparable with slight benefits for the word2vec algorithm for proper hyper-parameter configurations. The FastText algorithm gives significantly better quantitative results in comparison to the GloVe and word2vec algorithms. However, this is due to the qualitative patterns present in the results that will be elaborated in Section 4.6.4 and comes with significantly higher computational costs.

For a comparison of the different algorithms, concrete hyper-parameter configurations have to be chosen. W15, G05, F08 hyper-parameter configurations, cf. Appendix, are selected by default in the following. B. For the GloVe algorithm, 40 iterations and a vector size of 150 are derived as suitable values from the results of the hyper-parameter study. The Window Size has been selected somewhat arbitrary with a value of 23[7]. The word2vec algorithm appears less susceptible to hyper-parameter changes. For

[7]The hyper-parameter study of the Window Size hyper-parameter of the GloVe algorithm shows that results get better for larger values of the Window Size hyper-parameter. On the other hand, computational effort increases. Due to a disk size overflow, the results of the hyper-parameter study were broken for values larger than 23. Hence at that time, a value of

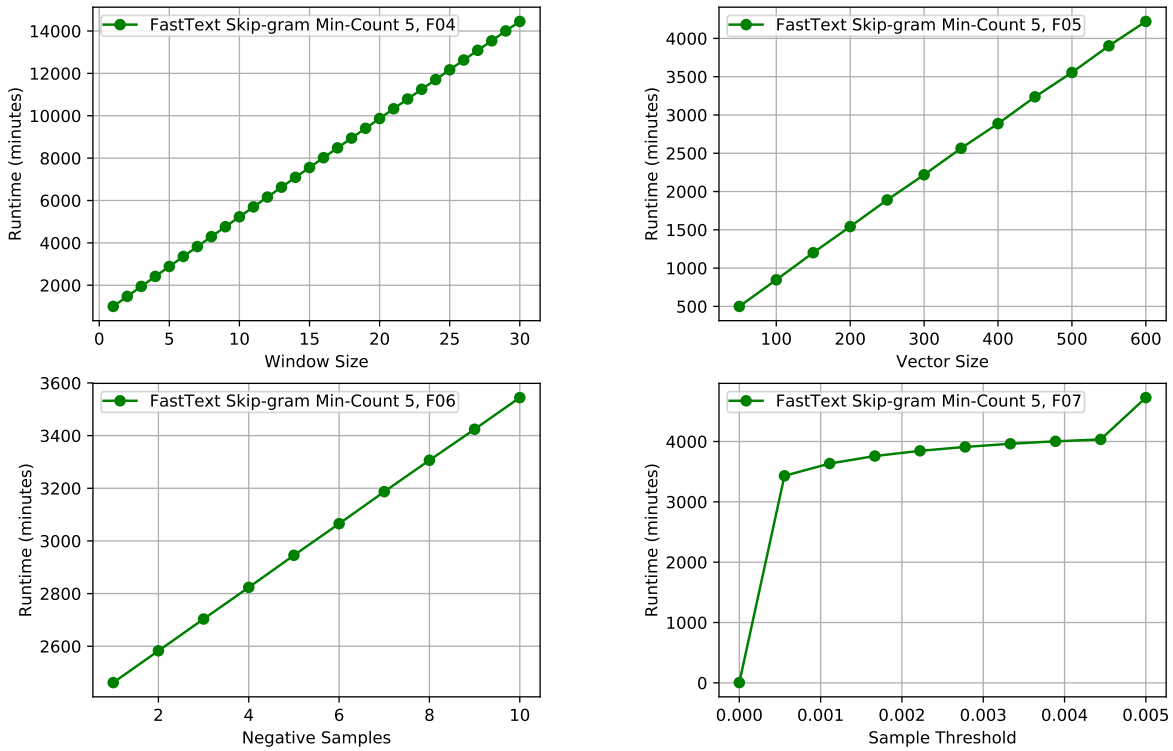

Figure 4.26.: Runtime of the hyper-parameter studies for FastText algorithm's hyper-parameters Window Size, Vector Size, Negative Samples and Threshold.

the Sample Threshold hyper-parameter, even better choices may exist between the bases of the grid of the hyper-parameter study. However, the impact of the Sample Threshold hyper-parameter are comparably small. The FastText algorithm also appears less susceptible to hyper-parameter changes than the GloVe algorithm. The Skip-gram model architecture with 60 iterations[8] and default values for the other hyper-parameters are selected. The Min-Count hyper-parameter has been set to five for all word embeddings algorithms. In Section 4.6.7, it will be shown that the word embeddings vectors are significantly more stable if enough examples are presented to the algorithms. Five different examples appear to be enough to obtain stable word embeddings vectors.

23 for the Window Size hyper-parameter was selected as a proper value at that time. However, it will remain an open question what (presumably quite large) Window Size hyper-parameter value is best due to the ever-increasing better results for larger window sizes.

[8]The FastText algorithm's hyper-parameter study takes a long time and the best value of 8 was not known at that time. However, on a global scale the loss in quality is vanishingly small.

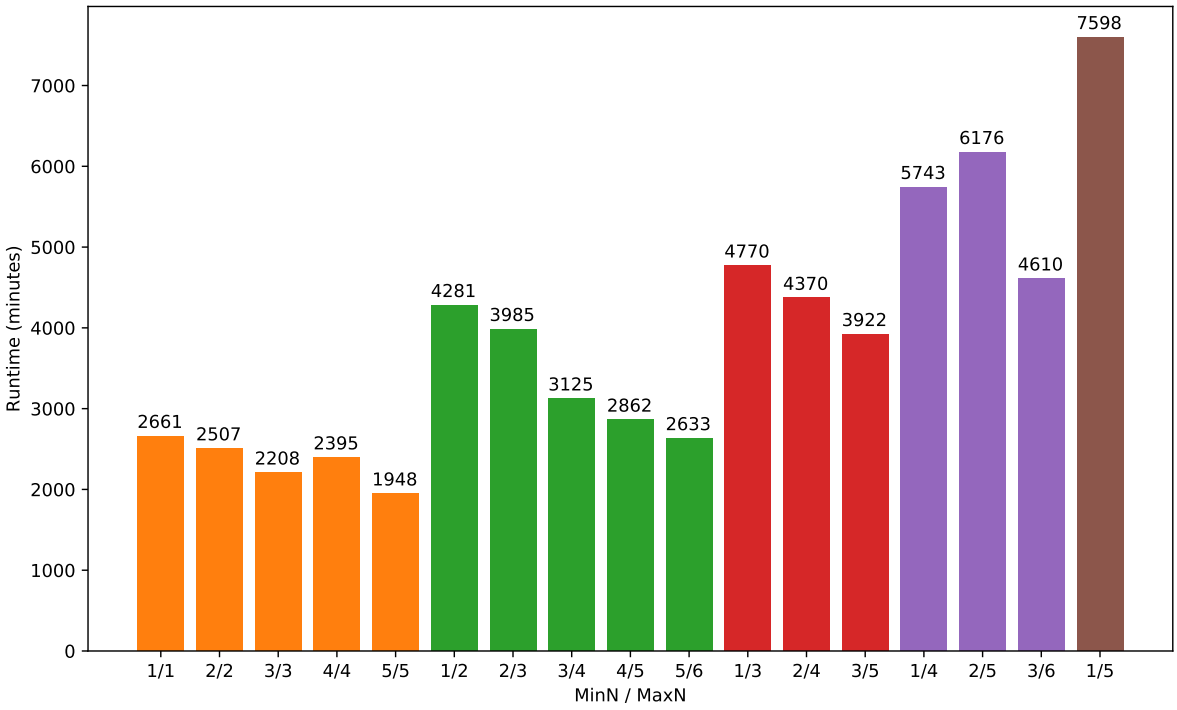

Figure 4.27.: Runtime of the hyper-parameter studies for FastText algorithm's hyper-parameters MinN and MaxN.

4.6.4. Comparison of Contemporary Word Embeddings Algorithms for the Synset Vector Approach

The goal of this section is to compare the GloVe, word2vec and FastText algorithms for the task of thesaurus reconstruction and extension. The word embeddings algorithms are compared using the Synset Vector Approach. The comparison includes quantitative and qualitative evaluations as well as exemplary candidate terms to derive patterns in the results. The Single Word and Synset Vector Approaches are compared in Section 4.6.5.

First, the word embeddings algorithms are evaluated quantitatively with precision/recall curves. The evaluation helps to estimate the capability of the different word embeddings algorithms with the Synset Vector Approach to reconstruct a given thesaurus. Figure 4.28 displays the precision/recall curves for the selected hyper-parameter configurations. The different word embeddings algorithms GloVe, word2vec and FastText are compared with precision/recall for the hyper-parameter configurations derived in the previous section. The precision/recall records are sub-sampled with a step size of 50 after 30 records and dropped after 1000 records. Word embeddings algorithms lead to significantly different results when compared quantitatively. The GloVe algorithm shows worse results than the word2vec and FastText algorithms according to precision and recall. This goes in line with the results of the RP-Score evaluations. While these results are significantly better than the results of previous count-based DSMs, see Section 4.6.6, the results again indicate that the Synset Vector Approach with

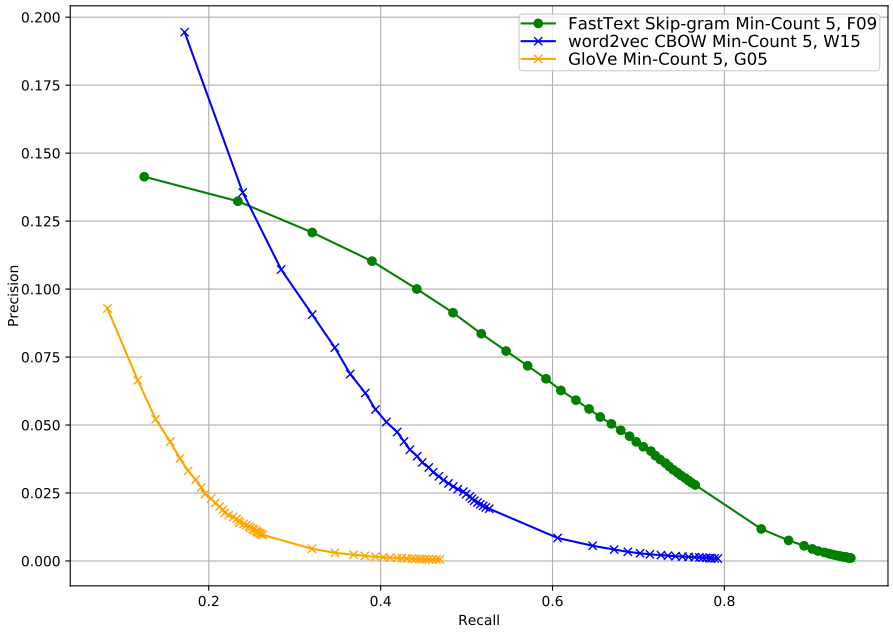

Figure 4.28.: Quantitative evaluation of word embeddings algorithms for the Single Word Approach.

any considered word embeddings algorithm performs not good enough for full automation of thesaurus construction.

Secondly, a qualitative evaluation is used to assess the potential of the Synset Vector Approach and word embeddings algorithms for thesaurus extension that cannot be evaluated quantitatively. Figure 4.29 illustrates the results of the qualitative evaluation. In general, the significant differences of the number of reasonable results confirm the results of the quantitative evaluation: The GloVe algorithm performs worse than the word2vec and FastText algorithms. The FastText algorithm performs best with an almost equal margin of the word2vec algorithm to the GloVe algorithm. On the one hand, an average of over 50% true synonyms in the first ten candidate terms suggested by the Synset Vector Approach with FastText algorithm is a surprisingly good result. On the other hand, it confirms the result of the quantitative evaluation that the investigated word embeddings algorithms produce too noisy results for full automation. From the results, it can be concluded that a quantitative evaluation can be used to compare the approaches and word embeddings algorithms. However, a precision of 0.075 (and an average recall of 0.6) of the first ten candidate terms for the FastText algorithm might not suggest that over 50% synonyms are included. Remember that for the quantitative evaluation, a train-/test-set split is required. For the qualitative evaluation, a train-/test-set split is not required and the full synsets are used as input to the Synset Vector Approach. It could be argued that the different input synset sizes are the reason for the drastically different results of a factor of around ten. However, it is improbable that the differences in the results can be explained with the input synset sizes alone. It is more likely a strong indicator that a "gold standard" thesaurus for a given real-world corpus does not exist.

Figure 4.6 displays the average occurrence frequency of the suggested candidate terms in the training corpus. The occurrence frequencies of the candidate terms calculated with the FastText and word2vec

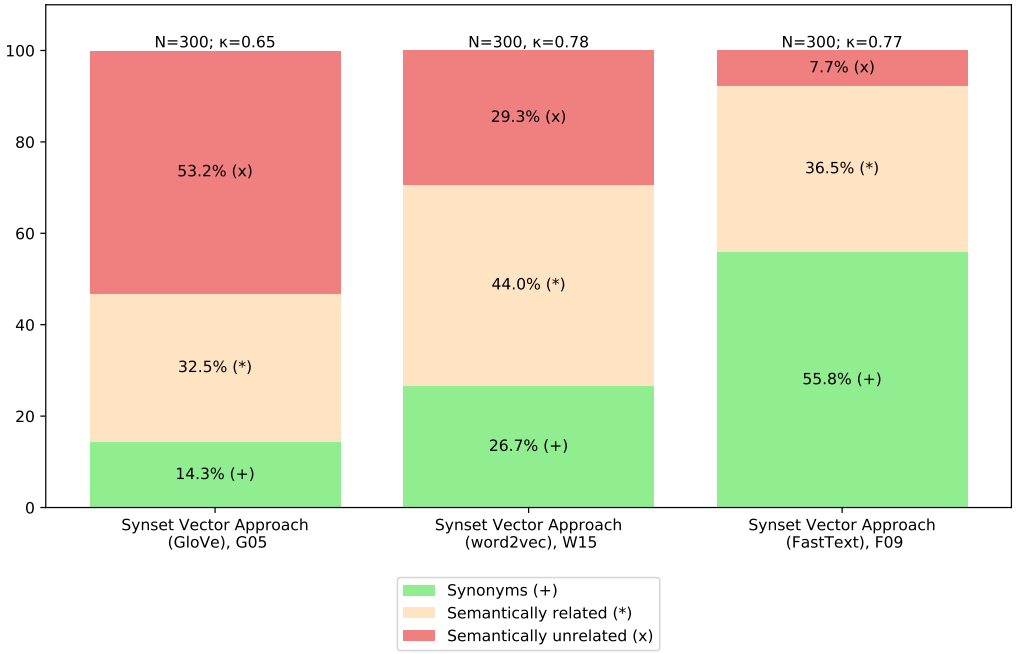

Figure 4.29.: Qualitative evaluation of word embeddings algorithms for the Synset Vector Approach.

algorithms are roughly equal. In contrast to the word2vec and FastText algorithms, the GloVe algorithm favors terms that occur significantly more frequently in the training corpus. This can be seen as an indication that the GloVe algorithm is more similar to count-based DSMs than predictive DSMs. Another interesting aspect is that the different word embeddings algorithms calculate very different candidate terms as it can be observed by the percentage of shared candidate terms listed in Table 4.7.

The results of the qualitative evaluation differ a lot. A manual result inspection can deliver more insights. Example results for two existing thesaurus synsets are listed in Figure D2 a) in the appendix. The existing synset on the topic of "informationsschrift" ("document for information") with four terms is of small to medium size in comparison to other synsets in the existing thesaurus. The scope of the existing thesaurus synset is medium to broad in comparison to other synsets and not tax law specific.

Approach (Word embeddings algorithm)	**Average term occurrence frequency**
Synset Vector Approach (GloVe), G05	2476.80
Synset Vector Approach (word2vec), W15	1681.35
Synset Vector Approach (FastText), F09	1843.62

Table 4.6.: Average occurrence frequencies of candidate terms suggested by the Synset Vector Approach.

Approach 1	Approach 2	Shared candidate terms
Synset Vector Approach (GloVe)	Synset Vector Approach (word2vec)	13%
Synset Vector Approach (word2vec)	Synset Vector Approach (FastText)	9.16%
Synset Vector Approach (FastText)	Synset Vector Approach (GloVe)	6.5%

Table 4.7.: Shared candidate terms of the GloVe, word2vec and FastText algorithms for the Synset Vector Approach

Many reasonable candidate terms fall into the scope of the existing synset. It is as an example for closed compound words because the term "informations" is part of all terms in the existing synset. The FastText algorithm delivers a large fraction of reasonable candidate terms. The example results give a good intuition that the candidate terms obtained with the FastText algorithm tend to be syntactically very close to the existing synset terms. The FastText algorithm mimics syllable embeddings. This explains the large fraction of syntactically close terms that are then often semantically close, too. In contrast to the FastText algorithm, the word2vec and GloVe algorithms suggest many terms that are semantically close but syntactically different. Both the word2vec and GloVe algorithms suggest reasonable terms, but the word2vec algorithm suggests slightly more useful and a larger number of reasonable terms.

The existing synset on the topic of "nachschusspflicht" ("reserve liability") is an example for a small existing synset with only two terms. The terms are tax-law-specific and the synset is also an example for a synset with a very small scope, i.e., only a few syntactically different candidate terms can be expected in the vocabulary. Similar to the first example, the FastText algorithm suggests mostly syntactically close candidate terms and several reasonable terms are found. The word2vec and GloVe algorithms suggest few syntactically similar terms but attempt to suggest many syntactically different terms. The word2vec and GloVe algorithms have a hard time to suggest reasonable terms because the scope of the existing synset is very narrow.

Both quantitative and qualitative evaluations suggest that the FastText algorithm is superior to the word2vec and GloVe algorithms. However, as can be seen by the example results, the FastText algorithm produces results that are close in terms of syntax. The good results of the FastText algorithm are also due to the nature of the existing thesaurus that contains many syntactically similar terms.

In summary, the GloVe and word2vec algorithms produce qualitatively similar candidate terms, but the amount of good candidate terms is higher for the word2vec algorithm. The FastText algorithm suggests a very high number of good results. However, due to the use of sub-word level embeddings that mimic syllable embeddings, the FastText algorithm suggests mostly candidate terms that are semantically and syntactically very similar to the terms in the input synset. In contrast to that, the candidate terms calculated with the GloVe and word2vec algorithms are semantically similar but very often syntactically different. It could be argued that further hyper-parameter optimization of the GloVe algorithm, for example, of the Window Size hyper-parameter, could lead to better results. However, the word2vec algorithm appears to be much more robust to hyper-parameter changes than the GloVe algorithm and therefore is easier to use. It seems unlikely that the quantitative results of the word2vec algorithm can be achieved with the GloVe algorithm.

4.6.5. Comparison of the Single Word and Synset Vector Approaches with the word2vec Algorithm

The Synset Vector Approach takes several query terms as input and calculates an average query term vector (a synset vector) that is then used to find synonymous words. In contrast to that, the Single Word Approach takes single query terms as input to find synonymous words. The Single Word Approach serves as a vehicle to compare a count-based DSM to word embeddings based approaches. Count-based DSM implementations also take only single query terms as input. The comparison of the Single Word and Synset Vector Approaches further serves as a justification for the Synset Vector Approach.

A direct comparison of Single Word Approach and Synset Vector Approach in a straightforward manner is not possible. The Single Word Approach calculates a fixed-length candidate term list for each query term. In contrast to the Single Word Approach, the Synset Vector Approach produces one fixed-length candidate term list per synset. The Synset Vector Approach degenerates to the Single Word Approach for input synsets consisting of single terms. This would be the case for a large fraction of synsets for quantitative evaluations that use a train/test split of the existing thesaurus because most synsets comprise two terms. However, these are not the exciting cases. For synsets comprising more than one term, combining the candidate term lists of all terms to a single fixed-length candidate term list is difficult. A possible way to achieve this is, for example, to select a fixed-length result list by ranking the terms of multiple candidate term list according to cosine similarity. The alternative is to evaluate a concatenation of the candidate term lists of all input synset terms. This would imply the rating of 844 candidate terms for the evaluation conducted here. However, manually selected example results will demonstrate the superiority of the Synset Vector Approach over the Single Word Approach.

The qualitative evaluation in this section is carried out as follows: The qualitative evaluation of the Synset Vector Approach in Section 4.6.4 is re-used. For the Single Word Approach, the first term of a synset is chosen to calculate one fixed-length candidate term list of ten candidate terms. This is equivalent to select one random query term from a synset. The ten candidate terms are evaluated equally to the evaluation of the Synset Vector Approach. Thus, only a fraction of the full list of 844 candidate terms are considered. The full input synsets (i.e., no train/test split) are used. This means that all synsets contain at least two terms. The results of the qualitative evaluations conducted so far are shown in Figure 4.30. The results suggest that the Synset Vector Approach is superior to the Single Word Approach when a random query term is selected for the Single Word Approach.

The ratio of shared candidate terms by the two approaches is 21.66%. In comparison to the other approaches and word embeddings algorithms, many candidate terms are shared. However, given that the candidate terms are calculated with the same word2vec word embeddings model, it is a small ratio. The synsets of the evaluation thesaurus for the qualitative evaluation comprise 98 terms. Calculating candidate term lists with ten candidate terms results in a total of 980 candidate terms. 136 of the 980 terms are duplicates. Consequently, 844 unique candidate terms are contained in the candidate term lists of all input terms. Among these 844 terms, 245 are shared with the 300 candidate terms obtained by the Synset Vector Approach, i.e., a ratio of 81.6%. The Single Word Approach seems to be able to find a large fraction of the candidate terms obtained by the Synset Vector Approach at the expense of a significantly more extensive candidate term list. The downside of this investigation is that it does not answer the question of how many additional valuable synsets would be obtained by considering the

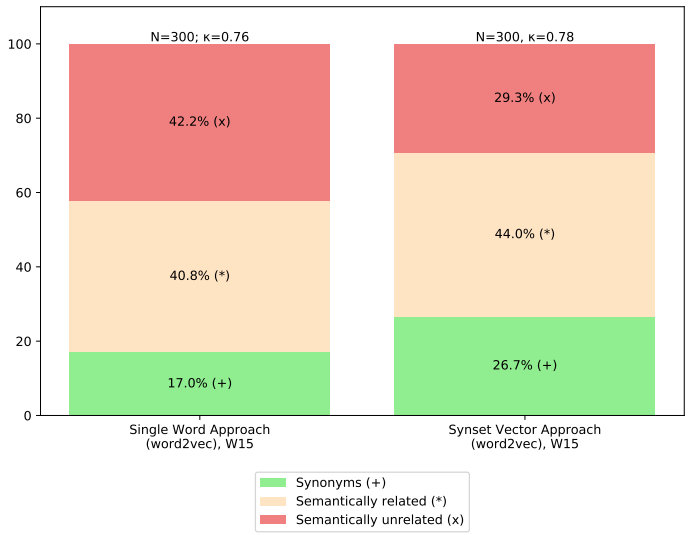

Figure 4.30.: Qualitative evaluation of Single Word (word2vec) and Synset Vector (word2vec) Approaches.

candidate term lists of 844 terms. However, a manual inspection of the candidate term lists will show additional insights.

Example results for the candidate terms suggested by the two approaches are listed in Figure D1 b) in the appendix. The existing synset on the topic of "firmen-pkw" ("company auto") with 13 terms is an example for a very large synset in the existing thesaurus. Despite its size, the scope of the synset is still sharp in comparison to other synsets. The use of the terms in non-tax law specific language indicates that a large number of reasonable candidate terms can be expected. Both syntactically similar and syntactically different terms can be expected. The synset serves as an excellent example for a case where the Synset Vector Approach is superior to the Single Word Approach. The example results for this synset show that the Single Word Approach does not capture the semantic aspect "company" of the given synset. In contrast to the Single Word Approach, the Synset Vector Approach that uses the mean vector of the word embeddings vectors of all 13 terms captures the "company" aspect of the given synset significantly better.

The existing synset on the topic of "gasthaus" ("guest-house") has two input terms and is very small. The scope of the synset is sharp, but similar to the first example, many candidate terms that are syntactically different can be expected due to the heavy usage of the term in general language. The example results are similar to the first example because both approaches find many different reasonable candidate terms. However, also for this very small existing synset, the Synset Vector Approach delivers a significantly larger number of correct candidate terms.

In summary, the Synset Vector Approach is equal to the Single Word Approach for single term input synsets but different for larger input synsets. The Synset Vector Approach can lead to improved results compared to the Single Word Approach, especially in the cases of larger input synsets or synsets of

higher semantical complexity. When possible, the Synset Vector Approach should be favored over the Single Word Approach.

4.6.6. Comparison of the JoBimText and Single Word (word2vec) Approaches.

In this section, a representative of count-based DSMs is compared to the contemporary word2vec algorithm based Single Word Approach on the task of thesaurus extension. The JoBimText algorithm is selected as a count-based DSM implementation. The patterns observed in the results of the different word embeddings algorithms suggest that contemporary word embeddings algorithms are superior to count-based DSMs for the use case at hand.

For the evaluation, the virtual machine published by the authors of the JoBimText tool suite is used[9]. The JoBimText algorithm calculates a distributional thesaurus from a training corpus. The implementation to calculate a distributional thesaurus uses the Hadoop framework. Thus, the distributional thesaurus can be calculated using several computing devices at the same time. The Datev training corpus with around 1GB of textual data requires an adjustment of the default virtual machine settings in order to deal with the large corpus. The virtual machine is assigned 12 GB RAM, up to 90% CPU resources of the host and 500 GB virtual disk size.

The JoBimText algorithm requires a specific input format of the training corpus. In contrast to the pre-processing steps listed in Section 4.5, the input format requires one sentence per line with full punctuation and no pre-processing of the casing. The punctuation is not removed but pre-processed into white-space separated tokens. For example, the string "Hello, my name is Peter." is transformed into "Hello , my name is Peter .". Note the white-space character before punctuation character. The sentence segmentation is carried out with the spacy framework. spacy's sentence segmentation algorithm for the German language is a good start but it stumbles upon legal-specific language elements like citations. To mitigate this issue to some degree, the sentence segments obtained from spacy are post-processed. The sentence segments that consist of only a few characters or words are assigned to previous text segments[10].

The JoBimText algorithm comes with many hyper-parameters, for example, minimum feature count, minimum word count, minimum word feature count, minimum significance score and significance measure to name a few. The LMI significance measure and the bigram model are selected, cf. Riedl and Biemann (2013). The same configuration is used for a comparison of the JoBimText and word2vec algorithms by Ramrakhiyani et al. (2015). The hyper-parameters to calculate a single distributional thesaurus with the JoBimText algorithm are set to default values. The default minimum token occurrence frequency of a token for the JoBimText implementation is equal to word2vec algorithm's default Min-Count hyper-parameter.

The distributional thesaurus calculated with the JoBimText algorithm has a size of around 300MB,

[9] https://sourceforge.net/projects/jobimtextgpl.jobimtext.p/files/hadoop-VM/, version of April 13, 2015, last accessed July 17, 2019

[10] Technical details: spacy's sentence segmentation stumbles upon references contained in the legal documents. For one document, all sentence segments are sequentially examined. If a sentence segment has less than 30 characters or less than five tokens, the sentence is appended to the previous text segment until the resulting text segment has a length of at least 30 characters or five tokens. This heuristic does not lead to perfect results but mitigates a majority of the errors on the given corpus to a reasonable degree.

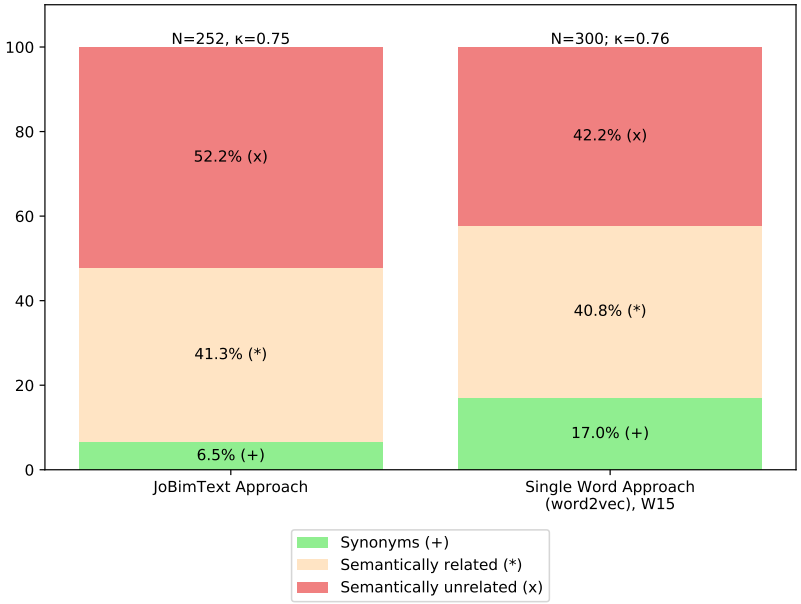

Figure 4.31.: Qualitative evaluation results for the JoBimText and Single Word (word2vec) Approaches.

which is in the range of the size of comparable word embeddings models. However, the calculation for a single distributional thesaurus took more than 12 hours and consumed around 300GB of disk space. No further hyper-parameter studies have been conducted. One could argue that comparable results of the JoBimText and Single Word Approaches with word2vec could be achieved through further hyper-parameter optimization. However, the patterns that emerge in the presented results indicate that also further hyper-parameter optimization most likely would lead to inferior results of the JoBimText Approach.

The results of the JoBimText Approach are hard to compare to the results of the Single Vector Approach. The distributional thesaurus calculated by the JoBimText algorithm results in candidate term lists of varying size. For several query terms, zero neighboring words are returned. For the qualitative evaluation, the same synsets are selected from the given thesaurus as for the Single Word Approach, but the total number of candidate terms is only $N = 252$ for the JoBimText Approach compared to $N = 300$ for the Single Word Approach. Figure 4.31 shows the results of the qualitative evaluation. From the limited amount of candidate terms, significantly smaller ratios of true synonyms and semantically related terms are obtained by the JoBimText Approach than by the Single Word Approach with the word2vec algorithm.

Example results for candidate terms suggested by the JoBimText Approach and the Single Word Approach with the word2vec algorithm are listed in Figure D1 a) inn the appendix. The first example with a synset on the topic of "steuerfestsetzungsverfahrens" ("taxing procedure's") with two terms has a narrow scope. It is an example of input terms that are closed compound words and a synset that is tax-law specific. The synset has complex semantics from a tax law point of view. Few candidate terms that correctly capture the semantics of the input synset should be expected because terms that share syntactic elements can still have different semantics, for example, "steuerfestsetzungsfrist" ("taxing pe-

Approach (Word embeddings algorithm)	Average term occurrence frequency
JoBimText (-)	4338.48
Single Word Approach (word2vec), W15	1471.38

Table 4.8.: Average occurrence frequency analysis for the Single Word and Synset Vector Approaches with the word2vec algorithm.

riod"). The second example is a synset on the topic of "lohnberechnung" ("wage calculation"). It has four terms and is of small to medium size. It is again an example with terms that are closed compound words. The terms are used in tax law but also business-related language. The scope of the synset is sharp but has complex semantics because many syntactically similar and different terms exist but that often have slightly different semantics. For example, the terms "gehalt" ("pay") and "lohn" ("wage") are used in tax law while the term "vergütung" ("compensation") is used by most but not all lawyers in the context of civil law. Despite the complex semantics, several reasonable candidate terms should be expected due to the use in more general language than tax law specific and law specific language.

For both examples, the number of suggested correct synonyms is low. However, the average ratings suggest that the Single Word Approach with the word2vec algorithm does capture the semantics of the existing synsets better. The analysis of the average term occurrence frequency of the candidate terms, shown in Figure 4.8, suggests that the JoBimText Approach favors tokens that have a high occurrence frequency in the training corpus. The example results confirm the quantitative results of the qualitative evaluation. For the second example synset on the topic of "lohnberechnung" ("wage calculation"), the JoBimText Approach only suggests two candidate terms that are very "simple" terms: "abrechnung" ("accounting") and "berechnung" ("calculation"). These terms can be easily imagined to occur very often in any training corpus. The Synset Vector Approach with the word2vec and FastText algorithms suggests more reasonable results than the JoBimText Approach. The ratio of shared candidate terms for the qualitative evaluation with 1.99% is meager. The results suggest that the JoBimText Approach produces significantly different results than the Single Word Approach. The JoBimText Approach favors terms that frequently occur in the training corpus. The manual results inspection confirm these results.

The JoBimText algorithm has been compared to the word2vec algorithm previously to find synonyms for query terms by Ramrakhiyani et al. (2015). The authors use a similar categorization with three categories of "synonyms", "hypernyms, hyponyms or siblings" and "other". In contrast to the evaluation presented here, Ramrakhiyani et al. selected query terms from the 150 most occurring terms in the training corpus and inspected the top five candidate terms. Ramrakhiyani et al. reports a 30% increase in Average Precision @5. The results of Ramrakhiyani et al. go in line with the results presented in this thesis, where the Single Word Approach with the word2vec algorithm delivers significantly better results than the JoBimText Approach. However, in Ramrakhiyani et al. (2015), only a subset of the 150 most frequently occurring tokens in training corpus and the top-five candidate terms are considered, while here the Min-Count hyper-parameter is only restricted to five and the top ten candidate terms are inspected. This can explain the stronger difference of a factor of 2.5 more correct synonyms in the evaluation presented here.

In summary, the results of the qualitative evaluation suggest that the Single Word Approach with the word2vec algorithm is superior to the JoBimText Approach to find synonyms for thesaurus extension. The JoBimText algorithm is a representative of count-based DSMs. The JoBimText Approach favors frequently occurring words in the training corpus for both query and candidate terms. From another point of view, the word2vec word embeddings models seem to be more capable of dealing with much less frequently occurring tokens in the training corpus than the JoBimText algorithm. Calculating a word2vec word embeddings model is much easier than calculating and accessing a distributional thesaurus calculated with the JoBimText algorithm.

4.6.7. Comparison of the Intersection and Synset Vector Approaches with the word2vec Algorithm

The motivational observation for the Intersection Approach is that even for word embeddings models that reached convergence, the candidate term lists still vary a lot when hyper-parameters are slightly varied. This holds in particular when setting the Min-Count hyper-parameter to one. The basic idea of the Intersection Approach is to intersect the candidate term lists of several word embeddings models calculated with different hyper-parameter configurations so that only a stable subset of candidate terms is retained. From another point of view, the goal is to filter out candidate terms that occur only by chance in candidate term lists. The Intersection Approach utilizes the Synset Vector Approach. Several candidate term lists are calculated with the Synset Vector Approach with varying hyper-parameter configurations for the word2vec algorithm. The goal of this section is to compare the Intersection Approach to the Synset Vector Approach. First, the conditions for the Intersection Approach to be applicable are investigated. Next, the hyper-parameters of the Intersection Approach are examined. Finally, a qualitative evaluation compares Intersection and Single Word Approaches.

The Intersection Approach comes with additional degrees of freedom and hyper-parameters. The hyper-parameter configurations to calculate the word embeddings models can vary in one or several hyper-parameters. A maximum candidate term list length K needs to be chosen because intersecting candidate term lists with a length of the full vocabulary size does not filter candidate terms. The number of intersection operations I can be increased to intensify the filtering behavior of the Intersection Approach. Note that I is defined as the number of intersection operations. One intersection operation takes two candidate term lists. Hence, setting I to two requires three different word embeddings models. The input to an intersection operation can be the output of a previous intersection operation. Note that the outputs of the Synset Vector Approach are candidate term lists of fixed-length. This length is specified by the Intersection Approach hyper-parameter K. These candidate term lists are input to the intersection operations. The output of an intersection operation is again a candidate term list. However, the size of this candidate term list is equal or to the smallest input candidate term list or smaller.

Similar to the JoBimText Approach, the Intersection Approach does not return fixed-length candidate term lists. Only the maximum length of candidate term lists is limited by the maximum candidate term list length hyper-parameter K. The Iterations hyper-parameter of the word embeddings algorithms is a good candidate to obtain word embeddings models with varying hyper-parameter configurations and investigated in the remainder.

For $I = 0$, i.e., only one word embeddings model is used, the Intersection Approach degenerates

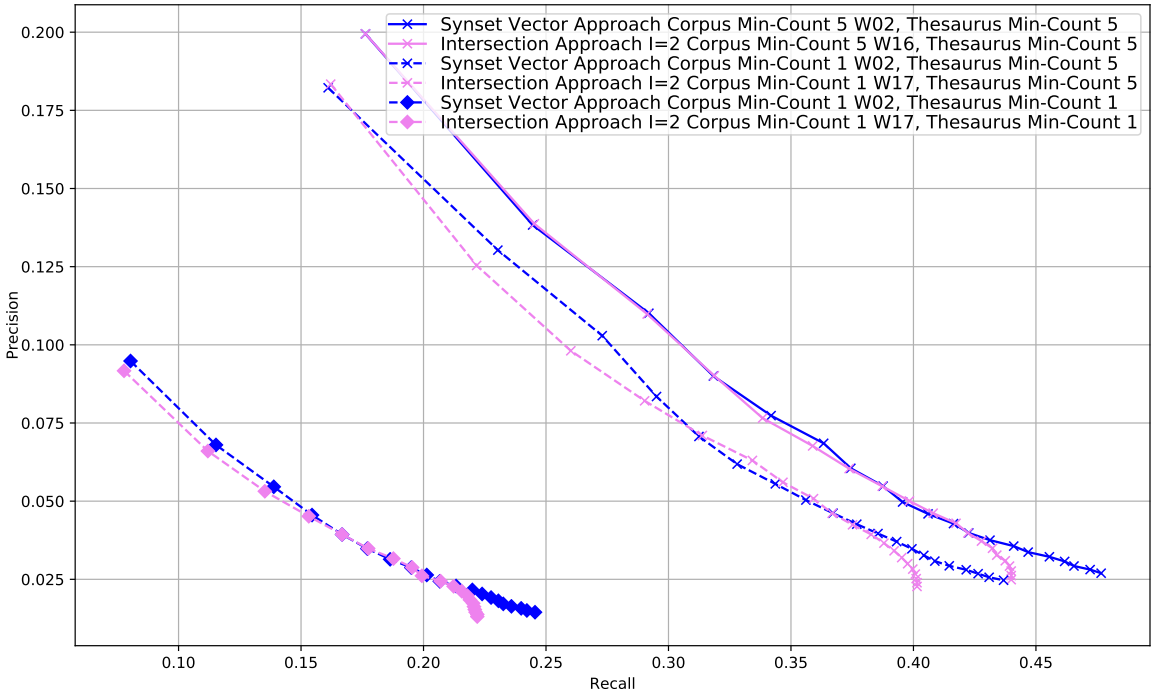

Figure 4.32.: Comparison of the Min-Count hyper-parameter for the Intersection and Synset Vector Approaches.

to the Synset Vector Approach. Therefore, the Synset Vector Approach is a suitable baseline for the Intersection Approach. Figure 4.32 shows an initial precision/recall analysis with different hyper-parameter selections for the Synset Vector Approach and the Intersection Approach with $I = 2$. The Min-Count hyper-parameter for the word2vec algorithm is varied as well as the Min-Count hyper-parameter for the evaluation thesauri.

The significant results of Figure 4.32 lie in the differences among pairs of Synset Vector and Intersection Approaches' precision/recall curves with equal Min-Count hyper-parameter configurations for training corpus and evaluation thesaurus. These pairs of curves are very close from a visual point of view. For training corpus and evaluation thesaurus obtained with the Min-Count hyper-parameter set to five, the precision/recall curves coincide for a wide range of maximum candidate term list lengths K. An interpretation of this observation is that word embeddings models for the Min-Count hyper-parameter set to five reached convergence to a substantial degree. In this case, the Intersection Approach is not applicable.

Remember that an evaluation of a word embeddings model where the Min-Count hyper-parameter is set to five and an evaluation thesaurus with the Min-Count hyper-parameter set to one is not possible due to the out-of-vocabulary problem. The remaining pairs use word embeddings models calculated with the Min-Count hyper-parameter set to five. The evaluation thesaurus with the Min-Count hyper-parameter set to five can be re-used. However, due to the three times larger vocabulary, i.e., three times more potential candidate terms, inferior results can be expected, and this is confirmed by the results

shown in Figure 4.32. In the case that both training corpus and evaluation thesaurus are calculated with the Min-Count hyper-parameter set to five, results become even worse, as it can be expected. However, for both pairs of Intersection and Synset Vector Approaches' precision/recall curves, the precision/recall curves coincide less. In absolute terms, the differences appear small but remember that only two intersection operations are applied.

Another interesting aspect is that for all pairs of Intersection and Synset Vector Approaches, the recall drops significantly for larger values of K. Within a precision/recall curve result set sizes of $K = 1, \ldots, 20$ are traversed. An explanation for the drop in the recall is that the intersection operations start to filter stronger when input candidate term lists become larger. Moreover, the drop in the recall values is only present in the precision/recall curves of the Intersection Approach.

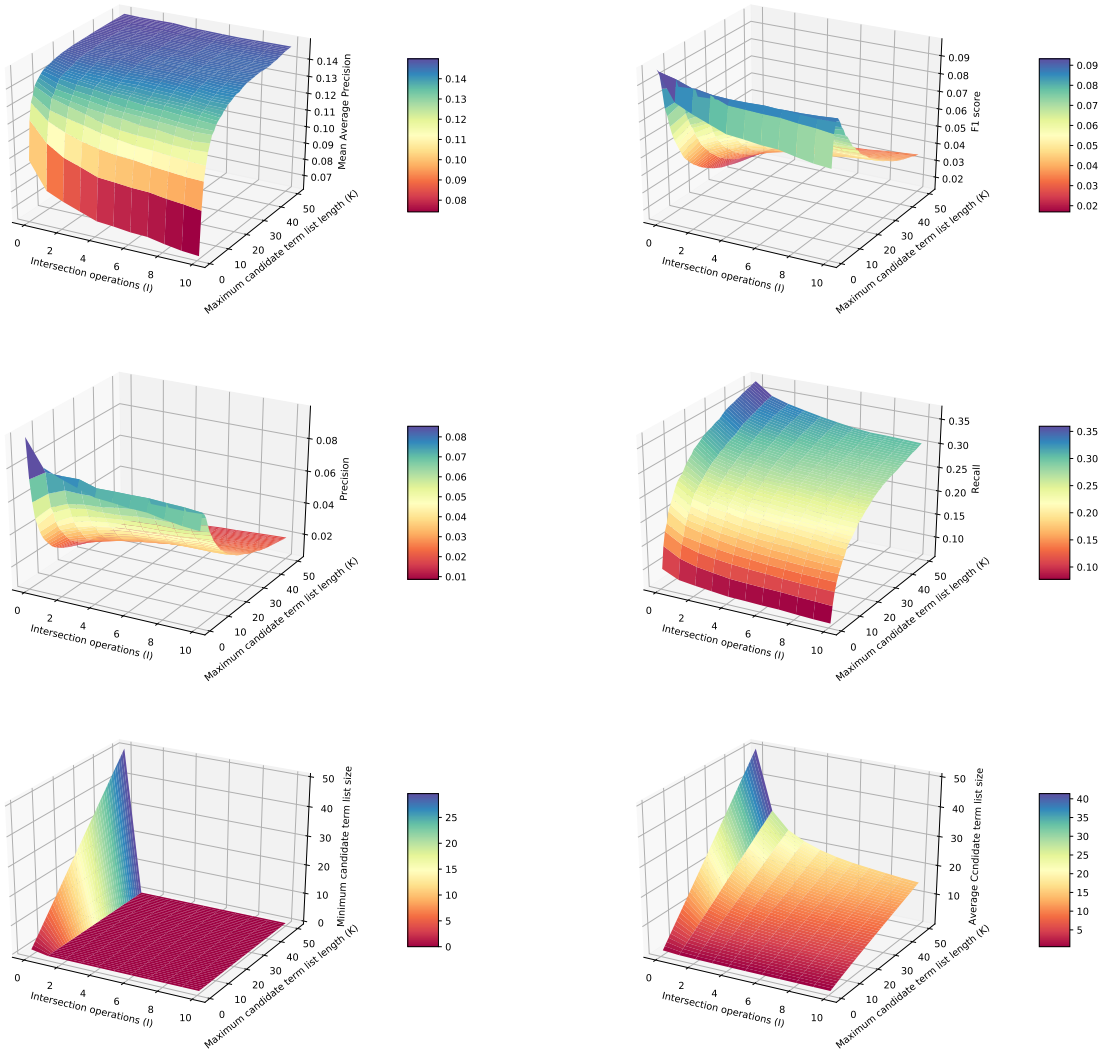

Figure 4.33.: Hyper-parameter study of the hyper-parameters Maximum candidate term list length (K) and number of intersection operations (I) for the Intersection Approach.

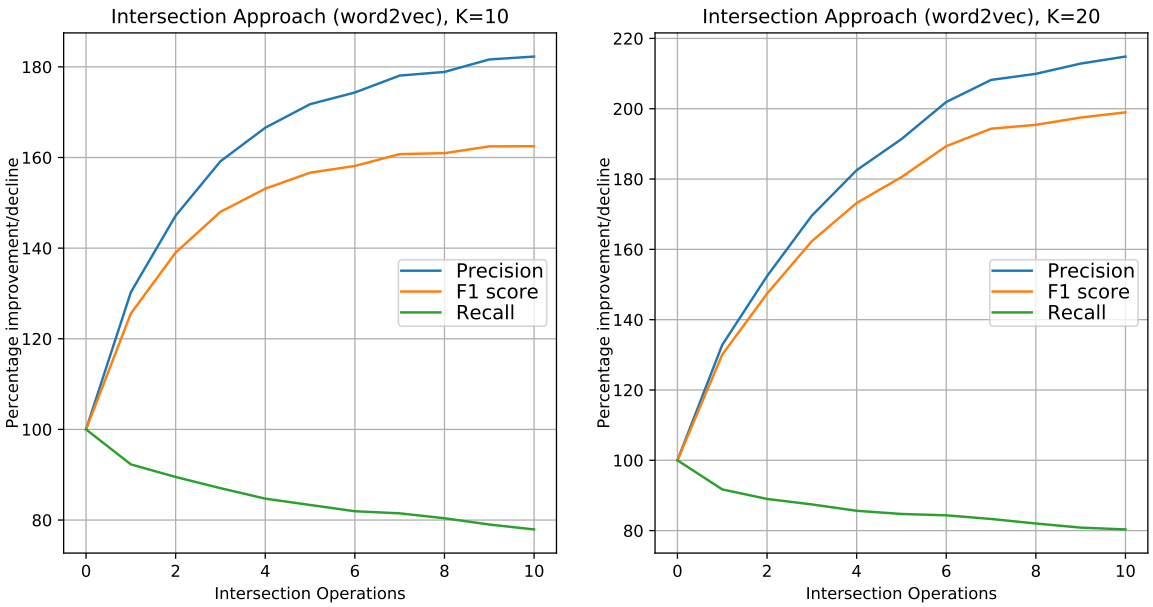

Figure 4.34.: Change of precision, recall and F_1 score for the Intersection Approach depending on the number of intersection operations (I).

Next, values for the I and K hyper-parameters of the Intersection Approach need to be selected. Figure 4.33 shows the behavior of several relevance measures and candidate term list sizes dependent on a combination of intersection operations and maximum candidate term list length hyper-parameters. For the evaluation shown in Figure 4.33, all word embeddings models have been calculated with the word2vec algorithm with hyper-parameter configuration W18. The MAP is calculated using Equation 3.9, i.e., the total amount of expected correct results according to the evaluation thesaurus is used. However, the other ways to calculate the MAP lead to qualitatively similar results.

The results show that the average size of final candidate term lists is reduced with an increasing number of intersection operations, i.e., the filtering of the Intersection Approach is working. An increased number of intersection operations improves precision, i.e., correct candidate terms are retained. However, recall declines for an increased number of intersection operations, i.e., reasonable candidate terms are filtered (according to the evaluation thesaurus), too. The F_1 score shows a more complex behavior. For smaller values of the maximum candidate term list lengths hyper-parameter, the Intersection Approach works best. However, in general, for a larger number of intersection operations the F_1 score rises, while the MAP slightly drops especially for small values of K. Remember that the MAP gives a high weight to the first ranking positions. The minimum result list length is reached after one intersection operation, i.e., for certain evaluation thesaurus synsets, all candidate terms are filtered. This goes in line with the large RP-Scores observed in Section 4.6.2 where several correct relationships according to the evaluation thesaurus are not reflected in the word embeddings models, but very large ranking position distances among correct words occur.

Figure 4.34 shows slices of the three-dimensional sub-plots of precision, recall and F_1 score of Figure 4.34 for $K = 10$ and $K = 20$. To better understand the changes in the relevance measures dependent on intersection operations, Figure 4.34 shows the relative changes of the relevance measures. Again,

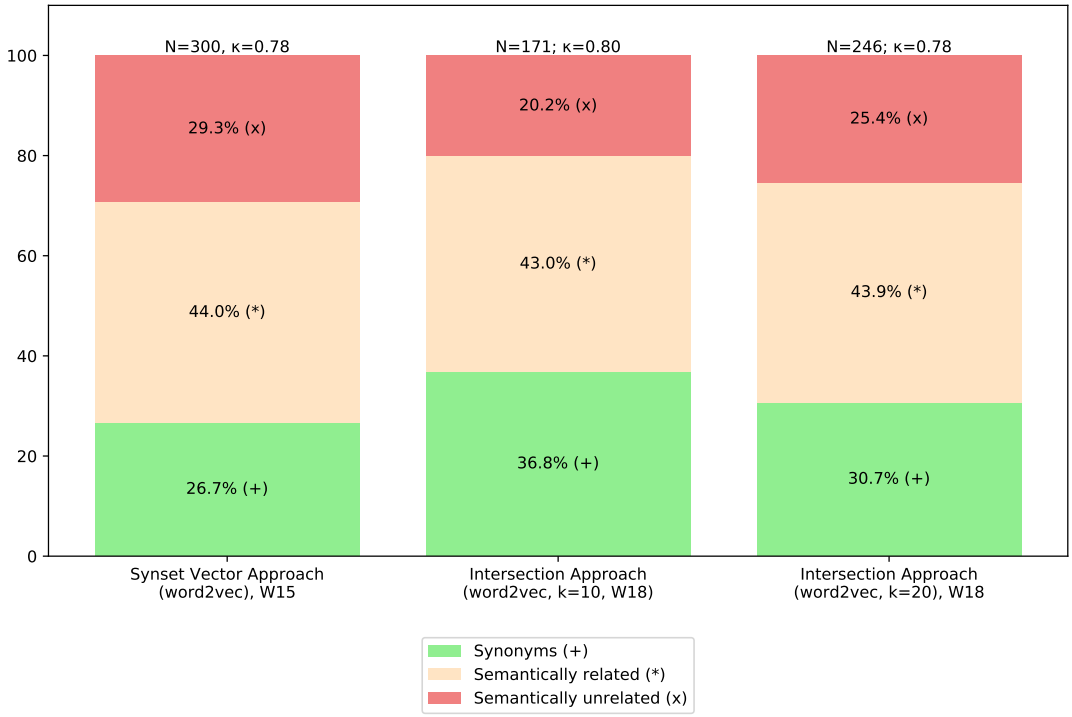

Figure 4.35.: Qualitative evaluation results for the Intersection and Synset Vector Approaches.

for both selected values of the maximum candidate term list length K, the precision increases while the recall drops. However, for both selections of K, the increase in precision exceeds the decline in recall so that the F_1 score increases.

The results of the quantitative evaluation of the additional hyper-parameters of the Intersection Approach do not lead to clear recommendations for the hyper-parameters. Due to the selection of ten candidate terms for the qualitative evaluation, $K = 10$ and $K = 20$ are selected for the qualitative evaluation. For $K = 10$, the filtering effect of the Intersection Approach can be examined. For $K = 20$, it can be investigated, if further reasonable candidate terms are found in larger candidate term lists.

The results of the qualitative evaluation presented in Figure 4.35 suggest that in relative terms, more correct candidate terms are retained with the Intersection Approach for $K = 10$ than proposed by the baseline Synset Vector Approach. Even if the filtering of noisy terms should work to some degree for K=20, additional intersection operations might lead to even better results.

The number of shared candidate terms between the Synset Vector Approach and the Intersection Approach can be expected to be high. Table 4.9 shows the number of shared candidate terms among the approaches. The ratio of shared terms is high in comparison to shared terms analysis with other approaches and word embeddings algorithms. However, the values are less than expected. Note that the Synset Vector Approach uses the Min-Count hyper-parameter set to five, while the Intersection Approach uses the Min-Count hyper-parameter set to a value of one, i.e., the results are not directly comparable. Nevertheless, the results of the shared terms analysis suggest that additional reasonable

Approach 1	Approach 2	Shared candidate terms
Synset Vector Approach (word2vec)	Intersection Approach (word2vec,k=10)	32.27%
Intersection Approach (word2vec,k=10)	Intersection Approach (word2vec,k=20)	40.52%
Intersection Approach (word2vec,k=20)	Synset Vector Approach (word2vec)	35.89%

Table 4.9.: Shared candidate terms of Synset Vector and Intersection Approaches.

Approach (Word embeddings algorithm)	Average term occurrence frequency
Synset Vector Approach (word2vec), W15	1681.35
Intersection Approach (word2vec), K=10, W18	1642.11
Intersection Approach (word2vec), K=20, W18	1575.08

Table 4.10.: Average occurrence frequency analysis for the Intersection and Synset Vector Approaches.

candidate terms are found that would not have been found with the Synset Vector Approach when the Min-Count hyper-parameter is set to five.

The average occurrence frequency of candidate terms in the training corpus is presented in Table 4.10. The results support to a minor degree that the Intersection Approach detects reasonable candidate terms that occur less than five times in the training corpus. For $K = 10$, eight and for $K = 20$, fourteen candidate terms of the qualitative evaluation of the Intersection Approach occur less than five times in the training corpus. In both cases, this constitutes around 5% of all candidate terms. Several terms of the eight and fourteen candidate terms are true synonyms.

The example results for two synsets for the Synset Vector Approach and two different Intersection Approach configurations are listed in Figure D2 b) in the appendix. The existing synset on the topic of "speditionsunternehmen" ("forwarding enterprise") with six terms is of medium size, has a comparably sharp scope and the terms are used in business language a lot. All input terms of the synset are closed compound words and nouns. Due to the many possibilities to form open compound words, cf. Section 4.2, many reasonable candidate terms can be expected.

The existing synset on the topic of "storno" ("reversal") with three candidate terms is small and serves as an example for a rather less sharp synset from a linguistic point of view. The synset contains nouns and verbs at the same time. Like for the first example, the terms are used in general and business language a lot. Nevertheless, only a moderate amount of reasonable results should be expected because the semantic scope of the synset is still sharp.

For both examples and both Intersection Approach configurations ($K = 10, K = 20$), the Intersection Approach suggests less than ten candidate terms, i.e., several candidate terms are filtered. The second example synset is very good because only correct synonyms are retained for the Intersection Approach

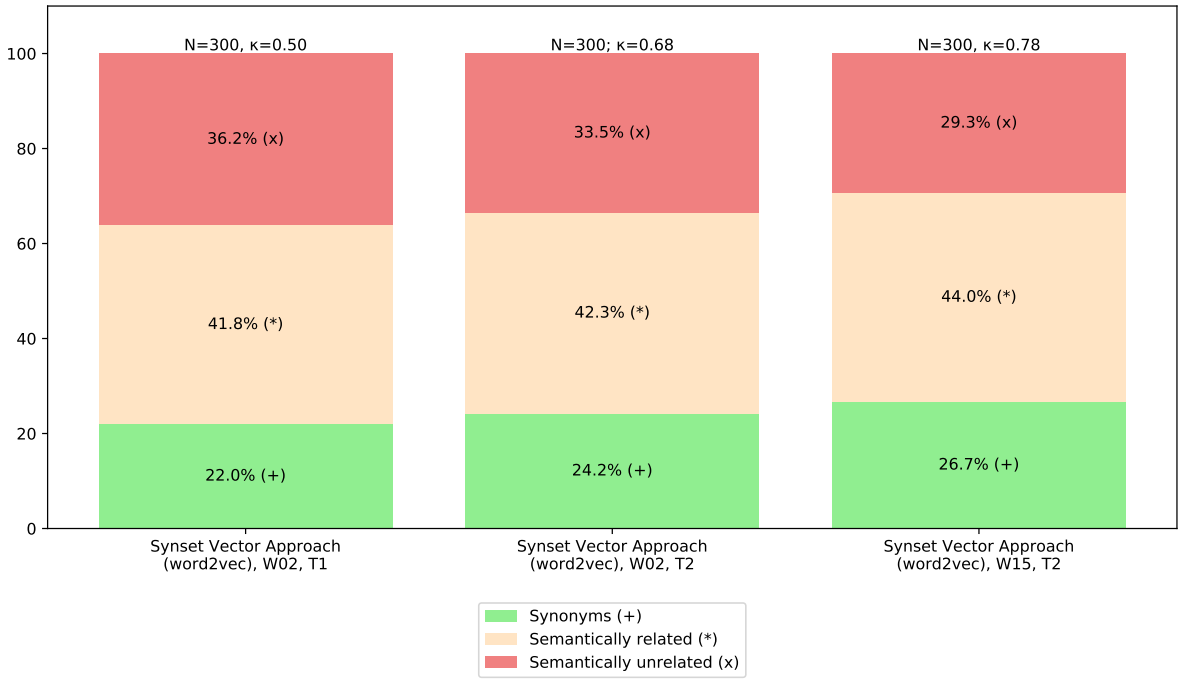

Figure 4.36.: Qualitative evaluation results of different hyper-parameter configurations and evaluation thesaurus selections for the Synset Vector Approach.

with $K = 10$. However, the results for the first example synset show, that still semantically unrelated candidate terms are contained in the results. For both example results, the filtering characteristics of the Intersection Approach with $K = 20$ is less good which could indicate that more intersection operations would be required to obtain better results. This especially applies if a high precision is more important than a high recall.

In summary, the Intersection Approach can be used to filter reasonable candidate terms from the Synset Vector Approach candidate term lists if Min-Count hyper-parameter values smaller than five are used. The number of intersection operations can be used to adjust the filtering effect. The filtering effect is not perfect and reasonable candidate terms are removed, too. Nevertheless, the filtering of the Intersection Approach can be used to improve the proportion of reasonable candidate terms presented to human experts for review.

4.6.8. Sensitivity Analysis of the Synset Vector Approach with the word2vec Algorithm

The goal of this section is to investigate the impacts of small variations of the hyper-parameters of the word2vec algorithm and variations of the selection of synsets to the results of the qualitative evaluation for the Synset Vector Approach with the word2vec algorithm.

Figure 4.36 shows the results of the two variations to the qualitative evaluation. The two columns on the left in Figure 4.36 vary in the selection of the evaluation synsets (T1 and T2). The hyper-parameter

selection is equal (W02). The evaluation synsets have been selected randomly from the 2500 evaluation thesaurus synsets and have no synsets in common. Note that the evaluation synsets selection on the center column is the synset selection described in Section 4.6.1.2. The almost identical results underline that 30 synsets, irrespective of the selection of synsets, are sufficient to lead to stable results for the qualitative evaluation.

The center and the right columns in Figure 4.36 vary in the selection of hyper-parameters of the word2vec algorithm. The two different hyper-parameter configurations of the word2vec algorithm "W02" and "W15" vary in the Vector Size (150 vs. 400), Window Size (8 vs. 5), Negative Samples (8 vs. 5) and Sample Threshold (1e-3 vs. 5e-4) hyper-parameters but share the Model Architecture (CBOW), Iterations (40) and Min-Count (5) hyper-parameters. Despite the changes in the hyper-parameter configurations, the results of the qualitative evaluation are very close. However, note that the candidate term lists among the two hyper-parameter configurations only share 26.5% of the suggested terms (for the same evaluation synset selection), i.e., the suggested candidate term lists vary strongly despite very similar qualitative evaluation results. It can be further noted that Cohen's kappa values, i.e., the agreement among raters varies between 0.5 and 0.78, but that the aggregated percentage values for the three categories *synonyms*, *semantically related* and *semantically unrelated* are very similar. Thus, Cohen's kappa values larger than 0.5 can be seen as sufficient for the qualitative evaluations with 30 evaluation synsets and $N = 300$ terms classified by two raters.

In summary, the results in this section indicate that the qualitative evaluation and the Synset Vector Approach with the word2vec algorithm are very robust to both small changes in the word2vec algorithms hyper-parameter configurations as well as evaluation synset selection from the given thesaurus. However, changes in the hyper-parameter configurations of the word2vec algorithm can lead to very different candidate term result lists. The results further indicate that the results of the qualitative evaluation of Label Propagation and Synset Vector Approaches are comparable despite that different evaluation synset are selected and that different word2vec algorithm hyper-parameter configurations are used.

4.6.9. Comparison of the Label Propagation and Synset Vector Approaches with the word2vec and FastText Algorithms

The Label Propagation Approach uses word embeddings and label propagation algorithms to extend thesauri. The Label Propagation Approach is motivated by the good results for ontology construction by Ravi and Diao (2016) that use label propagation algorithms. The goal of label propagation algorithms is to leverage the global structure of graphs, i.e., label propagation algorithms should be able to detect chains of synonymous tokens. The Label Propagation Approach has been investigated in Mueller (2018) in more depth. Small variations of the results of Mueller (2018) are presented here with the goal to compare the Label Propagation Approach with the other approaches investigated in this thesis.

The Label Propagation Approach consists of two major additional steps in comparison to the Synset Vector Approach. First, word embeddings models and existing thesaurus synsets are used to generate a partially labeled graph. Second, the graph is used as input to a label propagation algorithm that uses labeled nodes to label unlabeled nodes, i.e., to assign additional tokens to the existing synsets. The Synset Vector Approach is used as a baseline to evaluate the results of the Label Propagation Approach.

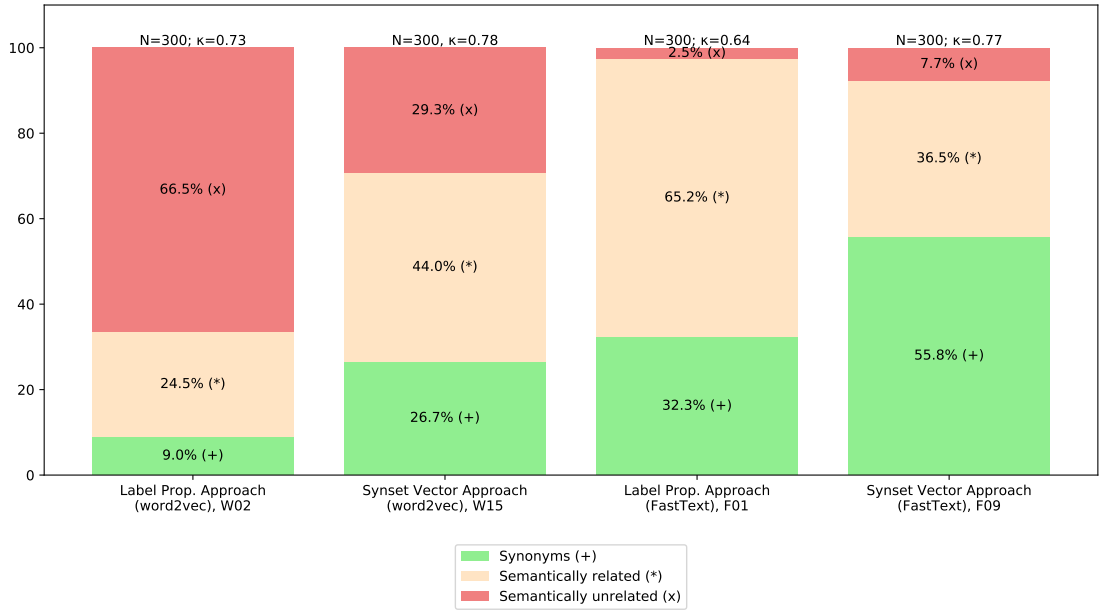

Figure 4.37.: Qualitative evaluation of the Label Propagation and Synset Vector Approaches.

Different algorithms and hyper-parameters for graph generation, two word embeddings algorithms and two label propagation algorithms are investigated in Mueller (2018). For the qualitative evaluation, the label spreading algorithm is used. For the label spreading algorithm, the α hyper-parameter is set to 0.2, the default value of the scikit-learn library's implementation. Further, 15 iterations of the label spreading algorithm are used. For the graph generation step, the k-nearest neighbor algorithm is used. For each node, 12 neighbors are calculated. The edges in the graph are weighted by the cosine similarity among the tokens. The edges are undirected and self-references are omitted. The full thesaurus is used as input to the graph labeling step. The tokens that are assigned to a synset by the label propagation algorithm come with a confidence score. The top ten candidate terms are selected by ranking the candidate terms according to the confidence score. Only synsets where at least ten tokens are assigned to are included in the qualitative evaluation.

The Label Propagation Approach takes word embeddings models as an input. The word2vec and FastText algorithms are used. The hyper-parameter configurations need to be selected. The hyper-parameter configurations, as well as the selected synsets for the qualitative evaluation, differ to the evaluation of the Synset Vector Approach. However, due to the results reported in Section 4.6.8, the results are still comparable. In contrast to the Synset Vector Approach that uses cosine similarity as a similarity measure, the label propagation algorithms output a significance score that is used to rank the candidate terms within a synset.

Figure 4.37 shows the results of the qualitative evaluation for the Label Propagation and Synset Vector Approaches with the word2vec and FastText algorithms. For both word embeddings algorithms, the Label Propagation Approach shows fewer good results than the corresponding Synset Vector Approach despite the larger computational effort. An analysis of the average occurrence frequency of the candidate terms in the training corpus does not give insights into the effects of applying label propa-

Approach (Word embeddings algorithm)	Average term occurrence frequency
Label Propagation Approach (word2vec), W02	1532.86
Synset Vector Approach (word2vec), W15	1681.35
Label Propagation Approach (FastText), F01	1357.59
Synset Vector Approach (FastText), F09	1843.62

Table 4.11.: Average occurrence frequency analysis for the Label Propagation and Synset Vector Approaches.

gation algorithms to this use case and dataset. Figure 4.11 shows the average occurrence frequency of the candidate terms of the qualitative evaluation in the training corpus. While the Label Propagation Approach with the FastText algorithm has a higher average occurrence frequency of the candidate terms, the values are not significant when compared to the average term occurrence frequencies of the Synset Vector Approach with the GloVe algorithm or the JoBimText Approach. Due to different evaluation thesaurus selections for the qualitative evaluation and because no patterns have been identified, neither the presentation of example candidate terms nor a shared candidate terms analysis are reasonable.

The poor results can be due to several reasons. It can only be speculated that either no global structure is present, that mathematical properties of global structures prevent label propagation algorithms from exploiting the global structure, see Yamaguchi and Hayashi (2017), or the graph generation needs to be varied as, for example, in Ravi and Diao (2016) that uses a much more complex graph generation procedure.

Initial experiments with other label propagation algorithms, for example, OMNIProp (Yamaguchi et al. (2015)) and CAMLP[11] (Yamaguchi et al. (2016)), also lead to very poor results.

In summary, despite promising results for similar tasks are reported in Ravi and Diao (2016), the Label Propagation Approach did not work for this use case and the given dataset according to the investigations carried out. The reasons for the poor results remain unclear, but several starting points are identified.

4.7. Discussion

In this chapter, the use case of thesaurus extension is investigated in depth. The use case of thesaurus extension is differentiated from the use cases of thesaurus construction and thesaurus reconstruction. The use case is highly relevant because many legal publishers use thesauri for query expansion. Several word embeddings based approaches are investigated to extend synsets of a German legal thesaurus. Experiments are conducted on a dataset of German tax law. Word embeddings based approaches are compared to traditional count-based DSMs.

[11] https://github.com/yamaguchiyuto/label_propagation, last accessed April 26, 2019

The thesaurus reconstruction use case is common in research. The use case of thesaurus extension requires qualitative evaluations that are difficult to conduct. Thus, little research on thesaurus extension has been carried out previously. Several challenges of the use case are identified. The scope of synsets and the border demarcation among neighboring synsets is difficult. Variations of word forms like different word spellings are often required to be reflected in thesauri. The German language frequently uses open compound words that are hard to identify. Fine-grained semantical differences of legal terms often matter for particular user inquiries. In German tax law, precise definitions of terms are used that also occur in the general natural language where the terms are defined less precise.

The JoBimText approach uses traditional count-based DSMs. The first, simple word embeddings based approach for thesaurus extension is the Single Word Approach that calculates candidate term lists for every term of an existing synset. The Synset Vector Approach calculates the mean of all word embeddings vectors of the input synset. The Intersection Approach intersects the candidate term lists calculated with the Synset Vector Approach. The Label Propagation Approach uses label propagation algorithms and word embeddings models to calculate a partition of all terms in vocabulary into synsets. The GloVe, word2vec and FastText word embeddings algorithms are considered.

The RP-Score is introduced as an alternative evaluation measure to the MAP relevance measure for hyper-parameter studies. The RP-Score measures the average ranking position distance among terms of an existing thesaurus. The RP-Score shows similar qualitative results like the MAP. However, the RP-Score is particularly well suited for the use case at hand. Further, the RP-Score is beneficial for smaller evaluation thesauri because more relations can be exploited in a more straightforward fashion than with the MAP relevance measure that requires train-/test-splits of evaluation thesauri. The RP-Score applies an equal weight to all ranking positions while the MAP puts a higher weight on top-ranked results.

For the GloVe, word2vec and FastText algorithms, exhaustive hyper-parameter studies are conducted. The Model Architecture, Iterations, Min-Count, Window Size and Vector Size hyper-parameters are identified as the most influential hyper-parameters. The FastText algorithm shows anomalies. The Skip-gram model architecture delivers the overall best results after a few iterations but performs inferior than the CBOW model architecture after around 45 training iterations. The Window Size hyper-parameter of the GloVe algorithm also shows abnormal behavior because no optimum is found for the Window Size hyper-parameter up to a value of 30.

The results of the qualitative evaluation of all investigated approaches and word embeddings algorithms are summarized in Figure 4.38.

All word embeddings based approaches can be combined with all word embeddings algorithms. The different word embeddings algorithms are compared using the Synset Vector Approach with quantitative and qualitative evaluations. A major result is that the FastText algorithm performs best in terms of quantity. In terms of quality, the FastText algorithm suggests mostly syntactically similar tokens. This can be explained by the construction of the FastText algorithm that calculates embeddings for character n-grams and subsequently accumulates all combinations of character n-grams contained in a token to represent a token. In contrast to that, the word2vec algorithm suggests mostly terms that are syntactically different but still semantically close. It could be argued that syntactically similar terms could be derived more easily by dictionaries or simple application of grammatical rules. However, the terms calculated with the FastText algorithm are in the training corpus and also include, for example,

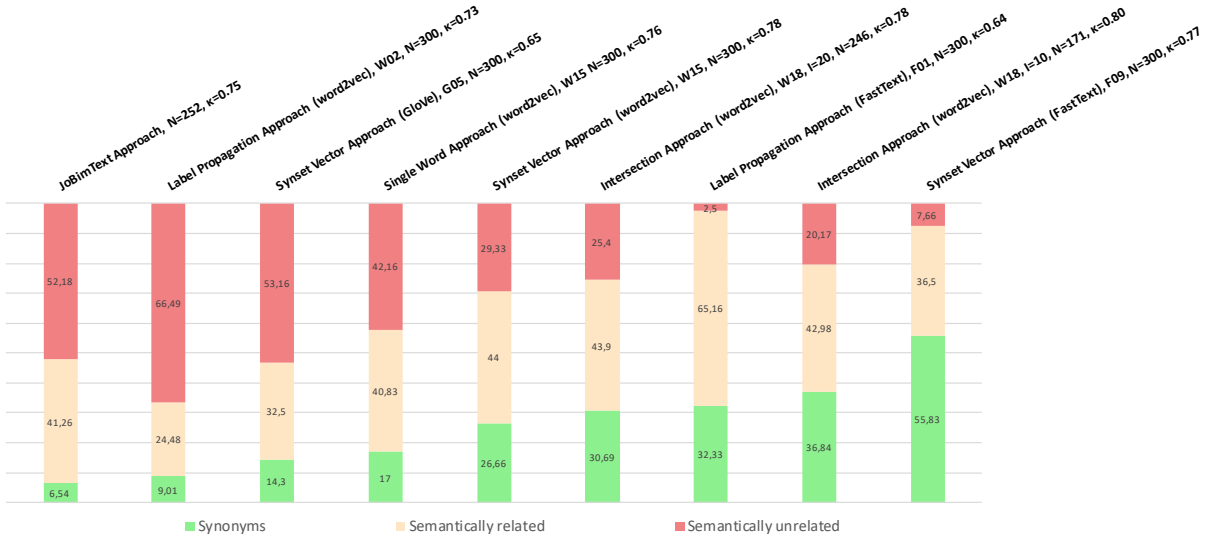

Figure 4.38.: Comparison of approaches and word embeddings algorithms.

spelling errors that could hardly be identified by more static approaches. The GloVe algorithm suggests qualitatively similar tokens like the word2vec algorithm. However, the GloVe algorithm cannot be recommended due to poor performance, needless complexity of the toolkit and lower robustness to hyper-parameter changes than the word2vec algorithm.

For the use case of thesaurus extension, very good results can be calculated with the Synset Vector Approach with the word2vec and FastText algorithms on commodity hardware in a few hours. Depending on the problem at hand, either the word2vec algorithm or the FastText algorithm should be selected. However, the results of the investigations suggest that the signals captured by word embeddings models are too noisy for full automation, i.e., the technical solutions described in this thesis require review by human experts. This result is underlined by the analysis of shared candidate terms among the approaches and word embeddings algorithms presented in Table 4.12. The different approaches and word embeddings algorithms have only a few suggestions in common, even for very similar approaches.

A major takeaway is that predictive DSMs like the word2vec and FastText algorithms perform significantly better than traditional count-based DSMs like the JoBimText or GloVe algorithms. This goes in line with other research results presented, for example, by Baroni et al. (2014) and Ramrakhiyani et al. (2015). An analysis of the average term occurrence frequency of the terms suggested by the algorithms in the qualitative evaluation suggests that count-based DSMs favor tokens that frequently occur in the training corpus, see Table 4.13. Vice versa, predictive DSMs (word embeddings) can compensate for the occurrence frequency of tokens in the training corpus significantly better than count-based DSMs.

The usage of the Synset Vector Approach is beneficial over the Single Word Approach because the Synset Vector Approach captures more complex semantics of synsets better. The Single Word Approach serves as a vehicle approach to compare the JoBimText and Synset Vector Approaches because of differently sized candidate term lists for existing synsets with more than one term.

All DSMs rely on statistical analysis. This poses a challenge to DSMs to derive semantics from infrequently occurring terms. Thus, the quality of word embeddings models heavily depends on the

	Synset Vector Approach (GloVe)	Single Word Approach (word2vec)	Synset Vector Approach (word2vec) C01	Synset Vector Approach (word2vec), C02	Intersection Approach (word2vec), K=20	Intersection Approach (word2vec), K=10	Synset Vector Approach (FastText)
JoBimText Approach	1.44%	1.99%	5.07%	2.89%	3.21%	3.30%	1.26%
Synset Vector Approach (GloVe), G05		9.66%	11.33%	13.00%	13.37%	12.95%	6.50%
Single Word Approach (word2vec), W15			16.33%	21.66%	20.14%	17.40%	6.16%
Synset Vector Approach (word2vec), W02				26.50%	25.27%	24.84%	7.66%
Synset Vector Approach (word2vec), W15					35.89%	32.27%	9.55%
Intersection Approach (word2vec), W18, K=20						40.52%	9.70%
Intersection Approach (word2vec), W18, K=10							9.55%
Synset Vector Approach (FastText), F09							

Table 4.12.: Shared candidate terms among approaches of the qualitative evaluation.

Min-Count hyper-parameter. Setting the Min-Count hyper-parameter to a value larger than one excludes words from the training procedure. As a consequence, the word embeddings vectors are more stable. A Min-Count hyper-parameter value of five (the default value of all considered word embeddings algorithms), in general, leads to stable word embeddings vectors. However, the down-side of setting the Min-Count hyper-parameter to a value larger than one is that not all tokens in the vocabulary are assigned a word embeddings vector. This is known as the out-of-vocabulary problem. The Intersection Approach can be used to filter noisy terms and to filter reasonable terms when infrequently occurring terms are included in the training of word embeddings models. The down-sides of the Intersection Approach are that the approach is computationally more expensive, comes with additional hyper-parameters and is not able to filter noisy signals in word embeddings models completely. Thus, the Intersection Approach can be used when precision is more important than recall.

The Label Propagation Approach is the computationally most expensive approach. The experiments

Approach (Word embeddings algorithm)	Average term occurrence frequency
JoBimText (-)	4338.48
Label Propagation Approach (word2vec), W02	1532.86
Synset Vector Approach (GloVe), G05	2476.80
Single Word Approach (word2vec), W15	1471.38
Synset Vector Approach (word2vec), W15	1681.35
Intersection Approach (word2vec), W18, K=10	1642.11
Label Propagation Approach, F01 (FastText)	1357.59
Intersection Approach (word2vec), W18, K=20	1575.08
Synset Vector Approach (FastText), F09	1843.62

Table 4.13.: Average candidate terms occurrence frequency in the training corpus.

conducted suggest that the label propagation technology is not well suited for thesaurus extension, at least, for the investigated dataset. None of the investigated variants of the Label Propagation Approach leads to improved results in comparison to the Synset Vector Approach. This is an unexpected result, given promising results are reported in the literature. Several possible reasons are identified.

An important result is that a quantitative evaluation, i.e., using an existing thesaurus as a "gold standard" evaluation set for thesaurus reconstruction leads to very small precision values. In contrast to that, the qualitative evaluation reveals that on average, up to 50% reasonable candidate terms are contained in candidate term lists with a fixed-length of ten candidate terms. This result underlines the assumption that a real-world thesaurus for a given real-world corpus is never "complete".

The validity of the results is assured by several measures. The used RP-Score evaluation measure is compared to the MAP relevance measure. Using several folds of train-/test-splits is shown to be unnecessary for the precision/recall and MAP relevance measures (for the investigated use case, dataset and evaluation setup). A convergence analysis of the qualitative evaluation shows that ten candidate terms for 30 synsets lead to stable results of the qualitative evaluation. A sensitivity analysis of the Synset Vector Approach with varying evaluation synsets selections and variations of the word2vec algorithms hyper-parameters shows that the results of the qualitative evaluation are reasonable and robust. A manual inspection of the candidate term lists strengthens the results derived from the qualitative evaluation.

The research presented in this chapter is subject to several limitations. The experiments are limited to the German language and the legal field of German tax law. Only synonym relations are considered while thesauri can contain other relationship types, too. Polysemes are not addressed specifically. Open compound words are widespread in German language but are difficult to deal with and are not investigated in this thesis. The word2vec toolkit comes with the phrase2vec command-line tool, see Mikolov et al. (2013c), that determines short phrases with statistical analysis. However, initial experiments showed that the results are very noisy and not suited for the use case. In general, it is difficult to generalize the results to other languages or other law domains.

No grid or random search is applied to the hyper-parameter studies. The effect of the Window Size hyper-parameter is not investigated in full depth. Goldberg (2016) suggests that different window sizes lead to different types of candidate terms. The effect of the window size on types of candidate terms is not investigated. Not all possible combinations of approaches to thesaurus extension and word embeddings algorithms are investigated. Word embeddings models are typically trained on very large corpora. Training corpus extension and leveraging pre-trained word embeddings models are not addressed. For the Intersection Approach, only the Iterations hyper-parameter as a form of varying hyper-parameter configurations is explored to some degree. Many other possible variations of the hyper-parameter configurations for the Intersection Approach are not investigated. Only a small number of label propagation algorithms is considered. The graph labeling step is carried out on a basic level. Advanced approaches to synset vector calculation that can leverage additional semantic resources, such as AutoExtend, are not considered.

Word embeddings could be used for other life-cycle activities around thesauri, for example, initial thesaurus creation or the merging of thesauri but have not been explored in this thesis. Especially the reconstruction of legal thesauri was investigated in research but not accomplished with promising results so far, cf. Vos (2017).

The very essence of instinct is that it's followed independently of reason.

Charles Darwin

CHAPTER 5

Implicit Query Expansion: Semantic Text Matching

In the previous chapter, the support of explicit query expansion approaches has been investigated, in particular by the extension of existing thesauri. Word embeddings based approaches for thesaurus extension have been investigated that exploit the similarity among word embeddings vectors of tokens. Additional semantic properties of word embeddings models are accessible via basic linear operations in word embeddings vector space, for example, addition. Thus, the focus of this chapter is to answer the question if vector space models that are based on word embeddings can be used to conduct query expansion in an automated and unsupervised fashion.

Information retrieval on text corpora is typically carried out with a vector space model. Vector space models use a way to represent text with vectors and use a text similarity measure among the vectors to retrieve relevant documents from a database. Several approaches that use word embeddings for query expansion have been investigated in research previously. A vast majority of the query expansion approaches investigates the expansion of terms on the query side of an information retrieval task. The query expansion approaches in this research expand terms on the query and document side at the same time. Therefore, an alternative text segment and document representation scheme that is based on the WE-DF text representations is investigated. The WE-DF text representation represents text segments by accumulating the word embeddings vectors of the composing tokens. Together with a text similarity measure such as cosine similarity the WE-DF text representation constitutes a vector space model that is also called WE-DF in this thesis. The name *WE-DF* follows the TF-IDF naming scheme because the Document Frequency (DF) of tokens can be seen as a weighting scheme and WE is short for word embeddings. The WE-DF vector space model has the potential to carry out an implicit form of query expansion, because similar tokens tend to have small angles between their word embeddings vectors. This also applies to accumulated word embeddings vectors, see Figure 5.1 for an example. Tokens of the text segment representation vector can be replaced with synonymous tokens implicitly.

A general problem that is called Semantic Text Matching is introduced. Semantic Text Matching is the

identification of semantically and/or logically related text segments. Semantic Text Matching problems are present in several legal use cases such as argumentation mining and information retrieval. Semantic Text Matching problems can be addressed with vector space models and are particularly well suited for approaches that use the WE-DF vector space model. Two representatives of vector space models that are based on word embeddings, in particular, the CHAPTER and the SENT Approaches are compared to the TF-IDF Approach. The TF-IDF vector space model serves as a baseline approach.

In this chapter, quantitative and qualitative evaluations are used. The approaches are compared with precision/recall relevance measures quantitatively. With quantitative evaluations, different preprocessing technologies are explored as well as variations of the approaches such as the extension of the training corpus for calculating word embeddings models.

The empirical part of the research explores an innovative use case. Lawyers shall select text segments in contracts, i.e., a natural language query. Text segments selected by the lawyers can be, for example, clauses in contracts. An information retrieval system shall recommend relevant chapters from German legal comments for the selected text segments. Legal comments are a concept specific to German law. Legal comments summarize statutes and court decisions. A dataset on German tenancy law is used. Clauses from six tenancy contract templates serve as query documents. Three legal comments provide condensed information on German tenancy law. Several challenges for the use case are identified.

The Keyword Search is a search method that is very common for legal information retrieval systems. Users enter keywords into a search box. Selecting proper keywords is part of the knowledge of experts. The Selection Search is an alternative search method where users select an (arbitrary length) text segment as input to an information retrieval system. Both search methods as well as the different approaches are implemented in a client/server information retrieval system with two frontends: A web application and a Microsoft Word AddIn. The two search methods and the approaches are evaluated in a user study with lawyers. For the user study, the Microsoft Word AddIn is used.

Parts of the research presented in this chapter have been previously published in Landthaler et al. (2016), Landthaler et al. (2018b) and Landthaler et al. (2019).

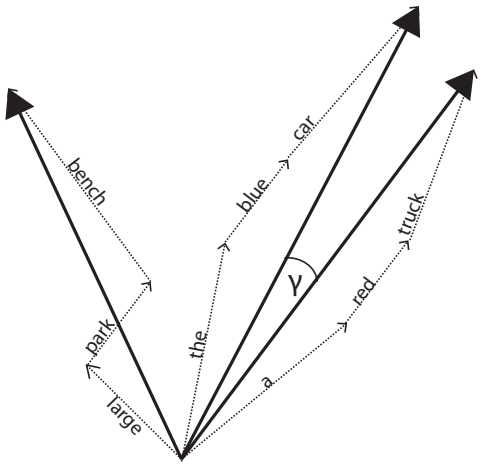

Figure 5.1.: Implicit query expansion with the WE-DF vector space model (Landthaler et al. (2016)).

5.1. Implicit Query Expansion

The goal of query expansion is to improve the recall of information retrieval systems while, in the best case, preserving a high precision. In practice, the terms of a user query are expanded so that synonymous terms are considered as alternative terms to the terms in the input query. A straightforward approach is to use a thesaurus to expand query terms. Query expansion approaches that use thesauri can be imagined to pose several queries to the vector space model where for each query terms are replaced with synonymous terms from the thesaurus. The thesaurus can be controlled. Thus, query expansion approaches that use thesauri can be considered as an explicit form of query expansion.

The main idea of the research presented in this chapter is based on the hypothesis that an implicit query expansion could be conducted by using word embeddings based approaches as alternative vector space models, i.e., vector space model that due to their properties implicitly conduct a form of query expansion. Many other approaches expand the user query. From this perspective, the investigated approach conducts a query expansion on the query side as well as on the document side because both queries and documents are encoded with the WE-DF text representation. This approach conducts query expansion in an automated way. However, it can not be controlled in way a thesaurus can be controlled.

A large variety of approaches to automate query expansion exists. Nowadays, many approaches attempt to leverage word embeddings. However, other approaches typically focus on replacing individual words, cf. Azad and Deepak (2019). For example, Ganguly et al. (2015) extend the probabilistic language model. All tokens of a query and all tokens of the documents are replaced based on word embeddings with a certain probability. Other research endeavours investigate differently weighted vector space models such as WE-DF, too. However, the WE-DF vector space model is not investigated on its capability to conduct an automated form of query expansion.

The universe of different possible approaches to achieve an implicit form of query expansion is vast, including approaches that use the WE-DF vector space model. One approach called Full Text Search Extension (FTSE) is presented by Landthaler et al. (2016). The acFTSE approach is inspired by the semantic relations that can be carried out with linear operations in word embeddings model's vector space. Another inspiration is the frequently occurrence of slightly different formulations of phrases in legal documents such as contracts, judgments or legislative texts. The idea of the acFTSE approach is to take an input query and to encode it as the sum of its composing word embeddings vectors. Token n-grams of the documents of a given corpus are encoded with the WE-DF text representation. The token n-gram embedding vectors are pooled and ranked against the input query vector. The acFTSE approach can also be considered as a form of a phrase analogy task or paraphrase identification, cf. Socher et al. (2011).

In Landthaler et al. (2016), the idea is proposed that users select an arbitrary length text segment of a given document as input to the search. The approach is feasible at least for small corpora when only a subset of all possible token n-grams is considered. Similar to the word2vec algorithm, the acFTSE approach uses a sliding window approach to select a subset of token n-grams of the corpus. The sliding window is shifted over the corpus by half the size of the input query. The size of the input query is determined as the number of tokens, not the number of characters. The most similar phrases are identified by a ranking of all documents against the query vector using cosine similarity. Only after the identification of candidate token n-grams, a more fine-grained refinement step is carried out. A sliding

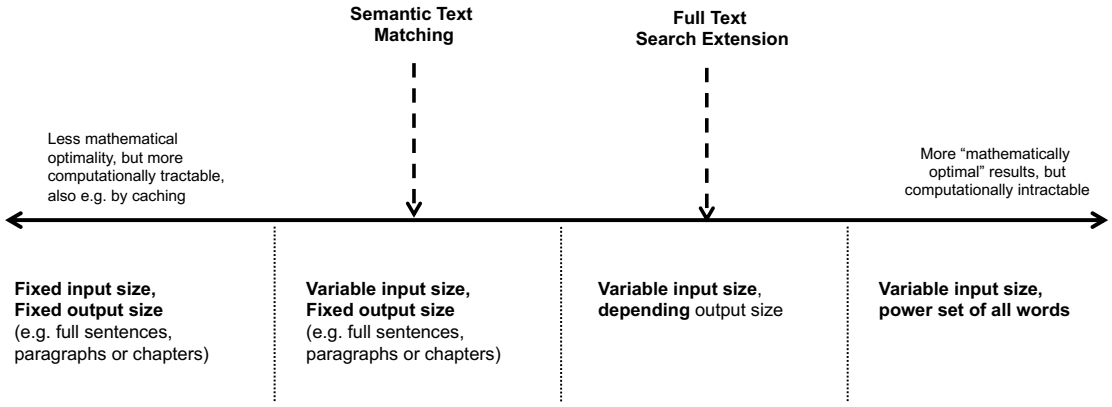

Figure 5.2.: A spectrum of WE-DF vector space models for implicit query expansion.

window with a step size of one token is moved over an area of three times the window size around the identified candidate token n-grams. The acFTSE approach has several limitations. The window size depends on the input query and therefore, no caching of the token n-grams is possible in advance. While caching basically is possible, it requires unfeasible amounts of memory. The corpus is only searched for phrases of the same length as the input query. However, humans think in semantically enclosed units such as sentences and not in fixed-length text segments. While the results of the case studies on GDPR and GCC yield promising results, the results also already showed a large amount of noise is contained in the results. Two possible reasons are the noise inherent in word embeddings models but also the many free dimensions in the language, i.e., the large number of possibilities to select and arrange words and to select grammatical variants.

To identify all possible token n-grams, or phrases, it would be necessary to calculate the power set of all token n-grams (including token n-grams of varying size n). Even for very small corpora, this is technically not feasible because the power-set size grows exponentially. Reasonable strategies to select subsets of the power-set need to be considered. Figure 5.2 illustrates a spectrum of possible problems to use word embeddings based approaches that could conduct implicit query expansion.

The acFTSE approach suffers from the dependency on the dynamic input query size. The other end of the spectrum constitutes a problem class with fixed input query sizes and fixed target phrase sizes. This type of constellation rarely occurs in real-world problems. Another problem class constitutes the Semantic Text Matching problem that allows dynamic input query sizes, but the corpus is represented with text segments that are identified in advance. To some degree, this problem class defers the problem of identifying text segments to a reasonable text segmentation of the corpus or its documents. Semantic Text Matching problems are a promising problem class and are discussed in more detail in the next section.

5.2. Semantic Text Matching

In the previous section, Semantic Text Matching problems are identified as a promising problem class for further investigations. In the remainder of this thesis, the focus is on Semantic Text Matching problems.

For this purpose, Semantic Text Matching is defined as the problem to identify semantically and/or logically related text segments in different documents (potentially of different document types). An example for a Semantic Text Matching problem in the German legal domain is that contract clauses need to satisfy the restrictions of certain judgments or provisions. Figure 5.3 illustrates the Semantic Text Matching problem in an abstract form. Semantic Text Matching requires text segments as input. The segmentation of texts into text segments is not part of the Semantic Text Matching problem.

The Semantic Text Matching problem is an abstract problem. It is not tied to a particular application domain. However, Semantic Text Matching problems often occur in the legal domain. For example, a sub-task in argumentation mining is to match premises and claims. For the matching of premises and claims, word embeddings have been used previously, see Section 3.1. The Semantic Text Matching problem is related to the textual entailment problem. Textual entailment also attempts to solve the matching task of claims and premises in argumentation mining. However, textual entailment specifically attempts to identify *implication* relations among text segments. The Semantic Text Matching problem is less strict on the relation type between text segments than textual entailment, cf. Dagan et al. (2005).

The Semantic Text Matching problem can be viewed from different perspectives: The network view, the graph view, and the information retrieval view. Citation networks can be seen as explicit networks in a sense that references point to other documents or text segments in the same document (internal references) or other documents (external references). Analog to these explicit references, the Semantic Text Matching problem can be seen as the task to identify implicit references among text segments (or

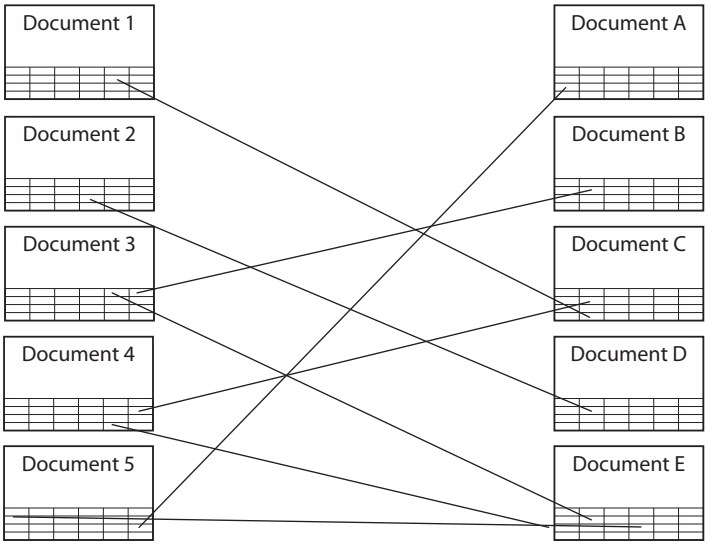

Figure 5.3.: Abstract Semantic Text Matching problem (Landthaler et al. (2018b)).

documents) where references (or relations) are not stated explicitly in the text. Landthaler et al. (2015) and Waltl et al. (2016) identified different types of references present in and among legal documents and differentiate among explicit and implicit references.

Explicit citation networks constitute a graph and can be visualized as such. Equivalently, the links of implicit references form a graph. In the case of two distinct text segment types, for example, text segments from two different document types, the graph is bipartite. This exemplifies the matching character of Semantic Text Matching. Furthermore, edges can be binary or weighted.

The Semantic Text Matching problem also constitutes a special form of an information retrieval task. In the case of two or few document types, the Semantic Text Matching problem is a more restricted form of (general) information retrieval. The more different document types are involved, the more the Semantic Text Matching problem approaches (general) information retrieval. However, the Semantic Text Matching problem includes a notion of text segmentation that is usually not modeled in information retrieval. The restriction to two or few document types has the benefit that side-effects that appear when many different document types are included can be reduced or eliminated.

Due to the relation to information retrieval, one approach to solve Semantic Text Matching problems is the application of vector space models. Vector space models use some form of text representation, for example, TF-IDF. Vector space models, in general, use text similarity measures to rank documents of a database against a user query. In general, a user query can be a search query crafted by the user but also documents. Here, manually or automatically crafted search queries are considered. In the case of documents that consist of several text segments, the text segments need to be identified first. In this case, the text segmentation is a challenge on its own. In some cases, a corpus comes with pre-segmented documents into text segments such as chapters, paragraphs with marginal numbers or clear indicators of semantically enclosed text segments. Otherwise, heuristics such as splitting at newline characters must be applied. The Semantic Text Matching problem is then solved so that a set of documents or text segments from the corpus is pooled. In the case that a vector space model is used, the text segments of the pool are ranked with a text similarity measure and the top-ranked results are hopefully relevant candidates that have a link to the selected text segment.

5.3. Use Case

An innovative use case is investigated. Contracts need to comply with the law. Typical contract management life-cycle activities include the initial creation and maintenance of contracts over time. New regulations, court decisions or risk assessments can be the triggers for contract analysis and contract editing. Neither the initial creation nor the analysis of contracts is a linear process, cf. Landthaler et al. (2018b). For example, clauses depend on each other, and lawyers need to jump between different clauses of a contract. Also, legal information retrieval is often not integrated in text processors and lawyers need to switch between different applications. The idea of the use case is to recommend tailored information from legal corpora to text segments selected by users from contracts, for example, complete clauses. This search method, called *Selection Search* in the following, can be seen as an alternative or complementary human-computer interaction method to the traditional Keyword Search. Optimally, the information retrieval system is tightly integrated into lawyers' daily workflows. The use case is an instance of a Semantic Text Matching problem as can be seen in Figure 5.4.

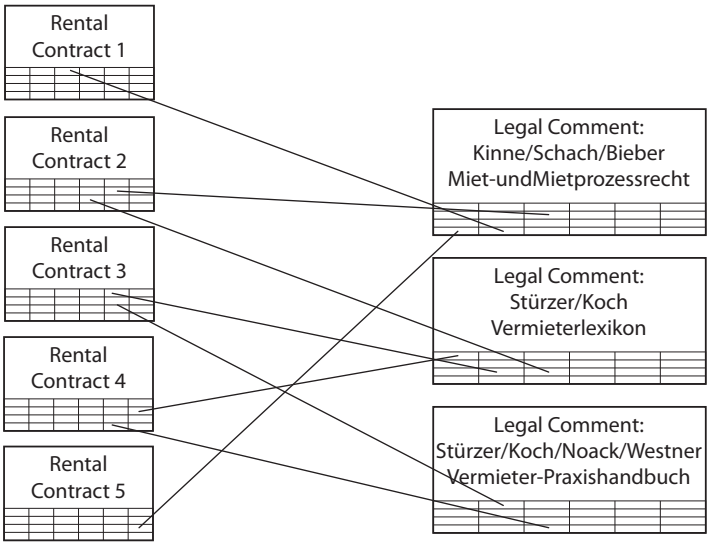

Figure 5.4.: Semantic Text Matching problem instance of the use case (Landthaler et al. (2018b)).

The use case is relevant for both lawyers and legal publishers. Legal publishers, on the one hand, face the challenge of an ever-growing amount of information in their databases.[1] This implies a follow-up challenge for legal publishers to provide their customers (for example lawyers) tools and means to access large knowledge databases. For lawyers, on the other hand, the investigated use case can improve the efficiency due to faster means of handling a search tool by selecting text rather than typing keywords and due to the possibly faster access to relevant information.

Two document types are considered in the use case. Template tenancy contracts are widely used in Germany due to the many regulations that apply to tenancy contracts. However, the regulations for template contracts are even stricter than the regulations for individual agreements. Legal comments are a document type that is specific to the German legal sphere. Legal comments contain condensed information from statutes and court decisions.

The use case is evaluated on six German template tenancy contracts and three legal comments on tenancy law. Legal information retrieval and thus also the investigated use case pose challenging problems. A selection of challenges are identified in the next section.

5.4. Use Case Challenges

Several challenges have been identified for the use case of recommending legal comment chapters to selected text segments from German tenancy template contracts. The challenges can be grouped into user, domain-specific and technical challenges.

1. **User intention challenges:** The user intention of a search process can vary a lot. This applies to general information retrieval and has been subject to much before, see, for example, Manning

[1] Very old legal documents are still relevant, for example, in tax law, documents from the Weimar Republic are still relevant today.

et al. (2008) or Hiemstra (2001). Likewise, for the investigated use case, the scope of a search query can vary heavily. A legal expert can seek general information about a legal concept or a user can seek information that is specific to a legal case, for example, on text-segment-specific factors. Here, factors are meant in the sense of Grabmair et al. (2015). For (legal) information retrieval systems, it is very difficult to guess the user intention through simple text. The information retrieval systems considered here act as a one-way street for search intentions. It is possible but challenging to construct a system that interacts with the user, for example, if further inquiries would be necessary to understand the user's search intention. A class of systems that interact with users are called dialogue systems and could be used here, too.

2. **Domain-specific challenges:** The use case can be seen as a legal information retrieval task. Therefore, the use case is potentially subject to all legal data specifics that are relevant for legal information retrieval, cf. Section 3.1. In this research, several specifics to legal data are addressed. The main focus is on the legal language, especially the usage of synonyms (implicit query expansion). The use case addresses to some degree the heterogeneity of legal document types because contracts and legal comments are involved. Indirectly, law and court decisions are considered because they are source documents for legal comments. The approaches addresses the size of the legal documents. A domain challenge that arises from the legal field of tenancy law are the implications of summation effects. Several contract clauses that are legal on their own can become illegal when combined in one contract.
3. **Technical challenges:** The use case is subject to general NLP and technical challenges such as the selection of proper pre-processing steps or choosing the right technologies. Two challenges are special to the use case. First, a limitation of the Selection Search is that users can not freely choose keywords but are dependent on an existing text that can be selected. While it is possible to enter or edit keywords as input to the Selection Search, this type of usage is not considered specifically. Second, many phrases in contracts are slight variations of natural language. However, slight variations in natural language can still lead to large differences in semantics, cf. Section 4.2.

Several legal data specifics that are relevant for information retrieval such as citations or temporal aspects of legal documents could be addressed for the use case but are neglected.

5.5. Non-functional Requirements

Information retrieval systems are subject to many non-functional requirements. The real-world application of the use case in an organizational context requires many additional non-functional requirements that are not addressed in this thesis. To carry out the research, one of the most important non-functional requirements is the response time. It is well known that users expect an answer of a search system within a few seconds, see, for example, Arapakis et al. (2014). The implemented system uses established open-source information retrieval software to satisfy this requirement for the baseline approach for the user study. The response time is a critical aspect for all considered word embeddings based approaches. The nmslib, see Appendix A, provides approximate k-nearest neighbor methods that are successfully applied to satisfy the response time requirements.

Another essential non-functional requirement is the quality of the search results. The quantitative

evaluation is carried out to select approaches for the qualitative evaluation (user study) but also to investigate pre-processing options and variants of the approaches. The goal of the user study is to find out how legal experts perceive the quality of the search results for the selected approaches.

For the user study, privacy requirements are an issue because the GDPR[2] became effective in Europe before the user study was started. For the user study, the participants were instructed to use documents without privacy-relevant information. For a practical application of the use case, this requirement needs to be treated carefully.

Other in parts realized, but less important non-functional requirements can include the scalability, the usage of open-source software, fault tolerance or robustness of the system to erroneous user inputs, ease-of-use, and the comprehensibility of the search functionality and the graphical user interface. Many of these non-functional requirements are important for production ready applications.

5.6. Technical Approaches

Semantic Text Matching problems can be approached with vector space models. Vector space models use a text representation where each text segment or document is represented as a single vector. Results for user queries are calculated with a ranking function such as cosine similarity. Two approaches investigated in the following are based on word embeddings. The CHAPTER Approach and the SENT Approach represent documents with the WE-DF text representation. The hypothesis is that WE-DF vector space model can conduct implicit query expansion. The traditional TF-IDF vector space model serves as a baseline. The different approaches are called search technologies, in contrast to Selection Search and Keyword Search that constitute different human-computer interaction methods. Thus, Selection Search and Keyword Search are called search methods in the following. The different approaches (CHAPTER, SENT and TF-IDF) are discussed in this Section in more detail.

5.6.1. WE-DF: CHAPTER Approach

The CHAPTER Approach represents a legal comment chapter as a single vector using the WE-DF text representation. The CHAPTER Approach is inspired by the semantic operations that can be computed as linear vector operations in word embeddings model's vector space. According to Mikolov et al. (2013c), two relevant semantic operations are synonymy and compositionality. The WE-DF vector space model has been used for paraphrase detection previously. The different approaches can be described by pipes and filters architectures. In Figure 5.5, the training of word embeddings models from the given corpus is displayed. Figure 5.6 illustrates how the CHAPTER Approach is applied to the dataset. Word embeddings models are calculated with a training corpus. In the standard variant, all pre-processed texts from the contracts and the legal comments are used as the training corpus. Afterward, a text segment in a contract, for example, clauses, and the legal comment chapters is represented by a single vector through the accumulation of word embeddings vectors of the composing tokens.

The input query, for example, contract clauses, are encoded with the WE-DF text representation using the previously trained word embeddings model, too. A ranked list of the most similar documents for the

[2] https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX%3A32016R0679, last accessed January 10, 2019.

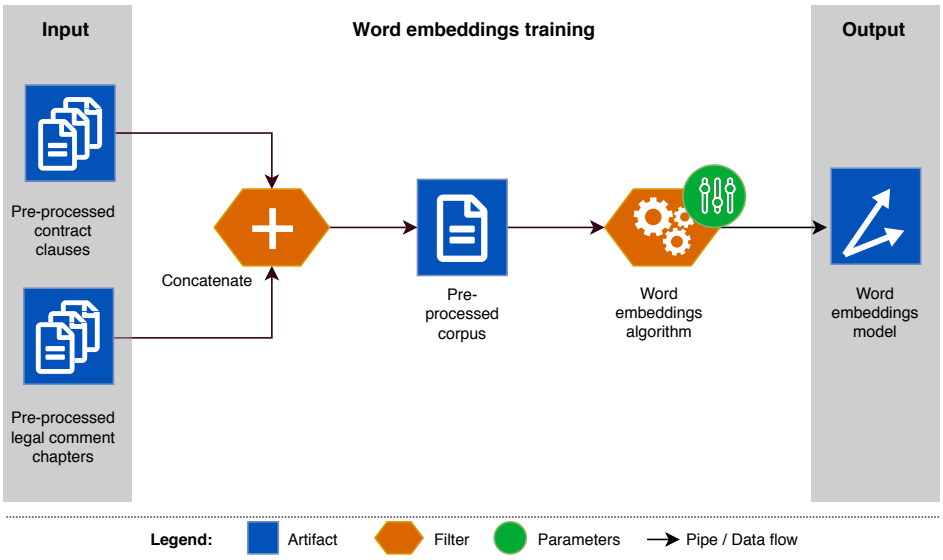

Figure 5.5.: Training pipeline for word embeddings algorithms.

input query is obtained by calculating the most similar vectors from the pool of documents to the input query vector. In the use case at hand, the pool of documents consists of the legal comment chapters. The CHAPTER Approach can be carried out with any word embeddings algorithm. Here, the word2vec and FastText algorithms are considered. The CHAPTER Approach depends on the hyper-parameters of the word embeddings algorithm. The CHAPTER Approach can be implemented efficiently by using a vector indexing method, for example, by the approximate nearest neighbor search provided by the nmslib, see Appendix A. For the quantitative evaluation, the complete ranked lists of all legal comment chapters are required. The index of the nmslib can be thought of as a bucket search approach, i.e., only a limited number of results per query can be calculated efficiently. Thus, the nmslib is not used for the quantitative evaluation, but only for the implemented systems that are accessed by users.

The cosine similarity of two vectors is not affected by the norms of the vectors. However, the WE-DF vector space model that uses accumulations of word embeddings vectors is sensitive to the norms of the individual word embeddings vectors. Normalizing the word embeddings vectors to unit length or not leads to different results.

The input query can contain words where no word embeddings vector exits in the word embeddings model, especially if the word embeddings model is pruned to contain only tokens of a certain minimal occurrence frequency (out-of-vocabulary problem). A straightforward solution to the out-of-vocabulary problem is to ignore tokens where no word embeddings vector exists. Another approach would be to use the FastText algorithm that can be used to estimate word embeddings vectors from character n-gram embeddings (think of "syllable embeddings").

The name CHAPTER Approach reflects that a complete legal comment chapter is represented by a single vector. The WE-DF vector space model is an approach that is based on the BOW assumption because the summation of word embeddings vectors ignores the word order. However, word embeddings vectors encode the word order to the extent of the window size used in the word embeddings algorithms from the full training corpus. In comparison to the TF-IDF vector space model, the vectors

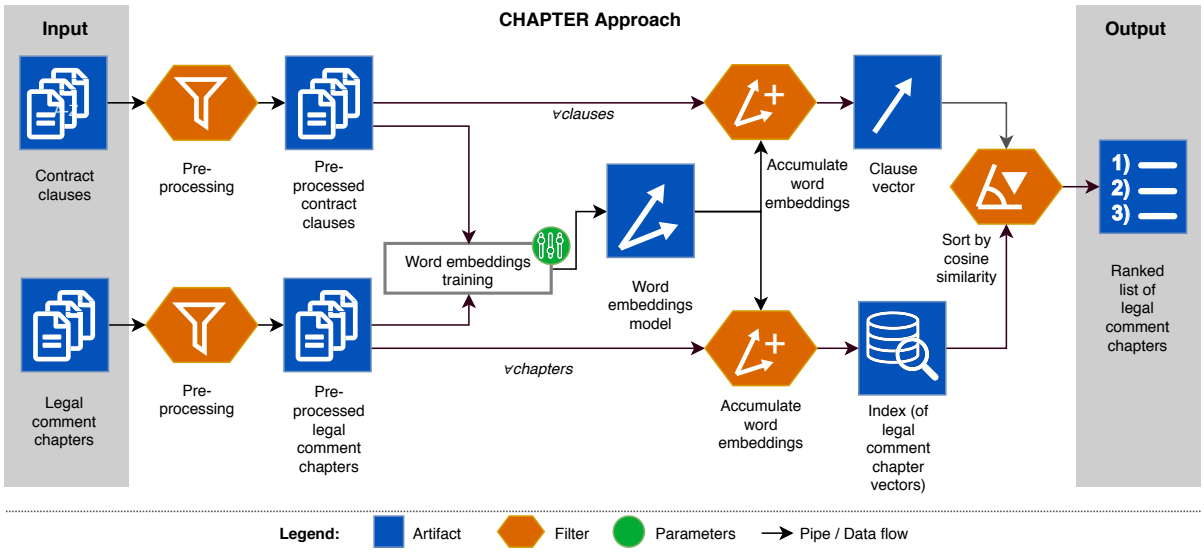

Figure 5.6.: Processing pipeline of the CHAPTER Approach.

that represent text segments or documents with the WE-DF text representation have a much smaller dimensionality. It could be expected to some degree that large documents cannot be properly represented because the summation of many vectors might lead to a loss of information encoded in the vectors.

5.6.2. WE-DF: SENT Approach

The SENT Approach can be considered as a variant of the CHAPTER Approach. Similarly to the CHAPTER Approach, the SENT Approach is based on the WE-DF vector space model. The input query is calculated equivalently to the CHAPTER Approach, i.e., encoded with the WE-DF text representation. The main idea of the SENT Approach is to segment the legal comment chapters further into sentences. The sentences are represented with the WE-DF text representation. Thus, the vectors that represent sentences can be called sentence embeddings. A set of sentence embeddings then represents a complete legal comment chapter. One motivation for the SENT Approach is that the input queries are typically much smaller than the legal comment chapters. To compare more equally sized text segments in the ranking step, sentences could be a suitable text segment size. The SENT Approach could further reduce the loss of information effects of aggregating too many word embeddings vectors to a single vector. Moreover, the legal comments contain sample formulations for contract clauses. The SENT Approach could be especially useful in the considered use case because contract clauses are often very similar to the sample formulations.

Figure 5.7 shows the application of the SENT Approach to the dataset. The sentence segmentation is carried out with spacy library, see Appendix A. However, the sentence segmentation of legal documents is a difficult problem. The sentence segmentation algorithm of the spacy library stumbles upon references. Often, a sentence segment consists of a few characters or tokens only. To mitigate these effects, the segmented sentences are post-processed by adding few character "sentences" to larger sentences with a simple count-based heuristic. While this post-processing does not lead to a high-quality

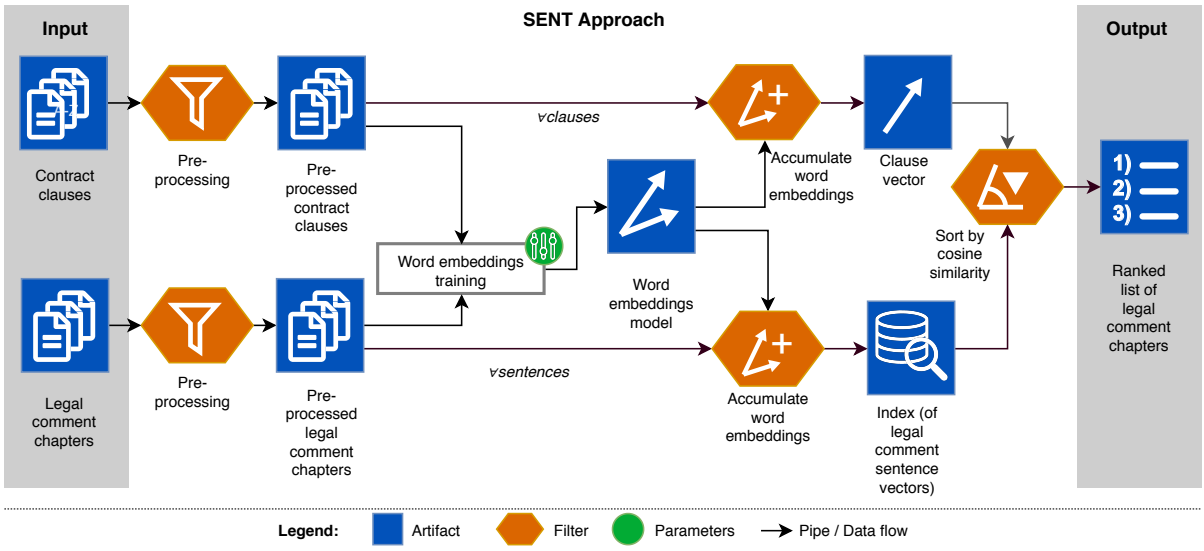

Figure 5.7.: Processing pipeline of the SENT Approach.

sentence segmentation, the central aspect of the sentence segmentation step is to segment the legal comment chapters into smaller text segments. The quality of the sentence segmentation is not crucial for the experiments conducted in this thesis.

5.6.3. TF-IDF Approach

The TF-IDF Approach uses the traditional TF-IDF vector space model, see Section 3.3.1. The TF-IDF vector space model is a classical approach that is based on the BOW assumption. The word order is completely lost. Nevertheless, the TF-IDF vector space model is used frequently and often leads to good results. Thus, the TF-IDF Approach is selected as a baseline approach. Figure 5.8 illustrates the processing pipeline applied to calculate a ranked result list of legal comment chapters for an input query.

Two TF-IDF vector space model implementations are considered: The TF-IDF implementation of the gensim library and Elasticsearch's *More like this* functionality. The gensim implementation of TF-IDF allows for greater control over the pre-processing. Elasticsearch uses an internal pre-processing. The gensim implementation of TF-IDF and Elasticsearch's *More like this* are compared in the quantitative evaluation. For the user study, only Elasticsearch's *More like this* is used. The benefit of Elasticsearch is that it is optimized for high performance (and delivers higher quality results).

5.7. Dataset

The dataset is specifically composed for the use case. Six German publicly available tenancy contract templates for private premises are collected from the web. Three German legal comments on German tenancy law are provided by the industry partner Haufe Group: Kinne et al. (2012), Stürzer and Koch

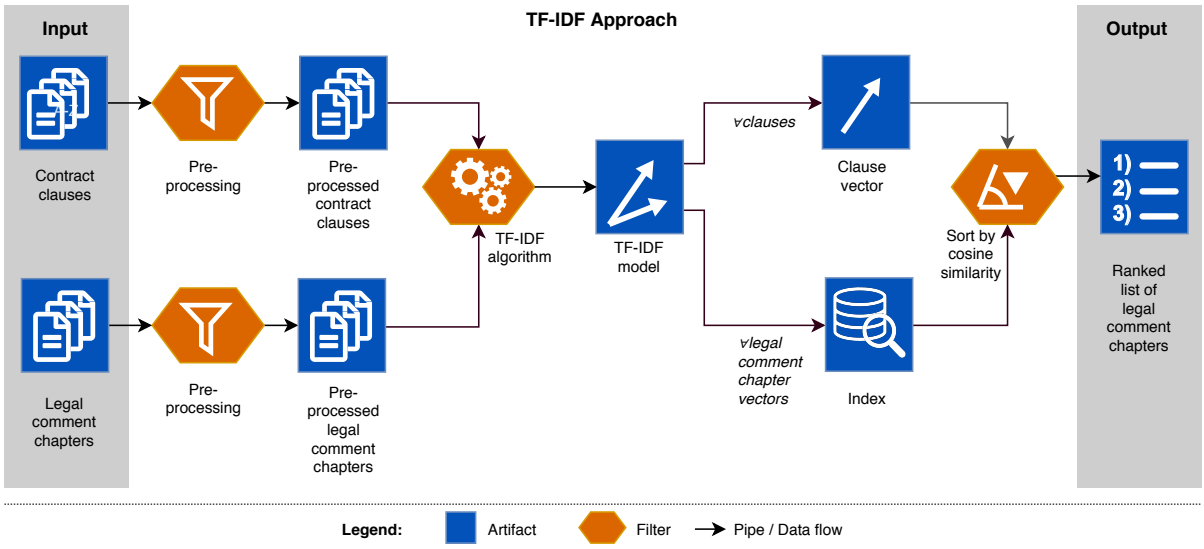

Figure 5.8.: Processing pipeline of the TF-IDF Approach.

Contract	#Clauses	#Words	Words per clause	#Tags
1	146	5354	31.71	195
2	80	2267	20.85	120
3	75	2234	31.18	115
4	90	2609	30.73	134
5	53	1304	30.40	73
6	104	3618	31.71	155
Total	**548**	**17386**	**31.09**	**792**

Table 5.1.: Tenancy contract statistics (Landthaler et al. (2018b)).

(2017) and Stürzer et al. (2015). The contracts differ in their length and coverage of different tenancy law aspects, see Table 5.1. Examples for clauses extracted from the contracts are displayed in Table 5.3. The legal comments differ in their lengths in terms of chapters, the total number of words and the average number of words per chapter, see Table 5.2. While all legal comments are in German language and on the subject of German tenancy law, they differ in their intended audience: Kinne et al. (2012) is intended for professional legal experts and difficult to read for non-lawyers. This legal comment has the most chapters and also has the longest chapters in terms of the number of words. Stürzer et al. (2015) is on the other side of the spectrum of the level of expertise and intended primarily for professional landlords. The legal comment has much fewer chapters than the first legal comment and the chapters are much shorter in terms of the number of words. Stürzer and Koch (2017) can be considered close to the first legal comment in terms of intended expertise level. All legal comments contain information relevant to clauses in tenancy contracts.

Doc.	Name	#Chapters	#Words	Words / chapter
1	Miet- und Prozessrecht	734	510129	283.08
2	Vermieterlexikon	890	462265	256.52
3	Vermieter-Praxishandbuch	178	99569	55.25
Total		**1802**	**1071963**	**198.28**

Table 5.2.: Legal comments corpus statistics (Landthaler et al. (2018b)).

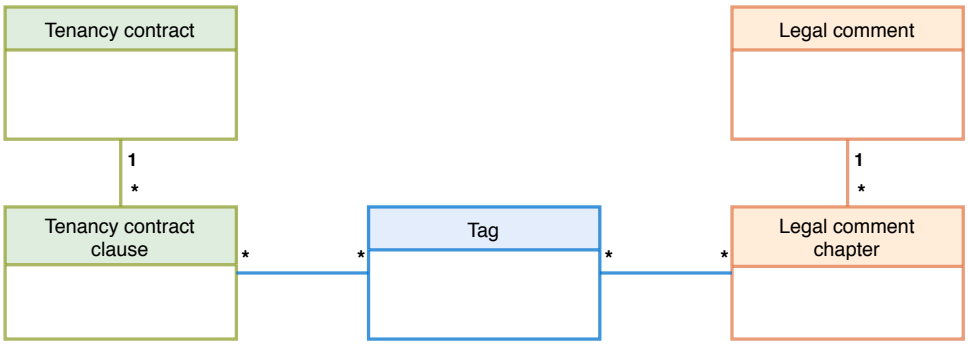

Figure 5.9.: Dataset model (Landthaler et al. (2018b)).

5.8. Evaluation Set Construction

For the quantitative evaluation, two evaluation sets are constructed. Both evaluation sets connect clauses in contracts and chapters in legal comments on the German tenancy law sub-topic *cosmetic repairs*.

The regulations for German tenancy contracts based on contract templates are complex. Conducting cosmetic repairs is by default duty of the landlord. The landlord can transfer the duty to carry out cosmetic repairs to the tenant. For the effectiveness of passing the duty to the tenant, many different details need to be considered. Slight variations of phrasings easily lead to ineffectiveness of clauses in a template tenancy contract. Certain combinations of individually effective clauses may lead to ineffectiveness due to accumulation effects. A large number of such legal details are discussed in the legal comments. However, the information is often spread over several chapters.

To simplify the process and to scan each contract and legal comment only once, tags are used as an intermediary between contract clauses and legal comment chapters. Figure 5.9 illustrates the data model. Firstly, the contracts are scanned and regulatory content that is present in at least two of the contracts is identified. The contracts are fully tagged and yield 214 different labels. One clause can be tagged with several tags because different regulatory contents are spread over two sentences in one contract but covered in one sentence in another contract, see Table 5.3 for an example. Nine of the 214 tags are part of the topic *cosmetic repairs*: *Landlord duty*, *Tenant duty*, *Scope*, *Periods*, *Period start*, *Kind & quality*, *Period deviation*, *Contract end* and *Costs*. 37 contract clauses are tagged with cosmetic repair tags, see Table 5.4 for more details.

All legal comment chapters are tagged regarding the nine cosmetic repair tags. For tagging the legal comments, first, a case-insensitive keyword search for the term "schönheitsreparatur" ("cosmetic re-

Contract	Clause text	Tags
4	Der (Die) Vermieter __ wohnhaft in __ und der (die) Mieter __ schließen folgenden Mietvertrag: (The landlord __ residing in __ and the tenant __ conclude the following tenancy contract:)	- Contracting Parties
2	Die regelmäßigen Schönheitsreparaturen während der Mietzeit übernimmt auf eigene Kosten der Mieter. (The regular cosmetic repairs during the rental period have to be done by the tenant on his own cost.)	- Tenant Duty
3	Während der Mietzeit verpflichtet sich der Mieter, auf seine Kosten erforderliche Schönheitsreparaturen fach- und sachgerecht durchzuführen. (During the tenancy period the tenant feels obliged to do the cosmetic repairs on his own cost in a proper and appropriate way.)	- Tenant Duty - Kind and Quality
1	Die Schönheitsreparaturen müssen fachgerecht ausgeführt werden. (The cosmetic repairs have to be carried out in a proper way.)	- Kind and Quality

Table 5.3.: Sample clauses from tenancy contracts and tags (Landthaler et al. (2018b)).

pair") is conducted that results in 159 chapters. The legal comment chapters are manually labeled in a "broad" and a "narrow" fashion. The narrow tagging includes chapters where a phrasing example for the phrasing of the particular tag is present, the full chapter covers the topic of the tag or the chapter contains single lines that cover highly relevant aspects for the particular tag (and for the respective contract clauses). In the broad tagging, additional chapters are included that contain indirectly or less critical aspects for a particular tag. For example, in the broad tagging, the legal comment chapter that informs about the default scope of cosmetic repairs is included for the tag "passing of the duty". If a "passing of the duty" clause is included in a contract, it might be relevant to know about the implications of not specifying the scope of this duty. In total, this leads to 582 relations between contract clauses and the legal comment chapters for the narrow evaluation set. The tagging for the broad fash-

Cosmetic repair tag	Contract clauses	Legal comment chapters narrow fashion	Legal comment chapters broad fashion
Landlord duty	2	20	40
Tenant duty	6	37	77
Scope	6	16	24
Periods	4	12	17
Period start	3	5	7
Kind & quality	4	14	17
Period deviation	3	9	10
Contract end	8	10	20
Costs	1	4	11
Total	**37**	**127**	**223**

Table 5.4.: Labeling statistics of cosmetic repair tags.

	Narrow evaluation set			Broad evaluation set		
Name	**#Matches**	**Vocabulary size**	**#Tokens**	**#Matches**	**Vocabulary size**	**#Tokens**
Landlord duty	40	3542	45705	80	6354	106325
Tenant duty	222	5531	272160	456	10510	707433
Scope	90	3197	118264	138	5587	250028
Periods	48	3583	99293	68	4497	139313
Period start	15	2637	42889	21	2969	54664
Kind & quality	56	3252	74237	68	3658	89569
Period deviation	27	3047	58183	30	3430	73444
Contract end	80	3080	155615	160	4527	285621
Costs	4	1052	3709	11	3151	19448
Total	**582**	**6750**	**870056**	**1032**	**11291**	**1725846**

Table 5.5.: Linguistic statistics of cosmetic repair text segments.

ion yields 1032 relations. Table 5.5 shows the number of relations (matches), the number of tokens and the vocabulary size for the nine individual cosmetic repair tags.

The dataset cannot be published due to the copyright protection of the texts. However, the process described to construct the evaluation sets can be seen as a way to efficiently construct evaluation sets for legal information retrieval tasks given certain conditions, in particular, when tags can be derived which is not possible in every case.

5.9. Pre-Processing

The six German tenancy contract templates are obtained in the PDF format from the web. All clauses are manually extracted to ensure a high-quality. The clauses are segmented into semantically atomic units but at least complete sentences. These atomic units are called clauses for the remainder. The legal comments are obtained in the EPUB/XHTML format that internally uses an XML derivative. Both contract clauses and legal comment chapters are pre-processed in the given order:

1. **XML tags removal:** The legal comments are obtained in XML format and are enriched with meta information such as links to external resources. The raw text per chapter is extracted by stripping all XML tags. This step is not necessary for the contracts.

Doc.	Name	#Narrow tags	#Broad tags
1	Miet- und Prozessrecht	49	96
2	Vermieterlexikon	52	91
3	Vermieter-Praxishandbuch	26	36
Total		**127**	**223**

Table 5.6.: Legal comment chapters labelling statistics (Landthaler et al. (2018b)).

2. **Newline replacement:** Single or compound newline characters are replaced with a single blank whitespace character.
3. **Whitespace tokenization:** The raw full-text strings are split into tokens by splitting the strings at whitespace characters.
4. **"§"-sign replacement:** The paragraph sign plays an important role in German legal documents. It indicates internal and external references. In a later step, all one-character tokens will be removed. To retain the information, the paragraph sign is replaced with a unique token consisting of a larger amount of characters that is unlikely to be used otherwise (PARAGRAPHSYMBOL).
5. **Special character removal (except hyphen):** All punctuation such as periods, commas, semicolons, quotation marks and similar characters need to be removed. The easiest way to accomplish this is to retain only alphanumeric characters. Note that hyphens between alphanumeric characters are retained because many compound words contain hyphens in the German language.
6. **One- and two-character token removal:** Despite the previous pre-processing steps, the pre-processed corpus contains many tokens of one or two characters that result from text organization elements and legal field-specific elements in the raw text. Examples for such elements are hierarchical text segment identifiers and list element identifiers. Thus, all tokens with less than three characters are removed. However, with this simple heuristic, not all undesired elements are removed, for example, the "iii" list item identifier of "i", "ii", "iii" is retained.
7. **Lowercasing:** In the German language, sentences usually start with upper case characters (with very few exceptions). Nouns start most of the time with upper case characters, too. Other word types start with a lower-case character most of the time. To maximize the number of exemplary usages for tokens, all tokens are lowercased. It might be useful to retain uppercase forms of nouns. However, the differentiation of nouns and non-nouns at sentence beginnings is difficult.

The pre-processing steps are almost equal to the pre-processing steps in Section 4.5. An additional step in comparison to Section 4.5 is the XML tags removal that is necessary because the data was provided in XML format. This pre-processing is required for the training of word embeddings models and is also used for gensim's TF-IDF implementation. Elasticsearch is feed with the raw texts because Elasticsearch applies an internal pre-processing of the documents.

5.10. Quantitative Evaluation

The major goal of this section is to compare word embeddings algorithms and the different approaches to Semantic Text Matching and implicit query expansion. Further, pre-processing options and variants of the approaches such as training corpus extension for the training of word embeddings models are explored. The results of the quantitative evaluation serve as a baseline for the results of the user study. The results are further important for the sensitivity analysis presented in Chapter 6. The dataset on German tenancy law and the previously constructed evaluation sets are used. Precision/recall is used as the primary relevance measure. For all experiments, the standard pre-processing described in Section 5.9 is applied unless stated explicitly otherwise.

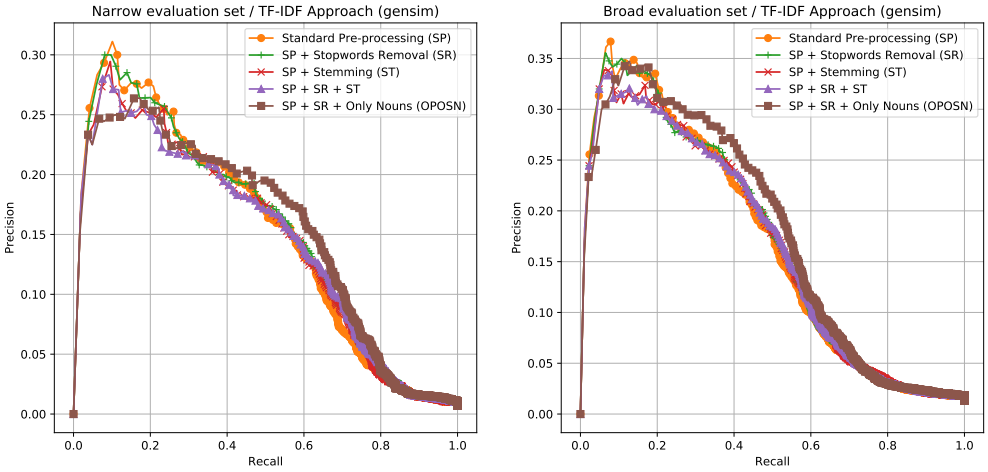

Figure 5.10.: Pre-processing options for the TF-IDF Approach using the gensim implementation.

First, different pre-processing options for the TF-IDF Approach are investigated. Most of the time, the success of pre-processing technologies depends on a specific corpus. For the TF-IDF Approach, gensim's TF-IDF implementation and Elasticsearch's *More like this* functionality are used. The gensim implementation allows for greater control over the pre-processing. For Elasticsearch, the internal pre-processing is used. Standard pre-processing technologies are stopwords removal and stemming. It is known that nouns often contribute most to topical search, see, for example, Landthaler et al. (2015). The impact can be measured by applying POS-tagging to the raw corpus strings and to retain only tokens classified as noun POS-tags. Here, spacy POS-tagger for the German language is used[3]. Figure 5.10 shows the precision/recall curves for the narrow and broad evaluation sets when the different pre-processing options are applied. The different pre-processing technologies lead to different results, but in general, the impact is small. The Standard pre-processing (SP) (no Stopwords removal (SR) and Stemming (ST)), leads to the best results in the top ranks as can be seen in the most left part of the diagrams. In the case that all other word types other than nouns are removed, the results are still comparable to the results where all word types are included. The results are slightly best when only nouns only POS-nouns (OPOSN) are kept on the right parts of the diagrams (i.e., not in the top ranks). One can conclude that nouns indeed contribute a lot to the topical search at hand, but other word types do not disturb the TF-IDF Approach.

The results of Chapter 4 suggest that the word2vec algorithm is best suited to find synonymous words that are not necessarily syntactically similar. The word2vec algorithm is chosen as the primary word embeddings algorithm for this chapter due to this property. For the word embeddings algorithms, again, all hyper-parameters could be investigated. The results of Chapter 4 show that the word2vec algorithm is comparably robust to changes to hyper-parameters. Hence, most hyper-parameters are set to the default values. The Vector Size hyper-parameter is set to 300 and the Iterations hyper-parameter is set to a value of 100. The Iterations hyper-parameter is set to a very high value because of the tiny training corpus. For a word-to-word similarity calculation, the cosine similarity is unaffected by the

[3]The raw text is the input to the POS-tagger. The following Penn treebank POS-tags are included: NN, NNS, NNP, NNPS https://spacy.io/api/annotation#pos-tagging, last accessed July 20, 2017.

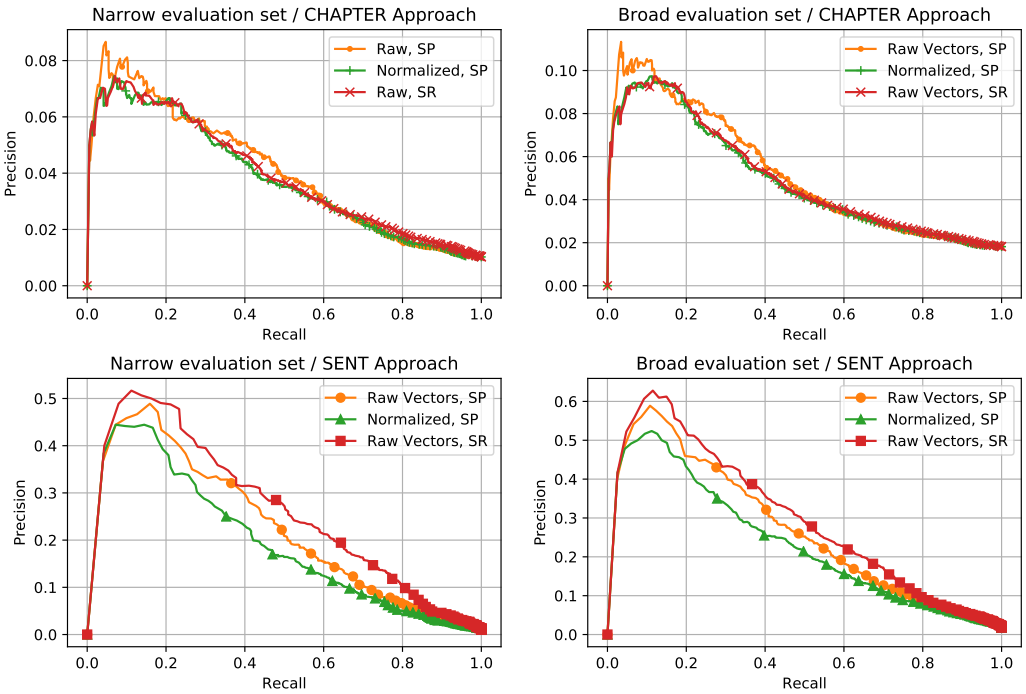

Figure 5.11.: Pre-processing options for word embeddings based approaches.

norm, i.e., the length, of word embeddings vectors. However, the accumulation of word embeddings vectors is affected by the norm of individual word embeddings vectors. Thus, word embeddings vectors can be left unchanged or normalized to unit length before the accumulation of word embeddings vectors. Like for the TF-IDF Approach, stopwords can be excluded before the accumulation of word embeddings vectors.

Figure 5.11 displays the results of stopwords removal and word embeddings vectors normalization for the CHAPTER and SENT Approaches. The hyper-parameter configuration W19, see Appendix B, for the word2vec algorithm is used to calculate the word embeddings model, cf. Appendix B. The SENT Approach is more susceptible to changes in the pre-processing steps. This can be rationalized by the much smaller number of tokens that are considered in the SENT Approach to represent sentences. An interesting aspect is that for the SENT Approach, stopwords removal has a positive effect, while for the CHAPTER Approach, removing stopwords leads to significantly worse results. Again, the different text segment sizes can explain the results. In general, the SENT Approach performs significantly better than the CHAPTER Approach. Note the different scales on the precision axis.

Figure 5.12 shows the word embeddings vectors' norms depending on the occurrence frequency of the respective token in the training corpus. For image file size reasons, the terms are sub-sampled below the occurrence frequency of 900 by a factor of 1:10. However, the qualitative results of the diagram are preserved. The red rectangles show the initial word embeddings vector norms before a word embeddings training algorithm is started. The norms vary a lot. There is a tendency that more frequently occurring tokens result in larger norms of the respective word embeddings vectors. The effects of down-sampling frequently occurring terms are visible on the right side of the diagrams. The

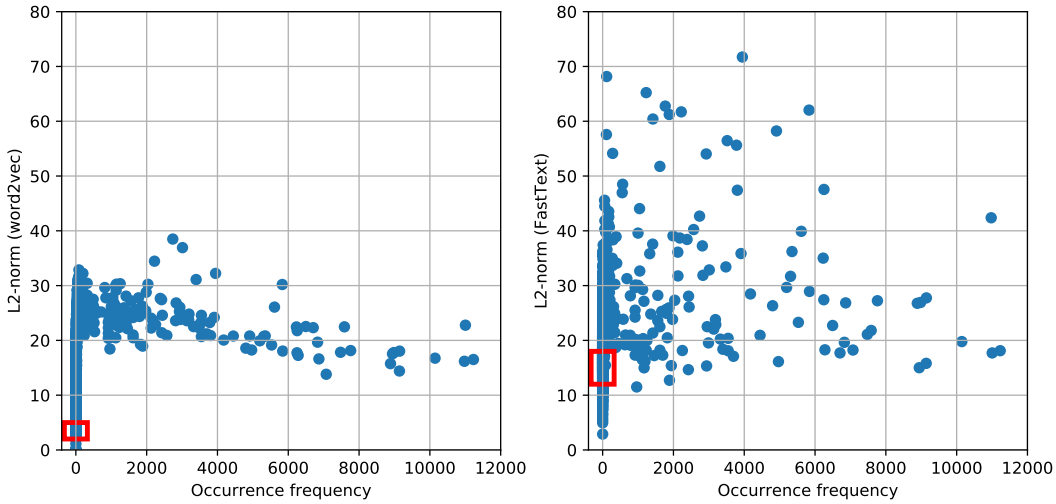

Figure 5.12.: Word embeddings vector norms.

most frequently occurring terms tend to have smaller norms than word embedding vectors of less frequently but still frequently occurring tokens.

Figure 5.11 shows that the word embeddings vector normalization does not lead to significantly better results for both approaches. The effects of stopwords removal and word embeddings vector normalization are almost neglectable.

Next, the word2vec and FastText word embeddings algorithms and the impact of the Min-Count hyper-parameter are explored. Figure 5.13 shows a comparison for both the CHAPTER and the SENT Approach. The results in Chapter 4 suggest that the Min-Count hyper-parameter (i.e., the minimum occurrence frequency of different training samples) is crucial for the quality of word embeddings vectors for individual tokens. In contrast to that, the WE-DF vector space model appears to benefit also from less frequently occurring training samples per token because setting the Min-Count hyper-parameter to a value of one leads to better results than setting the Min-Count hyper-parameter to a value of five. In the previous chapter, setting the Min-Count hyper-parameter to one leads to unstable word embeddings vectors for terms that occur less than five times in the training corpus. The training corpus is small. Hence, the out-of-vocabulary problem can become more severe and unstable word embeddings vectors are better included than removed. In all displayed cases, the FastText algorithm performs better than the word2vec algorithm. For the CHAPTER Approach, the difference is larger than for the SENT Approach. The results reported in Landthaler et al. (2018b) go in line with results displayed here. However, the results of Landthaler et al. (2018b) also show that for another evaluation set on the same corpus, the word2vec algorithm performs better than the FastText algorithm. Thus, the results of the comparison of word embeddings algorithms appear not to generalize well and depend on the evaluation sets. For the remainder, the word2vec algorithm is favored over the FastText algorithm because of its capability to detect synonyms that differ syntactically.

The training corpus that is used to train the word embeddings models is tiny. It is a common approach to reuse word embeddings models that have been pre-trained on large corpora. Figure 5.14 shows the

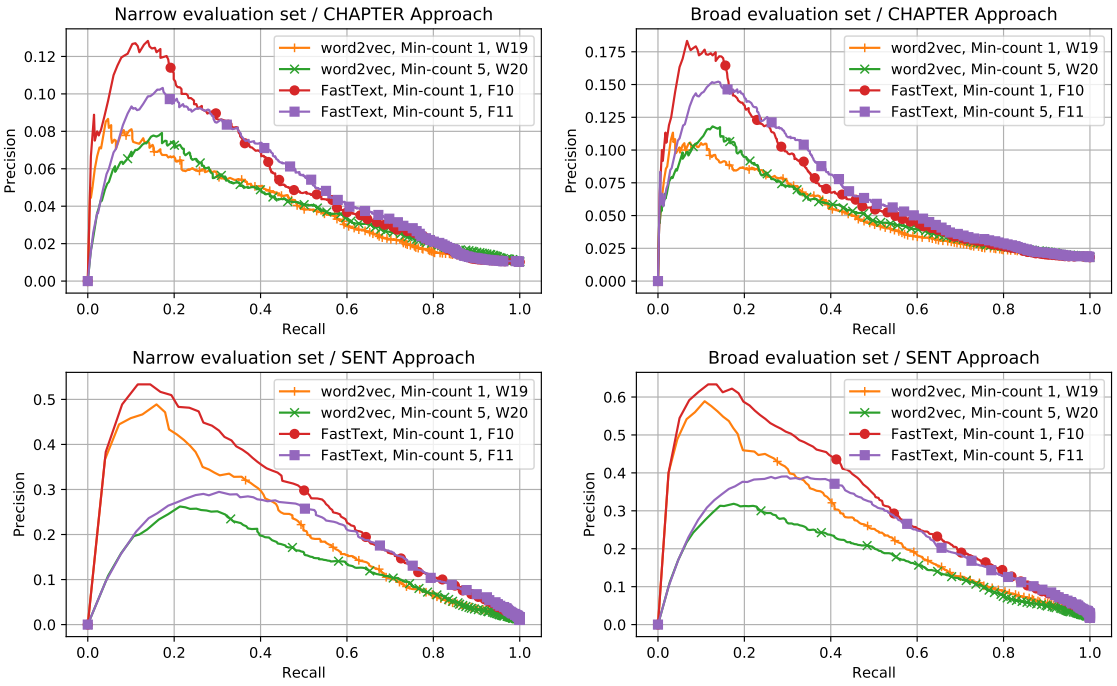

Figure 5.13.: Word embeddings algorithms and the Min-Count hyper-parameter.

results of different corpus extensions. First, the GCC[4] is added to the training corpus. Around one-fifth of the GCC is on the topic of tenancy law, but the GCC also covers other legal fields such as family law and inheritance law. The changes to the results are very small. The SENT Approach yields slightly better results but only in a small region of the precision/recall curves.

Next, a German Wikipedia[5] and the GCC are added to the training corpus. Wikipedia dumps are regularly used to compute pre-trained word embeddings models. However, the results suggest that just adding *some* more data can lead to significantly worse results. Note that for the German Wikipedia, GCC and standard training corpus again the Iterations hyper-parameter is set to 100. It could be argued that "adding more training data will result in diminishing results" (Mikolov et al. (2013a)) because more dimensions are required to encode the semantic information. However, the chosen value of 300 for the Vector Size hyper-parameter is commonly used to calculate pre-trained word embeddings models.

Figure 5.15 shows the precision/recall curves for the different approaches displayed in Figure 5.16 but split to the individual tags. The different performances of the different approaches carry on to the evaluation set subsets. The precision/recall curves for the different tags vary a lot, even for the same approach. A reason can be, that larger evaluation subsets in terms of legal comments per tag increase the chance to find relevant documents in the limited number of legal comment chapters (1801

[4]BGB (GCC), http://www.gesetze-im-internet.de/bgb/, last accessed November 24, 2015
[5]German Wikipedia dump ("dewiki-latest-pages-articles" of December 3, 2017) obtained via WikiExtractor from https://github.com/devmount/GermanWordEmbeddings, last accessed in December 2017

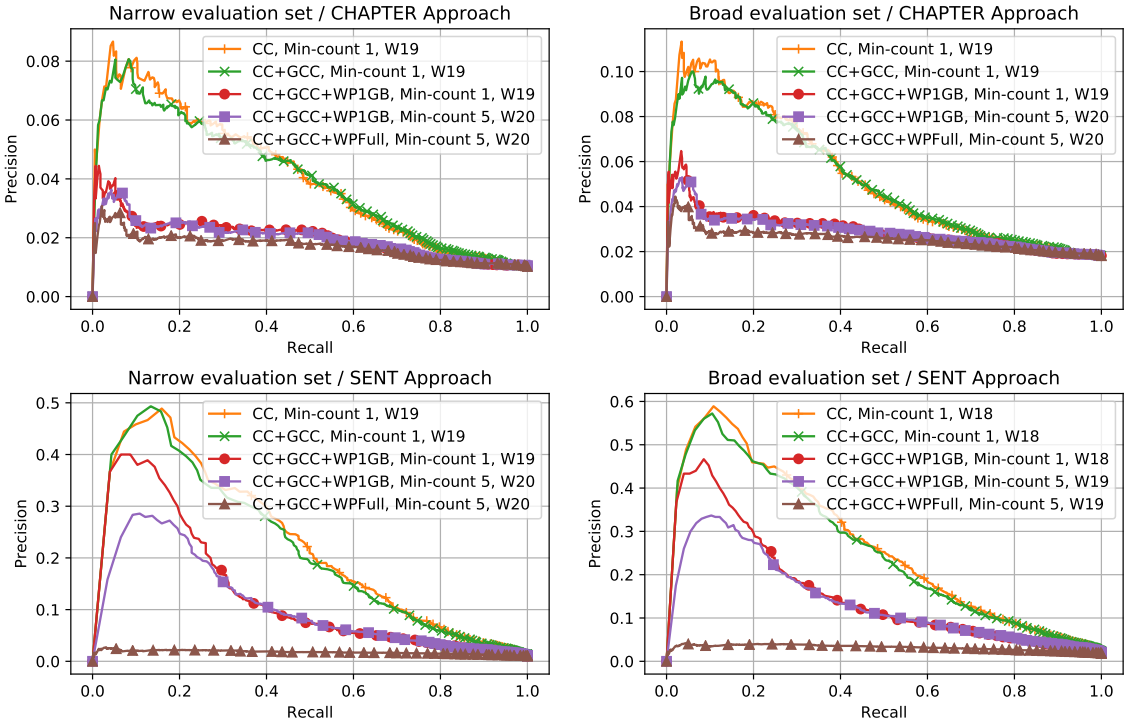

Figure 5.14.: Training corpus extension for the word2vec algorithm.

legal comment chapters). The differences in the performances per tag further underline the semantic complexity of the subject at hand.

Figure 5.16 provides a final comparison of the different approaches. The CHAPTER Approach performs worst, even worse than the TF-IDF Approach with the gensim implementation. An interesting aspect is that Elasticsearch's *More like this* functionality that is also based on the TF-IDF text representation performs significantly better than the gensim implementation of TF-IDF. This can be the case due to more advanced variants of TF-IDF and different pre-processing options. The SENT Approach performs best, especially in the top-ranked results of the narrow evaluation set. For the broad evaluation set, the SENT Approach still performs better than Elasticsearch's *More like this*, but less significantly.

In summary, according to the quantitative evaluation, the SENT Approach, where legal comment chapters are further segmented into sentences and sentence embeddings are compared against input queries, performs better than both TF-IDF implementations for the top-ranked results. The SENT Approach performs significantly better than the CHAPTER Approach. One can conclude from this result that WE-DF text representations are better suited to encode smaller text segments. In general, the results of the two evaluation sets are very similar. This can be seen as consistency among the two evaluation set. It can also be seen as a somewhat expected result because the broad evaluation set includes topically more distant relations than the narrow evaluation set. The narrow evaluation set is more strict than the broad evaluation set in a topical sense. Thus, it can be expected that fewer relations of the broad evaluation set appear in the top ranks. For the user study, the CHAPTER and

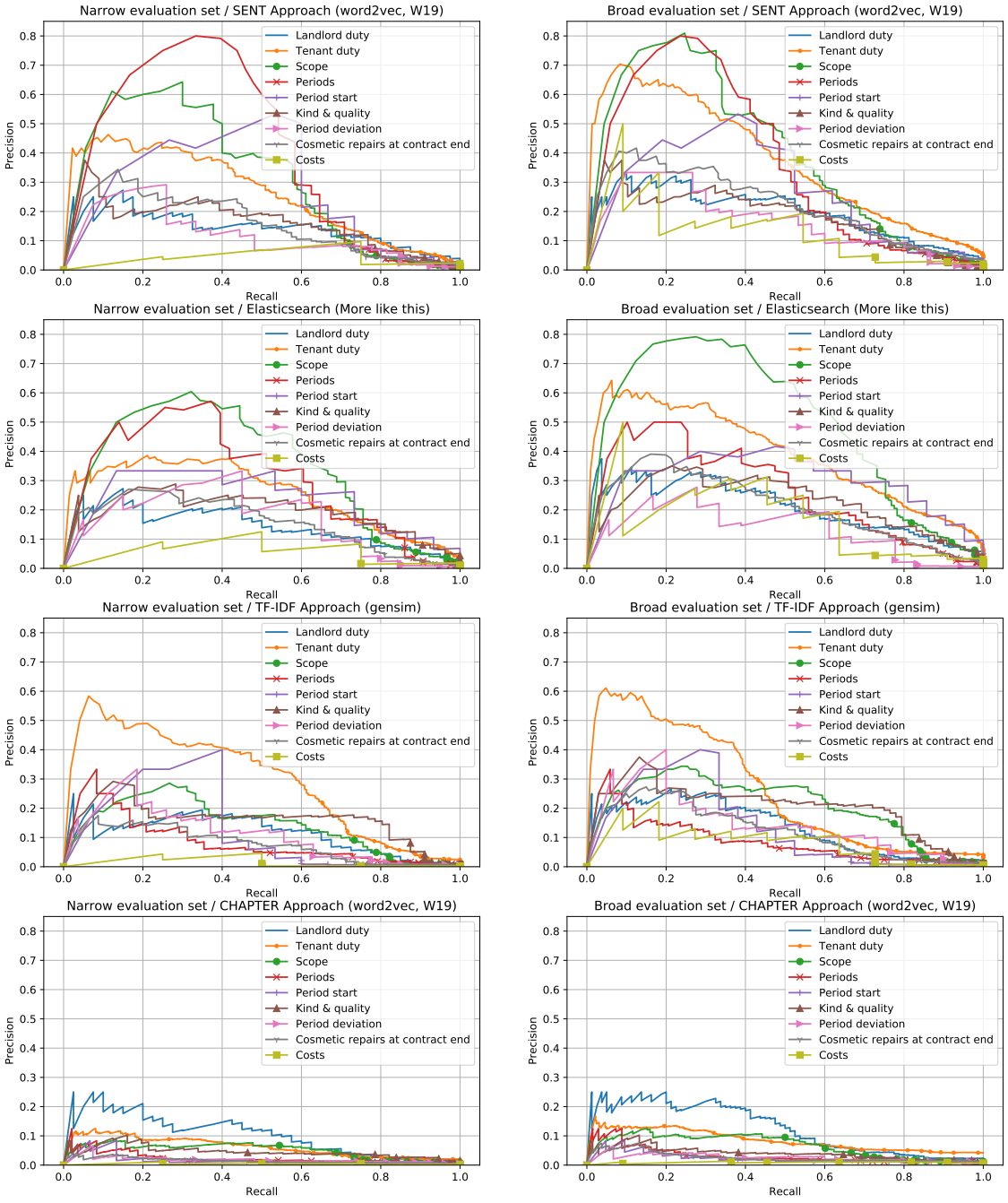

Figure 5.15.: Comparison of approaches for individual tags.

SENT Approaches with the word2vec algorithm and the TF-IDF Approach with Elasticsearch's *More like this* functionality are selected.

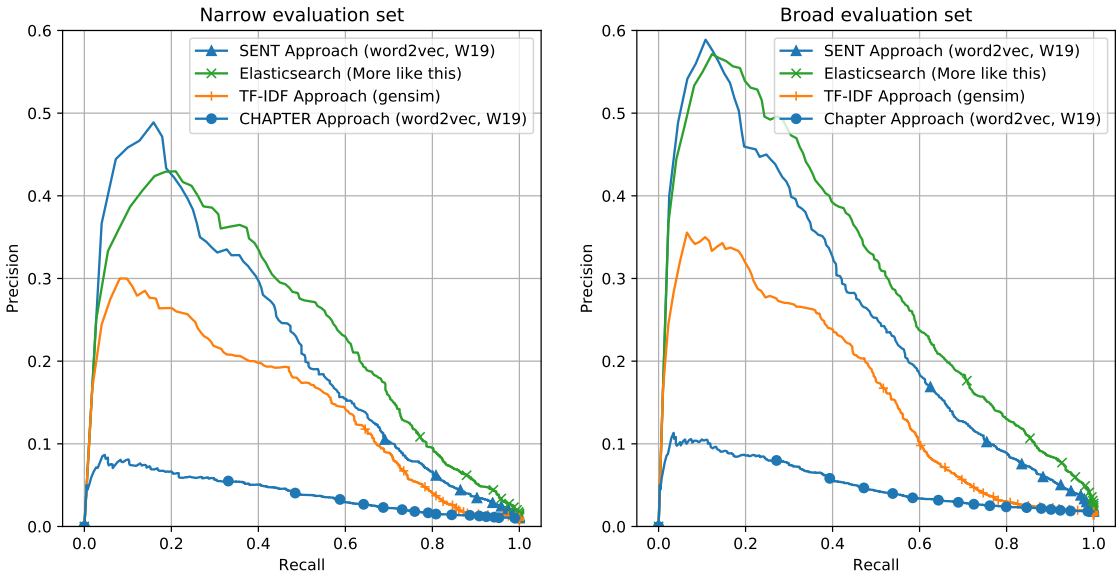

Figure 5.16.: Comparison of approaches.

5.11. Implemented System

A system has been implemented to assess the practical applicability of the use case and the suitability of technical solutions. The system has a client-server architecture. Two frontend applications and one backend application have been implemented. The frontend applications share the backend application. One frontend is a web application and the other frontend is a Microsoft Word AddIn. Figures 5.17 and 5.18 present screenshots of the implemented frontend applications. The screenshot of the Microsoft Word AddIn shows an excerpt of a German tenancy contract template on the left side and in the AddIn's results for the selected text "Die Schönheitsreparaturen müssen fachgerecht ausgeführt werden." ("The operations need to be carried out technically correct.") on the right side.

The web frontend is primarily for testing and demonstration purposes. It is implemented using the React Javascript framework. Microsoft Word is a widely used text-processing tool in the (German) legal domain. The Microsoft Word AddIn allows for tighter integration into lawyers daily workflows. It is implemented in Javascript and uses the Microsoft Word API for AddIns. The Microsoft Word AddIn has been implemented by a working student of the industry partner Haufe Group.

The user study was carried out with the Microsoft Word AddIn. The Microsoft Word AddIn has been prepared for the user study in several aspects. The Microsoft Word AddIn has been made publicly available in the Microsoft Application Store. The access to the Microsoft Word AddIn is controlled via customized access tokens. Both search methods, Keyword search and Selection Search, are implemented in the Microsoft Word AddIn. The Keyword Search in the Microsoft Word AddIn uses Elasticsearch's *query* functionality. The users can switch between three search technologies for the Selection Search. The search technologies were anonymized for the user study. The three search technologies are the CHAPTER, SENT and TF-IDF (MLT) Approaches.

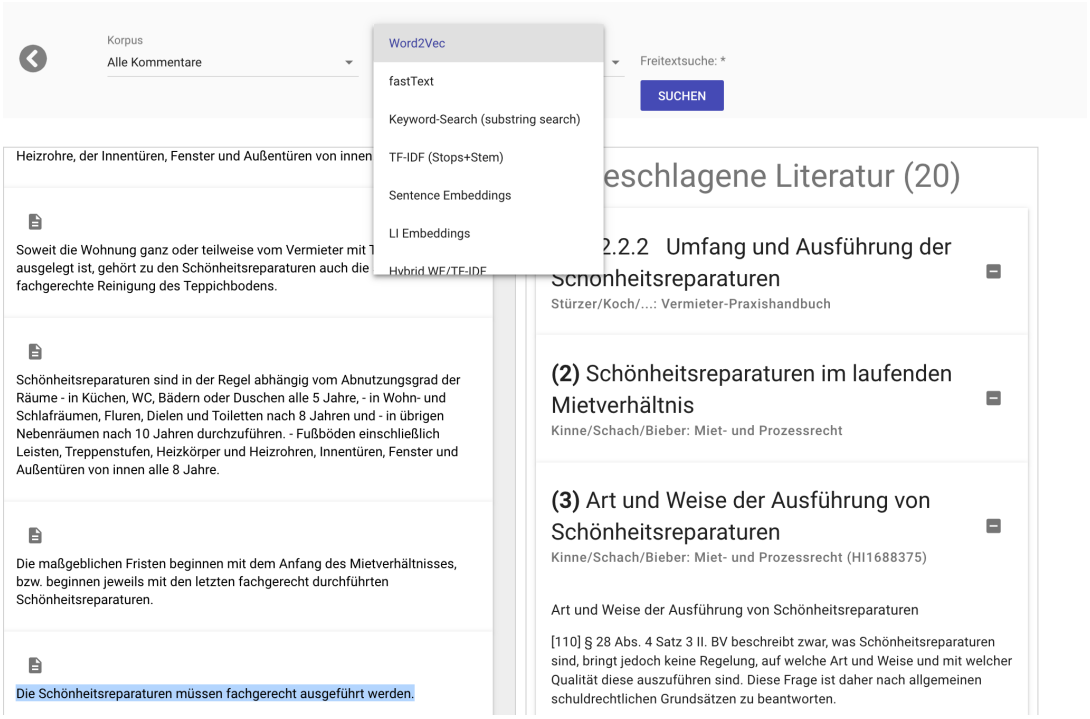

Figure 5.17.: Screenshot of the web application.

In the text-field labeled with "Suche nach" ("Search for"), the participants can modify search queries. The Microsoft Word AddIn allows users to explore the full legal comment chapter in search results by clicking on a button labeled with "Zeige Text" ("Show text"). The raw, i.e., not pre-processed, legal comment chapters are displayed to the users. For the Keyword Search, the users can enter and modify a keyword list in a text field. For the Selection Search, queries can be created by selecting text segments in the document opened in Microsoft Word. The user could further enter and modify a query in the text-field for the Selection Search, too. However, this option was not explicitly communicated to the participants.

The backend provides the search functionality. The backend is implemented as a Flask (see Appendix A) webserver in Python. The search functionalities are accessible via a REST API. The word embeddings based approaches use the nmslib (see Appendix A) that implements state-of-the-art approximate k-nearest neighbor indexing algorithms to ensure efficient retrieval. The k-nearest neighbor algorithm uses buckets to index document vectors and, therefore, the maximum result list that can be retrieved efficiently is limited. However, this does not conflict with the user study because it is sufficient to display the top 20 results to the users.

5.12. User Study

A user study was conducted to assess the practical relevance and the quality of the implemented solution in terms of usability. Parts of the user study have been previously reported in Landthaler et al.

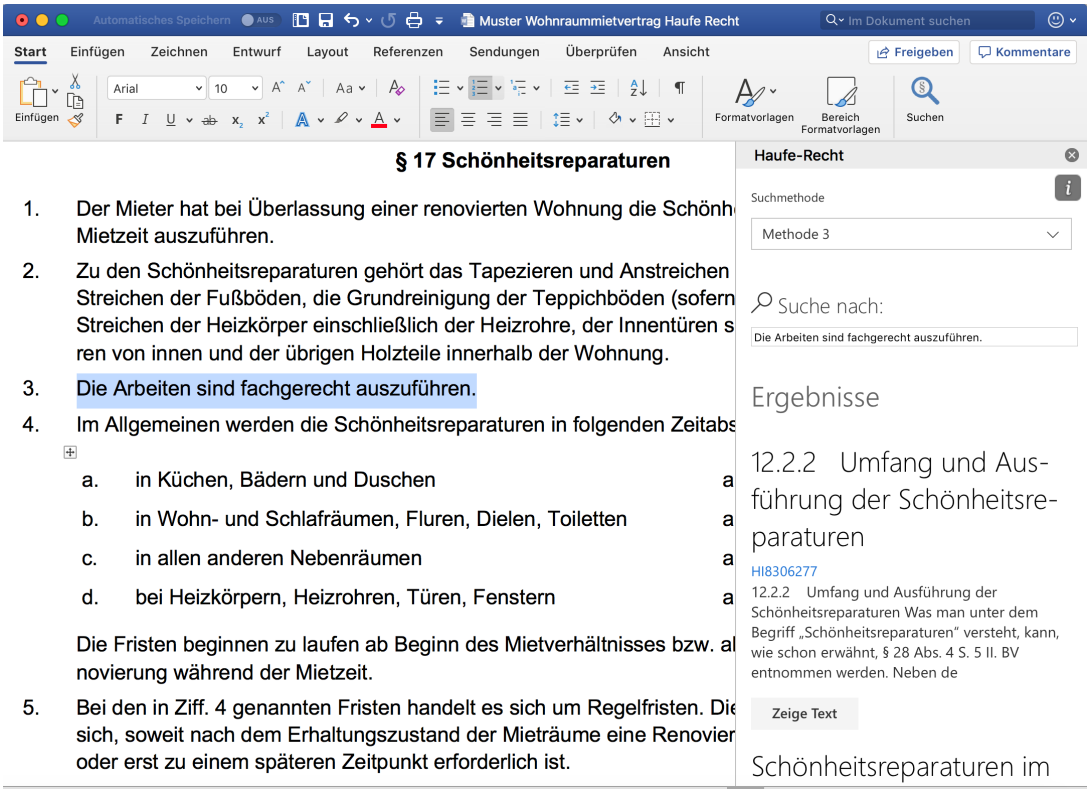

Figure 5.18.: Screenshot of the Microsoft Word AddIn (Landthaler et al. (2019)).

(2019). In comparison to Landthaler et al. (2019), ten additional participants could be acquired, resulting in a total of 25 participants. The study is not representative for the population of all lawyers in Germany[6]. However, according to Purchase (2012), p. 78, and Hinton (1995), "fifteen participants are required to ensure a statistical power of 0.8". In contrast to Landthaler et al. (2019), the results should be even more statistically significant. The results of the user study with 15 and 25 participants only differ slightly. In addition to the results presented in Landthaler et al. (2019), an analysis of the SUS-subscales and a more detailed analysis of the free-text comments are presented here.

Several diagrams that visualize the results of the user study use boxplots. Boxplots are particularly useful to summarize the results of ordinal data. Boxplots display five descriptive measures. Figure 5.19

[6]https://www.brak.de/statistiken, last accessed August 2, 2019

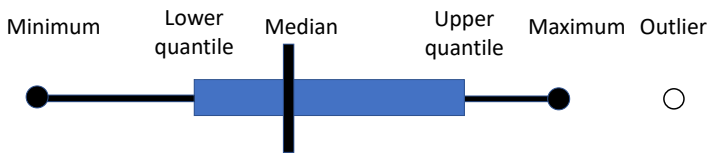

Figure 5.19.: Sample boxplot.

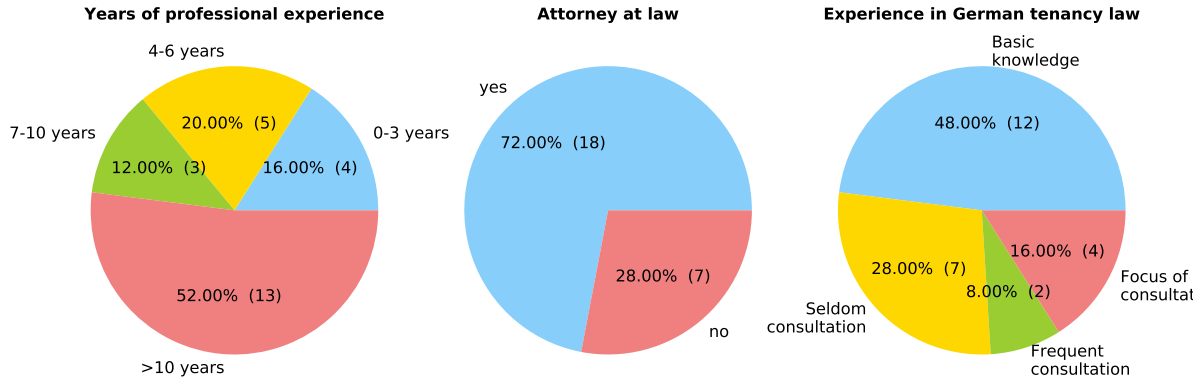

Figure 5.20.: User study participants characteristics.

illustrates the descriptive measures described by boxplots. The questions of the online questionnaire and corresponding answer options that are used in this section are listed in Appendix C.

5.12.1. User Study Participants

Figure 5.20 (Questions **Q01**-**Q03**, see Appendix C) gives an overview of basic information about the 25 participants[7]. All participants completed law studies. The majority of the participants are senior lawyers with more than ten years of professional experience. In contrast to Landthaler et al. (2019), the percentage of "senior" lawyers is slightly increased by around 6%. Around three-quarters of the participants are attorneys at law, i.e., licensed attorneys. The experience in the legal field of tenancy law is threefold: One-quarter of the participants stated that tenancy law is the focus of their consultation or that they conduct frequent consultation and can be considered experts in German tenancy law. Another quarter seldom conducts consultation and half of the participants attributed themselves only basic knowledge on German tenancy law. In contrast to Landthaler et al. (2019), the percentage of experts in tenancy law is decreased by around 9%.

Figure 5.21 shows that the diversity of the lawyers' areas of focus is broad. The largest fractions of the participating lawyers are engaged in tenancy law, labor law, business law, IT law, (6 mentions) each, followed by tax law (5 mentions) and IP law (4 mentions) for up to three possible selections from a given set of 21 legal fields. All other legal fields are mentioned at most three times. On average, each participant selected 2.74 legal fields, i.e., some participants selected less than three legal fields as fields of expertise.

[7]Details on participant acquisition: In the first step, an invitation email was sent to 6.600 lawyers in the customer database of the industry partner Haufe Group. Sixty-five lawyers responded that they want to participate. In the second step, an email was sent to the respondents that provided with a customized link to an online questionnaire, a sample tenancy contract template, cf. Harner (2017), and detailed instructions on the installation of the Microsoft Word AddIn for different Microsoft Word versions. Twenty-two lawyers started the questionnaire and 16 lawyers completed the questionnaire. Due to a missing question (in one of the SUS question batteries), eight records had to be removed from the evaluation. Because of the small number of records, additional lawyers were invited. Employees of the industry partner and personal contacts were asked to participate. For the results presented in Landthaler et al. (2019), five employees and two personal contacts additionally completed the questionnaire. For the results presented in this thesis, additionally, two customers, one employee and seven personal contacts completed the questionnaire.

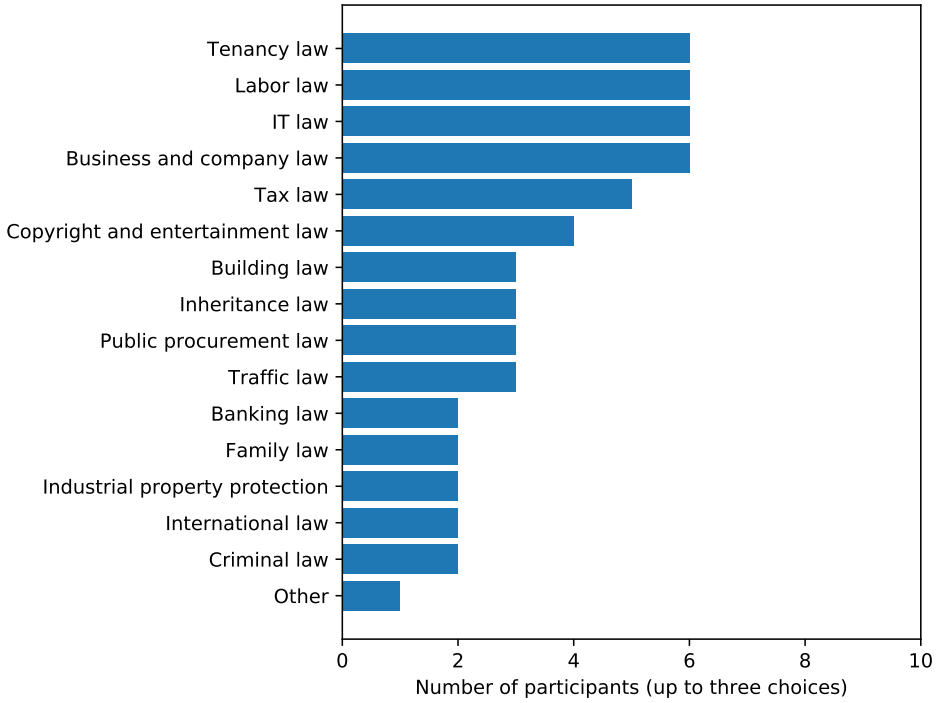

Figure 5.21.: Participant's focus on legal fields.

5.12.2. User Study Layout

One of the main goals of the user study is to assess the usability of the two search methods. The System Usability Score had been chosen, see Section 3.7.4. To accustom the participants with the study subject, the Microsoft Word AddIn and it's functionality, the participants were given a task on tenancy law. The task consists of a carefully crafted question. The participants had to decide whether a landlord can claim costs for a professional carrying out a cosmetic repair if the tenant did not carry out the cosmetic repair in a technically correct manner. The participants were given detailed instructions on how to use the Selection Search. The participants were navigated in the task description to a specific sentence in the sample contract and asked to select the sentence as input to the Selection Search. The system was prepared to cater for light variations and errors in the selected sentence. In this way, the output of the system was controlled for the experiment. The participants were asked to evaluate if the returned answers by the system contain relevant information. The detailed task description and the task served as a tutorial on using the Microsoft Word AddIn and the Selection Search. After the completion of the task, the participants were asked to answer the first SUS question battery. At this point, the participants were already familiar with the question and parts of the corpus. Next, the participants were asked to use the Keyword Search and to compare the experience with the Selection Search. The participants had to decide on their own what keywords they want to use to find relevant

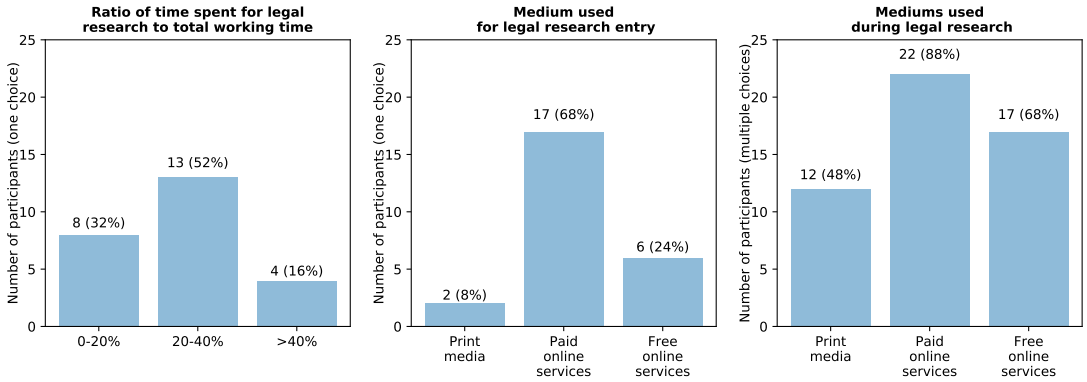

Figure 5.22.: Legal research behavior of user study participants.

results. Afterward, the participants were presented the SUS question battery a second time to assess the Keyword Search.

Another major goal of the user study is to evaluate the different search technologies that deliver results. The participants were asked to freely use the different (anonymized) search technologies serving the Selection Search and to freely use the Keyword Search. The participants could either use the provided sample contract template or a tenancy contract of their choice (and to pay attention not to use privacy law relevant content). The free usage included the free choice of text selection for the Selection Search. After the free usage phase, the participants were asked to judge the different search methods and search technologies with school grades. Moreover, the participants were asked to provide textual feedback. Finally, the participants were asked if they want to use Microsoft Word AddIn in the future.

The design of a user study is subject to many possible variants but also constraints. A guide to constructing experiments for human-computer interaction methods is provided, for example, by Purchase (2012). However, as lawyers time is expensive, it is difficult to acquire participants. Thus, while randomization techniques are powerful, the expected number of participants did not allow for greater variations, for example, the construction of several tasks. Moreover, several tasks lead to less comparability of the results. The free part of the user study is in contrast to a controlled experiment less scientific because the results are much less comparable and exposed to more influential factors that cannot be controlled. However, the superior goal is to assess the practical applicability of the Word AddIn, the search methods and search technologies.

5.12.3. User Study Results

The participants were asked questions about their legal research behavior, see Figure 5.22 (**Q05**-**Q07**) for a visual presentation of the results. The majority of the participants estimated to spend a fraction of 20 to 40% of their working time for legal research. One can conclude that legal research is an essential part of lawyers daily work. An interesting result is the high fraction of electronic resources used for legal research. 17 (68%) participants stated to start a legal research process with paid online services. In contrast to the legal research entry, in later stages of the legal research process, all resources, paid and free online services, and roughly a half of the participants uses printed documents.

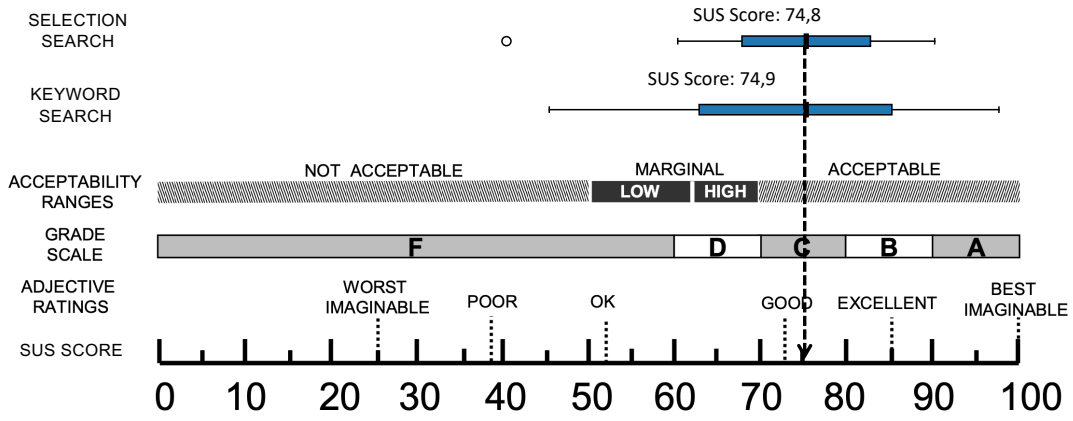

Figure 5.23.: SUS scores for Selection Search and Keyword Search, adapted from Bangor et al. (2009).

Figure 5.23 (**Q08**-**Q17** and **Q19**-**Q28**) presents the results for the SUS evaluation of the two search methods Selection Search and Keyword Search. The SUS score of a single SUS question battery is a value between zero and 100. The SUS scores are averaged over multiple participants. Bangor et al. (2009) proposes several interpretation schemes to interpret SUS scores that are visualized in Figure 5.23: "Acceptability ranges", "grade ranges" and "adjective ranges". According to the first two interpretation schemes, the two different search methods perform almost equally good. Both search methods achieve a solid "C" grade that can be interpreted as "good" or "in acceptable range". The eleventh question suggested by Bangor et al. (2009) (**Q18** and **Q29**) result on average in a value of 2.72 for the Selection Search and a value of 3.04 for the Keyword Search. The average values of the third interpretation scheme (the additional 11th question to the ten standard SUS questions) verify the measured SUS scores.

The boxplots used in Figure 5.23 allow for a judgment of the variety of the SUS scores. Overall, the SUS scores are in the upper ranges, so the majority of participants rate the search methods implemented in the Microsoft Word AddIn as good. However, the variance in the individual scores is high.

Figure 5.24 displays the aggregated counts for the ten individual SUS questions (SUS sub-scales). An inspection of the SUS sub-scales reveals that the two search methods are rated quite similarly by the participants. Mild differences can be seen on four sub-scales. In comparison to Keyword Search, more participants think that they would like to use Selection Search frequently. The participants think that the Keyword Search has fewer inconsistencies than the Selection Search. The Keyword Search is considered more comfortable to learn than the Selection Search. Last but not least, the participants feel slightly more confident in using the Keyword Search than the Selection Search.

Table 5.7 shows the SUS-Scores for selected subgroups of the participants. For the Keyword Search, the results are similar among all subgroups (except for subgroups with very few participants). The SUS Scores for the Selection Search vary more. A surprising result is that the participants whose self-perception of experience in tenancy law is low found the usability of Selection Search significantly less good than the more experienced participants. Another interesting aspect is that participants with either very few or very much professional experience rated the usability of Selection Search significantly less good than participants with four to ten years of professional experience.

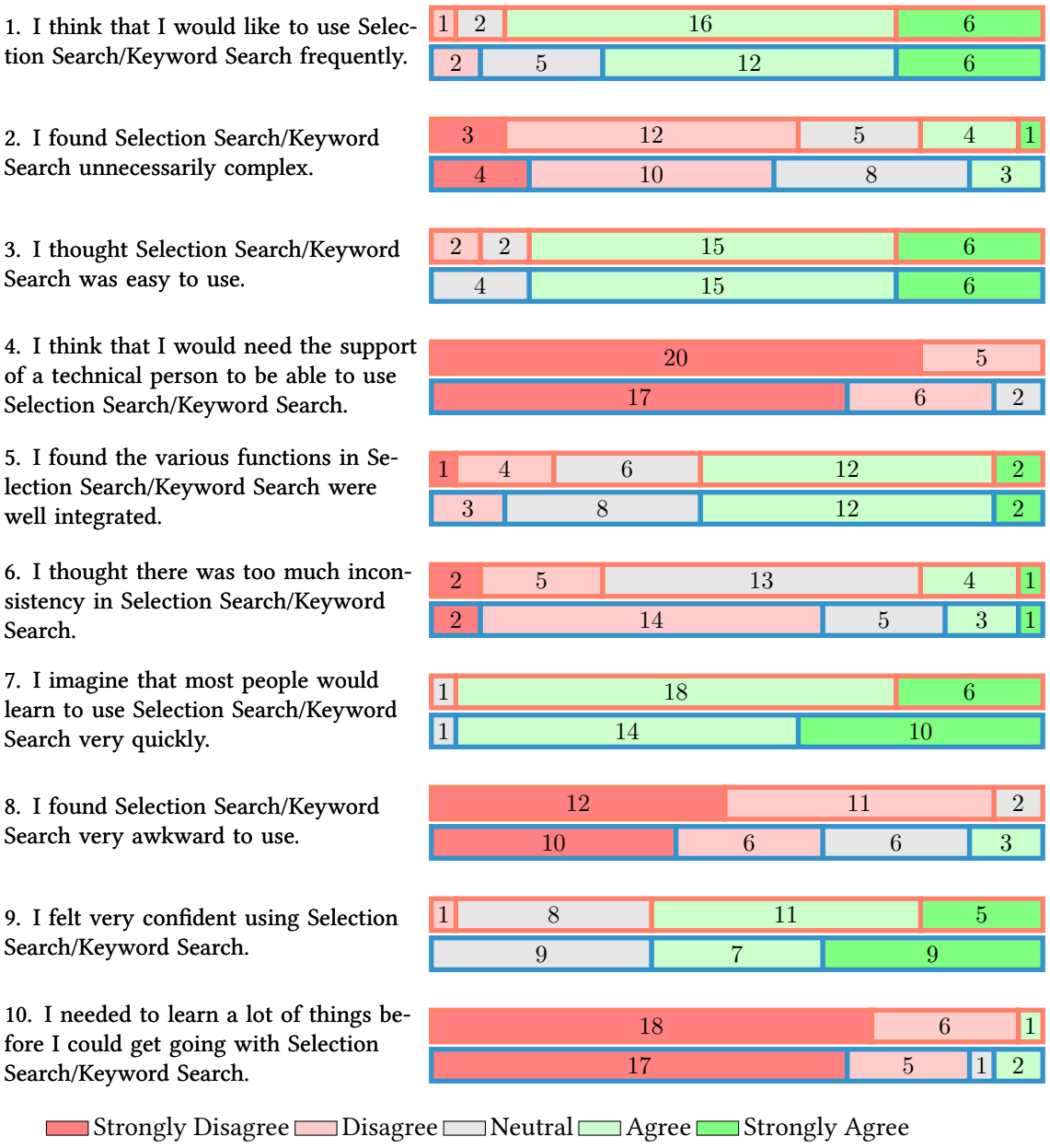

Figure 5.24.: SUS-subscale analysis. The bars display the number of opinions per answer option. For each SUS question, the upper bar displays the result of Selection Search (red frames) and the lower bar displays the result of Keyword Search (blue frames).

	Subset	Number of participants	Selection Search SUS score	Selection Search SUS score
	Attorney at law: yes	18	75.55	75.41
	Attorney at law: no	7	72.85	73.57
Experience in German tenancy law	**basic**	12	71.04	73.95
	rare consultation	7	76.42	73.57
	frequent consultation	2	77.50	86.25
	focus of consultation	4	81.85	74.37
	basic to rare consultation	19	73.02	73.81
	frequent to focus of consultation	6	80.41	78.33
	rare to focus of consultation	13	78.26	75.76
Years of professional experience	**0-3 years**	4	70.62	66.87
	4-7 years	5	80.50	82.00
	8-10 years	3	78.33	76.66
	>10 years	13	73.06	74.23
	0-7 years	9	76.11	75.27
	>8 years	16	74.06	74.68
	0-10 years	12	76.66	75.62

Table 5.7.: Correlation analysis of SUS-scores.

The participants were asked for the exclusive, most important reason to favor Selection Search or Keyword Search. For 19 participants, the quality of the results is most important. One participant chose the usability of the tool, two participants voted for the control over the search functionality and three participants chose the option "other reason", cf. Figure 5.26 (**Q35**). This result indicates that the quality of the search results is essential.

The judgment of the search methods and the different Selection Search's search technologies after the free usage by the participants did not lead to clear results, as can be derived from the results depicted in Figure 5.25 (**Q30**-**Q34**). The participants were asked to rate the different (anonymized) search technologies for the Selection Search and the Keyword Search with school grades. The results of the quantitative evaluation are confirmed in parts by the participants' judgment of the different search technologies.

As indicated by the boxplots displayed in Figure 5.25, the participants slightly prefer the SENT Approach over the CHAPTER and the TF-IDF (MLT) Approach.

Furthermore, the participants were asked to vote exclusively for the Keyword Search or one search technology for the Selection Search. The results depicted in Figure 5.25 (bottom, right subplot) show that the participant's opinion is polarized. It is split into around half for the Selection Search with

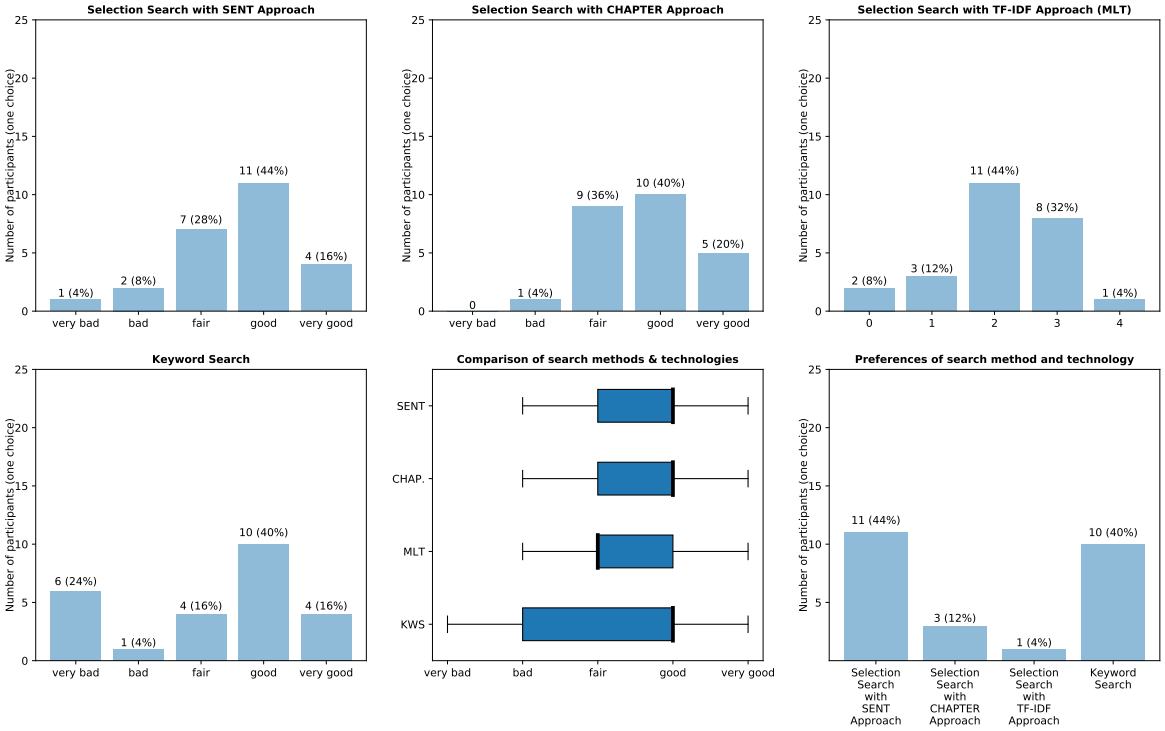

Figure 5.25.: Participant's rating of search methods and search technologies.

SENT search technology and Keyword Search. Very few participants voted for the other two search technologies for the Selection Search.

The very similar results of the comparison of the search methods and Selection Search search technologies is also reflected by the free-text comments provided by the participants. After using and judging all search methods and Selection Search search technologies, the participants had the opportunity to give free-text comments on the quality of the search results. Table 5.8 (**Q36-Q40**) summarizes the number of selected re-occurring mentions in the free-text comments by the participants when asked for the quality of the search results. The participants were further asked if any patterns in the search results are perceived, for example, whether longer or shorter selected text segments lead to better results. Two participants noted that Selection Search is depended on the existing text and complained that the contract text is semantically too broad to serve as input to a search query. In general, it is not possible to detect patterns in the quality of results depending on the length of the selected text segment. Two participants noted that selecting relevant text segments leads to good results. Another Two participants noted that Keyword Search requires skills to craft good query terms. One can conclude from that that the Selection Search requires training and experience, too.

During the compilation of the user study, it was not anticipated that the participants are able to use the Keyword Search in the same fashion as the Selection Search, i.e., to select individual words or text segments as input to the Keyword Search. In this case, the participants did not compose the keywords manually. Hence, the composition of search queries is significantly less flexible in this case. It was not possible to identify the actual number of participants that used the Keyword Search in this way.

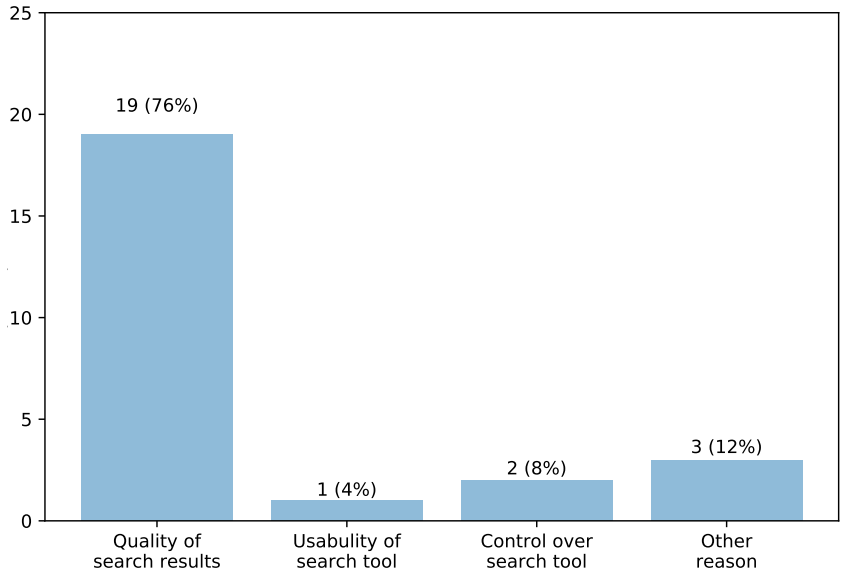

Figure 5.26.: Participant's relevancy estimation of legal information retrieval tools' features.

The free-text comments of the participant included several possible improvements. The participants would like to conduct a keyword search within the search results. Matching text segments could be highlighted. A save functionality for good search results is proposed, too. Several participants would like to get an explanation for Selection Search results. The majority of participants favored Selection Search with the SENT Approach over the CHAPTER Approach. Another significant subgroup did not identify differences in the quality of the results among the approaches for Selection Search (including MLT). The majority of participants did notice differences in the quality of results between Keyword Search and Selection Search. Many prefer Keyword Search over Selection Search because it is more transparent and many participants said that they like to craft query terms for Keyword Search. Several participants said that the functionality of the Keyword Search is very basic. The Keyword Search was limited in functionality compared to search technologies used at legal publishers. For example, no query expansion using a thesaurus or spelling correction was available.

There are several threats to the validity of the results. For example, some participants (actual number not accurately identifiable) used the Keyword Search by selecting text segments in the contract (which was neither intended nor anticipated in advance). The implementation of the Keyword Search was very basic. It did not include search query expansion, sub-string search or spelling correction. The participants were shown only one working example of the Selection Search before showing the SUS questions for the Selection Search. The order of presenting the Selection Search and the Keyword Search and the order of the search technologies for Selection Search could have been randomized to reduce potential side-effects (given a reasonably large enough number of participants).

Multiple free-text comments give rise to a need for better control and transparency of the search functionality. However, 23 participants (92%) of the participants said that they would like to use the Selection Search in the future (**Q41**).

In summary, legal research is an essential task of lawyers daily workflows. Digital resources are used

	Selection Search with SENT Approach	**Selection Search with CHAPTER Approach**	**Selection Search with TF-IDF (MLT) Approach**	**Keyword Search**
Selecting shorter text passages leads to better results.	4	0	0	1
Selecting longer text passages leads to better results.	2	2	1	1
No differences in results depending on length of text passages.	4	3	0	1
Correct and incorrect results are displayed.	3	0	3	0
Best results are obtained by selecting relevant text passages (irrelevant of length).	2	0	0	0
Skills to craft good query terms are required.	0	0	0	2
Differences in quality of results of the different methods are identified.	-	1	1	4
Differences in quality of results of the different methods are cannot be identified.	-	5	3	1

Table 5.8.: Free text comments analysis.

frequently, but also printed resources are still used frequently nowadays. The results of the quantitative evaluation are confirmed by the results of the user study to some degree. The SENT Approach is favored over the other search technologies, i.e, the CHAPTER and TF-IDF (MLT) Approaches. However, lawyers had a hard time to judge the quality of search results. The comparison of the search methods Keyword Search and the Selection Search leads to polarized results. Thus, the Selection Search can be seen as an alternative or complementary search method to traditional Keyword Search. The quality of search results is essential for legal information retrieval. The participants like the idea of using a Microsoft Word AddIn for legal research. However, for all technologies, a traceability or explanation of the composition of search results is very important. This explains why the Keyword Search is favored by around one half of the participants.

5.13. Discussion

In this chapter, the WE-DF text representation is used to represent text segments. The WE-DF text representation is considered as an alternative vector space model for (legal) information retrieval. Due to the semantic operations that can be carried out with linear vector operations in word embeddings model's vector space, the working hypothesis is that the WE-DF vector space model can be used to conduct a form of implicit query expansion. The hypothesis is tested with an innovative use case in German tenancy law. A user can select contract clauses or arbitrary text segments as input to a search query. The Selection Search in this form constitutes a natural language search. Moreover, the Selection Search is a different human-computer interaction method in comparison to the traditional Keyword Search. User intentions, domain-specific and technical use case challenges are identified.

The Semantic Text Matching problem is introduced as one type of problems that can leverage the WE-DF vector space model for implicit query expansion particularly well. Semantic Text Matching is an abstract problem of matching text segments (or documents) of one, two or several document types that are semantically and/or logically linked. Three different perspectives of the Semantic Text Matching problem are highlighted: The network view, the graph view and the information retrieval view. The text segmentation is not part of the Semantic Text Matching problem. Automating text segmentation is an important aspect of future research.

The German legal field of tenancy law is selected for the investigations. German legal comments contain condensed information from laws and court decisions. In the investigated use case, the Semantic Text Matching problem instance is to match contract clauses (or arbitrary-length text segments selected by users) and legal comment chapters.

The WE-DF vector space model is used as an alternative vector space model to the traditional TF-IDF vector space model. Several variations of WE-DF text representations are investigated in a quantitative evaluation. The TF-IDF vector space model serves as a baseline vector space model. Three different approaches are explored. The CHAPTER Approach accumulates word embeddings vectors for all tokens in a legal comment chapter or the input query, i.e., uses the WE-DF representation. The SENT Approach is a variant of the CHAPTER Approach. Large documents are segmented into smaller parts, for example, sentences. The TF-IDF Approach denotes using the traditional TF-IDF vector space model. However, different implementations that vary in details exist. In particular, the gensim TF-IDF implementation and Elasticsearch's *More like this* functionality are considered.

For the quantitative evaluation, two evaluation sets on cosmetic repairs in tenancy law are constructed manually. The construction of the evaluation set can be considered as a method to create evaluation sets for legal information retrieval tasks under certain conditions. Tags are used to label contract clauses and legal comment chapters. This reduces the labeling effort to a single scan over all documents of one type. The tags are identified as semantically atomic units that occur in at least two contracts. Thus, the identification of the tags still requires multiple scans over the contracts but not the legal comment chapters.

The CHAPTER Approach leads to the worst results in terms of precision/recall. The different TF-IDF implementations used (gensim and Elasticsearch) lead to different results. The Elasticsearch *More like this* TF-IDF implementation performs significantly better than the gensim implementation of TF-IDF. The gensim TF-IDF implementation leads to slightly better results than the CHAPTER Approach. The

SENT Approach shows significantly better results than the CHAPTER Approach and overall leads the to best results for the top-ranked results - even better than Elasticsearch's *More like this*. The impact of different pre-processing options to both vector space models is comparably small.

The out-of-vocabulary problem is relevant to the use case. Eventually, users select text that includes tokens that are not present in a word embeddings model. In contrast to the results of the previous chapter, an interesting aspect is that setting the Min-Count hyper-parameter of the word2vec algorithm to a value of one leads to better results than setting it to five. Thus, a lower quality word embeddings vector is better for the WE-DF vector space model than retaining only high-quality word embeddings vectors (at least for small training corpora).

The results of the quantitative evaluation suggest that the FastText algorithm leads to better results than the word2vec algorithm. The FastText algorithm's approach to train character n-gram embeddings could be used to address the out-of-vocabulary problem. However, this possibility was not further investigated and is left to future research. The word2vec algorithm is used due to its capability to detect synonymous words that syntactically differ for the user study and hints that the FastText algorithm's superiority is not valid in general.

The training corpus consists of six contracts and three legal comments. The training corpus for word embeddings algorithms is tiny in comparison to pre-trained word embeddings models that have been trained on billions of tokens. However, the word embeddings models trained on the small training corpus lead to the best results. This is in contrast to the general rule of thumb that more data leads to higher quality word embeddings models. The results indicate that the terms in the legal domain are susceptible to the training examples available to word embeddings algorithms and a training corpus enrichment is recommended to be carried out very carefully. This goes in line with results obtained by Erl (2018). More research on the effects of using different training corpora and mixtures of different training corpora is required but left for future research.

The results of the quantitative evaluation and manual inspection of the results together with the results of Landthaler et al. (2016) suggest that implicit query expansion is conducted by using the WE-DF vector space model. However, in terms of precision/recall measure, the WE-DF vector space model does lead to significantly better results than the traditional TF-IDF vector space model. The results indicate that WE-DF is suited to represent small text segments, but larger text segments can be efficiently represented as well by "simpler" TF-IDF vector space models.

The results are limited to the particular dataset, in particular, German tenancy law. The two created evaluation sets can only reflect the broad possible scope of a users search intentions to a small degree. The evaluation sets reflect only two comparably general granularities of user intentions. It remains a challenge for future work to evaluate individual user intentions better. However, more general evaluation sets still allow to detect tendencies of the effectivity of approaches. Due to the application of the principle of dual control, no inter-rater reliability can be calculated. Eventually, some insights are transferable to other legal fields, but general statements are hard to make. Due to a large number of hyper-parameters of word embeddings algorithms and the results of the previous chapter, the hyper-parameters are not investigated in depth. Due to the many word embeddings based approaches to query expansion and text segment representations, not all possible or conceivable approaches are investigated.

The practical relevance and applicability of the use case are assessed with a user study with 25 partic-

ipants. The majority of participants are senior lawyers. The study confirms related research that legal research is an essential part of a lawyer's work time. For the most part, lawyers start legal research with online services, both paid and free services rather than printed resources. However, printed resources are still used in legal research at later stages.

A Microsoft Word AddIn and a web application are implemented to assess the different search methods and search technologies for Selection Search. The results of the user study suggest that a text processor AddIn can be a valuable tool for lawyers during the audition of contracts. For the most part, the participating lawyers would like to use the Word AddIn in the future. The results of the user study further suggest that the quality of the search results of a legal information retrieval system are most important to the participants. However, it is a very challenging task for the participants to define "good" search results and as a consequence to judge the results delivered by the different search technologies (CHAPTER, SENT and TF-IDF Approaches).

When asked for a favorite search method, the participants form two equally sized groups. One-half of the participants are in favor of the Selection Search, while the other half favors the traditional Keyword Search. The input to the Selection Search depends on existing text to select (in the investigated form of Selection Search). Therefore, the study results suggest that the Selection Search could be useful as a complementary search method to Keyword Search. According to the study participants, an important aspect of the legal information retrieval system is the traceability of the employed technologies.

The results of the user study are limited due to the comparably small number of participants that is not representative of the population of German lawyers. Merely a small number of participants could be acquired because of the lengthy questionnaire, a lawyer's costly time and technical barriers such as the installation of additional software. The population size of German lawyers is around several hundred thousands and therefore 25 participants are not a representative sample. However, the results go in line with related research, cf. Section 3.1.

The use case demands technologies that ensure a deep understanding of the semantics of search queries. The investigated technologies and approaches are not powerful enough to cater to this need to a sufficient extent. However, recently introduced commercial applications of the use case demonstrate it's value. For example, Lexis Nexis introduced the SmartScan tool[8] that allows lawyers to select text segments and suggests literature from the Lexis Nexis database. A question that remains is why and how far the TF-IDF and WE-DF vector space models differ. This will be investigated in more detail in the next chapter.

[8] https://www.lexisnexis.at/produkte/lexis-smartscan-juristische-textanalyse-automatisierte-quellenrecherche/, last accessed July 20, 2019

Natural language is discrete. We shift to continuous representations of text, but they are not understood.

Marie-Francine Moens

CHAPTER 6

Sensitivity Analysis: eXplainable Semantic Text Matching

In the previous chapter, the WE-DF and TF-IDF vector space models are compared quantitatively. One result of the evaluation is that the CHAPTER Approach with the word2vec algorithm leads to worse results than the TF-IDF Approach. However, lawyers did not identify differences in the quality of the search results obtained by the two vector space models. The goal of this chapter is to explore the results more in-depth. Both approaches are based on vector space models. A legal comment chapter is represented by a single vector for both vector space models. From a technical point of view, both text representations are constructed as weighting schemes of corpus and document specific components. Thus, a deeper comparison of the two approaches can lead to deeper insights. So-called XAI approaches are a trend in AI that recently arrived at AI&Law research, too. A Sensitivity analysis has been applied to supervised classification tasks in general artificial intelligence but also in AI&Law. In this chapter, an XAI approach is presented that conducts a sensitivity analysis to (unsupervised) text similarity measures.

The investigations in this chapter are not specific to legal information retrieval but could be applied to any application of text similarity measures. However, the legal dataset on tenancy law of the previous chapter is re-used for the experiments. The sensitivity analysis can be applied to the individual words of a single text similarity calculation, for example, a single match of a Semantic Text Matching problem instance. The resulting significance scores for the tokens can be aggregated over several matches. Both variants, significance scores of single matches and aggregated significance scores are applied to compare the TF-IDF and WE-DF vector space model. Moreover, an analytical comparison of the weighting schemes of the two vector space models is presented. Parts of the research presented in this chapter have been previously published in Landthaler et al. (2018a). The research presented in this chapter is not complete and should be understood as an extended outlook.

6.1. Technical Approach

The main idea of the XSTM Approach is to subsequently remove tokens from a pair of text segments and to measure the change in the text similarity to assess the significance of a token to the text similarity. More formally, given two text segments t_1 and t_2 consisting of tokens $t_1 := \{w_1, w_2, ...w_N\}$ and $t_2 := \{x_1, x_2, ...x_M\}$, the process is as follows: Calculate the value of the text similarity measure of the two entire text segments $S = sim(t_1, t_2)$ where $sim()$ denotes the text similarity measure function. Remove a single token w_1 from one text segment: $t_1^* := \{w_2, ...w_N\}$ and calculate the text similarity measure among text segments t_1^* and t_2: $S_{w_1} := sim(t_1^*, t_2)$. The difference among the two text similarity values $\delta_{w_1} := |S - S_{w_1}|$ can be seen as an indicator of the significance of a token to the similarity of the two text segments. The process can be repeated for all tokens in t_1, t_2.

The XSTM Approach can be applied to any text similarity measure and any application where text similarity measures are used. Here, the XSTM Approach is applied to the Semantic Text Matching problem of matching clauses in contracts and legal comment chapters. This is the source of the name eXplainable Semantic Text Matching. The resulting relevance values per tokens can be aggregated or averaged among several pairs of text segments. The motivation to apply the XSTM Approach to this use case is to obtain a deeper understanding of how word embeddings algorithms work and to explore commonalities and differences of WE-DF and TF-IDF vector space models.

6.2. Empirical Evaluation

The WE-DF vector space model leads to slightly worse results than the TF-IDF vector space model in the quantitative evaluation of the previous chapter. Much fewer dimensions are used by the WE-DF text representation than by the TF-IDF text representation to represent text segments. On the one hand, it could be expected that accumulating vectors of much fewer dimensions leads to worse results for large text segments. On the other hand, both approaches are weighting schemes of corpus wide and document specific statistics. The goal of this section is to gain more insights into the behavior of the two approaches with empirical investigations. First, single matches are explored. Second, the XSTM Approach is applied to several matches. The single match case study has been previously presented in Landthaler et al. (2018a).

6.2.1. Case Study: Single Matches

The XSTM Approach has been integrated into the web application frontend described in Section 5.11 for the SENT Approach. In contrast to the CHAPTER approach, the SENT Approach segments legal comment chapters into sentences that are pooled and ranked for a user query, see Section 5.6.2. Figure 6.1 depicts a visualization of the results of applying the XSTM Approach to a single match of a contract clause and a sentence from a legal comment chapter. A radar chart is a visually pleasing way to present the results of an XSTM analysis. However, the downside of a radar chart is that it does not scale to a large number of tokens. Tokens of both query and matching sentence in the legal comment chapter are considered in the XSTM analysis. The word2vec word embeddings vectors are not normalized. Tokens

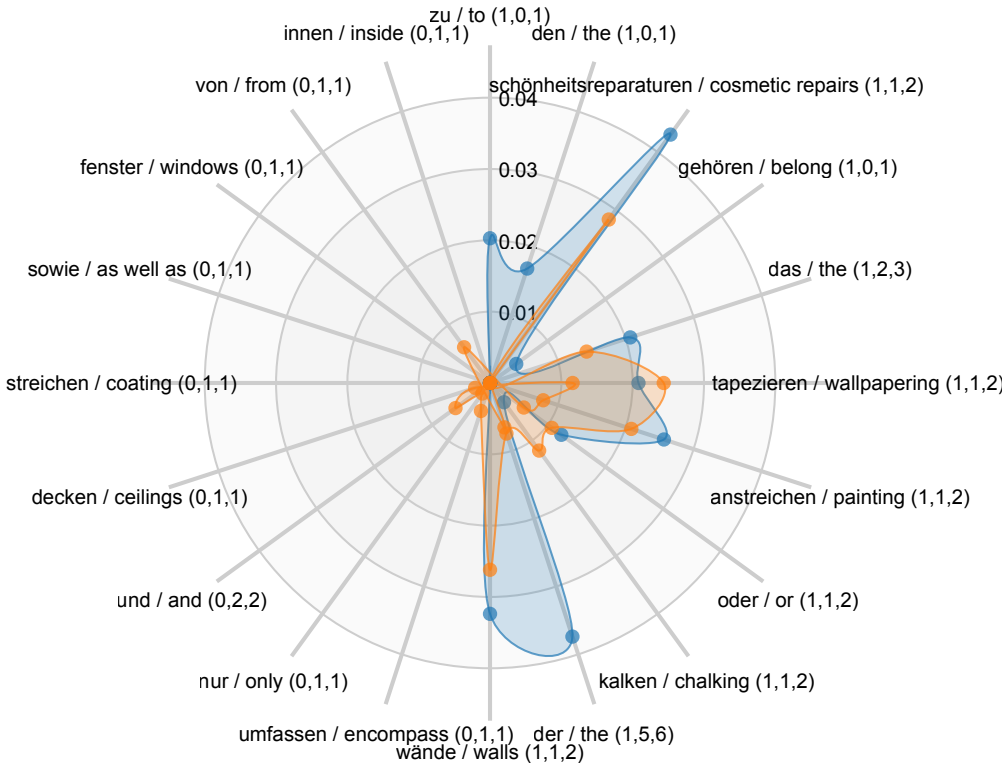

Figure 6.1.: Visualization of the results of an XSTM analysis of a single match (Landthaler et al. (2018a)).

that occur multiple times in the query and matching sentence strings contribute multiple times to the significance scores.

The query string (significance scores displayed in orange) is *"Zu den Schönheitsreparaturen gehören das Tapezieren, Anstreichen oder Kalken der Wände. (To the cosmetic repairs belongs the wallpapering, painting or chalking (of) the walls.)"*. The matching sentence in the legal comment (significance scores displayed in blue) is *"Schönheitsreparaturen umfassen nur das Tapezieren, das Anstreichen der Wände und Decken sowie das Streichen der Fenster von innen. ((The) cosmetic repairs encompass only the wallpapering, the painting (of) the walls and ceilings and the coating (of) the windows from the inside."*).

The axes of the radar chart represent the significance scores, i.e., the contribution, of the participating words. In brackets the following information is displayed: German token, English translation, token occurrence frequency in the query, token occurrence frequency in the matching sentence and token occurrence frequency in both, query and matching sentence). Note that the example has been simplified for better visualization. An interesting observation is that nouns seem to contribute most to the similarity among query and matching sentence. It supports the common hypothesis that nouns contribute most to topical search. However, it is difficult to derive meaningful information from a single match. In the next section, several matches are investigated in an aggregated fashion.

6.2.2. Case Study: Aggregated Matches

The XSTM Approach can be applied to single matches. Due to the small sizes of matching strings in the legal comments, the results of an XSTM analysis applied to the SENT Approach is suited best for visualization purposes. However, the XSTM Approach can also be used to compare WE-DF and TF-IDF vector space models. The CHAPTER and TF-IDF Approaches can be compared because they encode the same text segments. Here, the XSTM Approach is applied to several matches. Per match, one or several occurrences of a token contribute to the significance score of a token. The contributions to the significance score can easily be aggregated over several matches. Typically, the token vocabulary increases. Several technical aspects need to be considered for the (aggregated) XSTM analysis besides a selection of pre-processing technologies for both vector space models. The WE-DF text-representation further requires to select a hyper-parameter configuration for the word2vec algorithm to train a word embeddings model.

Several technical aspects can be varied for an aggregated XSTM analysis:

- **A selection of matches:** The top N ranked results of a query can be considered or a selection of matches according to an evaluation set, for example, the broad or the narrow evaluation set.
- **Aggregation of significance scores:** The significance scores of the query, matching text segments or both can be aggregated.
- **Aggregation of multiple occurrences:** Tokens that occur multiple times in a match can contribute once or multiple times per match to the significance scores of the tokens[1].
- **Stopword removal:** Stopwords can be removed or not.
- **Normalization:** For the WE-DF vector space model, the word embeddings vectors can be normalized to unit length or not.

The outputs of an aggregated matches XSTM analysis are ranked lists of tokens in the vocabulary. The tokens can be sorted according to the significance score in descending order. Example results of the top 15 most significant tokens are displayed in Table 6.1.

The excerpt of the ranked lists displayed in Figure 6.1 compares the WE-DF and TF-IDF vector space models, in particular how tokens are weighted by the vector space models. The matches are selected as follows: The contract clauses tagged with the *Landlord duty* tag serve as queries. For each query, the correct ranked legal comment chapters in the top 15 results are selected according to the *narrow* evaluation set for the tag *Landlord duty*. For all tokens, an XSTM analysis is carried out. The tokens are ranked by the significance scores in descending order. Both query and matching legal comment chapters are considered and Stopwords are removed. For the WE-DF vector space model, the raw word embeddings vectors, i.e., not normalized to unit length, are used.

Several tokens in the top 15 tokens of the ranked lists are shared. For example, the most significant token is *"schönheitsreparaturen" ("cosmetic repairs")*. It is an important term for cosmetic repairs related content. The occurrence frequency of the terms in the considered matches does not seem to be a crucial aspect for the significance scores. Nouns, verbs and abbreviations are included in the most significant tokens. I.e., the word type also does not seem to be a decisive factor. The selection of matches serves as

[1] For a "pure" WE-DF vector space model, the significance scores of multiple occurrence of tokens contribute multiple times

	WE-DF		TF-IDF	
Rank	**Word (EN)**	**Sig. Score (Occ. Freq.)**	**Word (EN)**	**Sig. Score (Occ. Freq.)**
1	schönheitsreparaturen (cosmetic repairs)	0.3091 (1010)	schönheitsreparaturen (cosmetic repairs)	0.2107 (1010)
2	unrenoviert (unrenovated)	0.0785 (28)	wohnung (flat)	0.0405 (3696)
3	vertragsbeginn (contract start)	0.0489 (35)	bgh (German Court)	0.0149 (5611)
4	renovierten (renovated)	0.0485 (14)	vermieter (landlord)	0.0145 (8944)
5	übergeben (hand over)	0.0333 (72)	mieter (tenant)	0.0140 (11230)
6	verantwortungsbereich (field of responsibility)	0.0326 (40)	vertragsbeginn (contract start)	0.0138 (35)
7	mieter (tenant)	0.0314 (11230)	§ (§)	0.0135 (13552)
8	tragen (carrying)	0.0220 (281)	arbeiten (working)	0.0134 (328)
9	durchführung (execution)	0.0190 (414)	vgl (see)	0.0123 (1987)
10	übergabe (handover)	0.0181 (112)	durchführung (conduct)	0.0116 (414)
11	durchzuführen (carrying out)	0.0172 (111)	verpflichtet (obliged)	0.0114 (1171)
12	kosten (costs)	0.0166 (2954)	pflichtverletzung (breach of duty)	0.0103 (105)
13	bgh (German Court)	0.0155 (5611)	rauchen (smoking)	0.0081 (31)
14	vermieter (landlord)	0.0130 (8944)	wum (-)	0.0078 (6230)
15	§ (§)	0.0124 (13552)	unrenoviert (unrenovated)	0.0075 (28)
	Vocabulary size: 487		**Vocabulary size: 503**	

Table 6.1.: Sample terms for most significant words according to an XSTM analysis. Significance scores are aggregated over correct matches over all cosmetic repair tags.

an example where different matches for the two vector space models are considered. Hence, the token vocabulary size differs for the two vector space models. To better compare the two vector space models, an identical set of matches between the approaches is beneficial. A possible selection of matches so that the token vocabularies of two vector space models are equal is to select all correct matches independent of their ranking positions in the result lists. Thus, for both vector space models, the same set of matches is selected.

The following experiments show how to use the ranked token lists of an XSTM analysis to compare vector space models in a more comparable fashion. The idea is to measure the similarity of ranked token lists. The similarity of the ranked token lists can be used as an indicator of the commonalities and differences of the two vector space models. Spearman's rank correlation coefficient is a measure that can be used to measure correlations between ranked lists, i.e., to assess the similarity of two ranked lists.

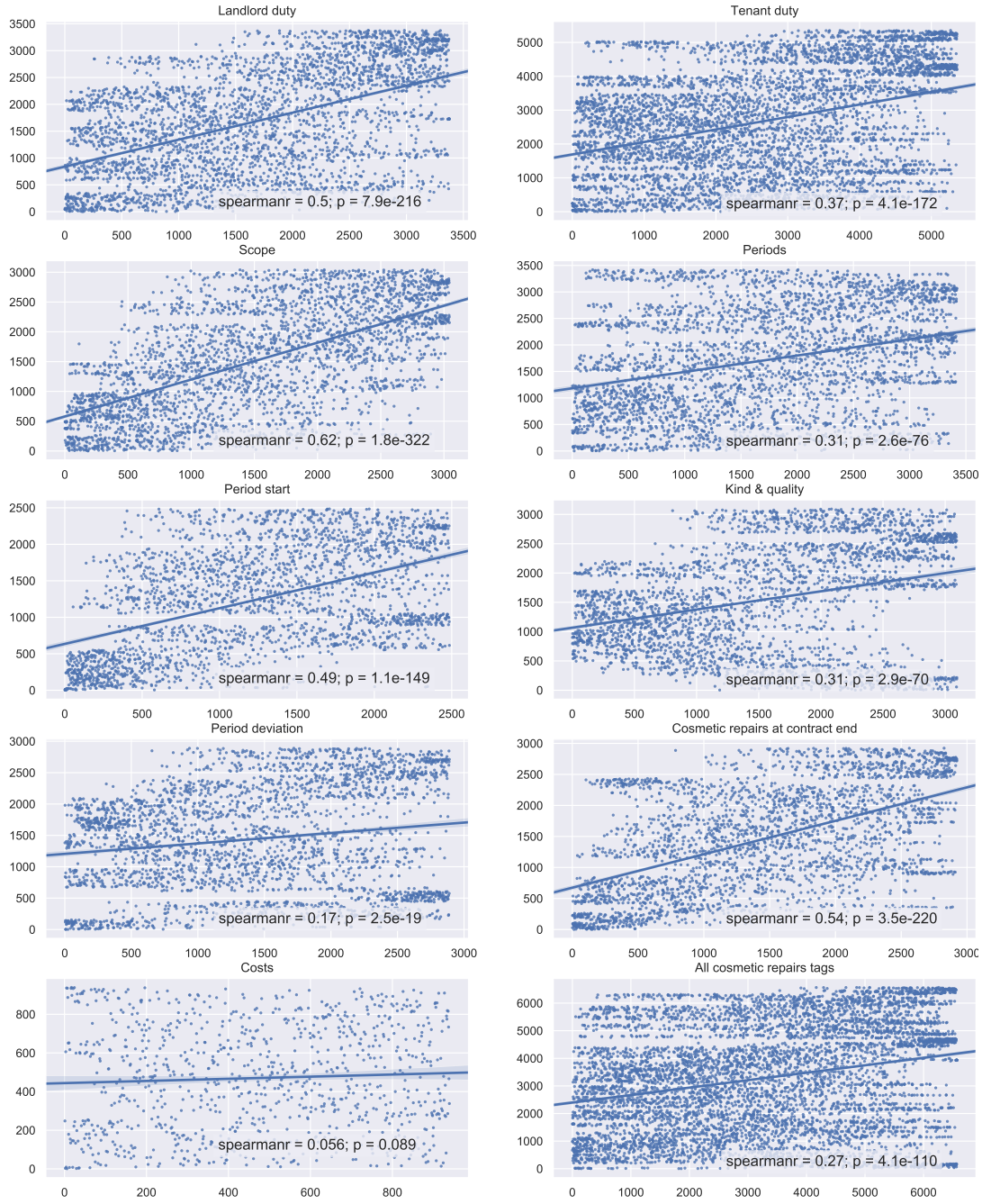

Figure 6.2.: Rank correlation analysis of WE-DF and TF-IDF vector space models for each cosmetic repair tag and all cosmetic repair tags.

Figure 6.2 shows the results of a Spearman's rank correlation analysis. The axes represent the ranked tokens. Thus, a datapoint close to the diagonal indicates tokens that are similarly ranked by both vector

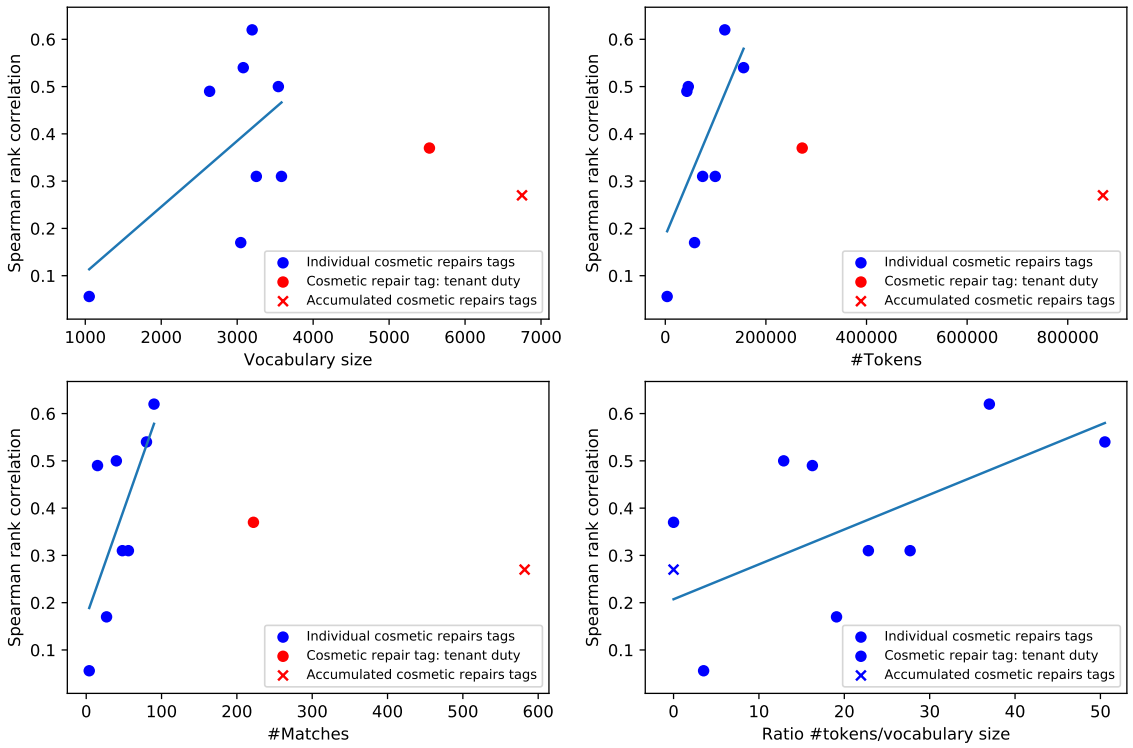

Figure 6.3.: Analysis of potential influence factors for Spearman rank correlation coefficients.

space models. All matches between contract clauses and legal comment chapters are used according to the narrow evaluation set for all cosmetic repair tags as well as an aggregation over all cosmetic repair tags together. The word embeddings vectors used for the WE-DF vector space model are normalized to unit length and stopwords are removed.

For some cosmetic repair tags, the Spearman's rank correlation coefficient is quite large with values around 0.62, i.e., the WE-DF and TF-IDF vector space models tend to weight tokens similar. Since the two vector space models encode the same text segments, *some* similarity can be expected. However, for other cosmetic repair tags, for example, *Period deviation* and *Costs*, the Spearman's rank correlation coefficient is rather small.

The next experiment is an attempt to explore potential influence factors on Spearman's rank correlation coefficient. Candidate influence factors are the number of matches, the token vocabulary size and the overall number of tokens. Additionally, the ratio of the total number of tokens to token vocabulary size is considered. Figure 6.3 displays an analysis of the different influence factor candidates on the Spearman's rank correlation coefficient. Dots represent values of cosmetic repair tags. The cross represents values that are aggregated over all cosmetic repair tags. Blue data points are included for the calculation of a regression line that is also depicted in Figure 6.3. Data points colored in red are considered as outliers. In the regression lines of the first three influence candidates no clear trend is

visible because of their steepness and strong outliers that are removed from the calculation of the regression lines. The result can be interpreted that there seems to be a correlation among Spearman's rank correlation and the ratio of the total number of tokens and vocabulary size. For the analysis of the Spearman's rank correlation and the ratio, all data points are included in the calculation of the regression line.

The results suggest that the occurrence frequency of tokens in matches have an impact on the similarity of the XSTM analysis of WE-DF and TF-IDF vector space models. However, these empirical results should be considered with caution. Only a few data points are considered (with a maximum of ten data points) and the potential correlations are not clearly visible. The next section gives a rationale that can explain the results. Thus, the experiments mainly serve as a blueprint of how the XSTM analysis can be used to compare different vector space models.

6.3. Analytical Comparison of the WE-DF and TF-IDF Vector Space Models

WE-DF (for the CHAPTER Approach) and TF-IDF are vector space models. A single vector represents a text segment, here legal comment chapters. The vectors that represent text segments are a linear combination of two types of vectors. One type of vectors represents corpus wide statistics. The other type of vectors represents document-specific token occurrence or token co-occurrence statistics. Both vector space models can be seen as complex statistics from an abstract point of view.

Table 6.2 shows a comparison of how WE-DF and TF-IDF document representations apply weighting schemes. Both vector space models are based on global and local components, i.e., vectors that represent global (corpus-wide) and local (document-specific) statistics. Both vector space models use cosine similarity measure. For the following considerations, it is assumed that the two vector space models encode the same text.

Word embeddings models and word embeddings vectors encode token-token co-occurrence statistics. Word embeddings vectors are global statistics because training samples used to train word embeddings models are sampled from the full corpus. One could argue that this is not a corpus-wide statistic like the corpus-wide TF, because word embeddings models are calculated from samples that typically have a size less than the document-sizes (window size). However, the samples are obtained from the full corpus. The TF vectors have a high dimensionality in comparison to word embeddings vectors. It is unlikely that the two global WE and TF vectors become very similar.

The global vectors are weighted with document-specific vectors. The IDF vectors increase the number of zero entries in the TF vectors. In general, the DF does not lead to zero entries when multiplied with WE vectors. However, the DF and the IDF address the same tokens. In the case that tokens occur many times in a document, i.e., ratio of the total number tokens to the vocabulary size of a document increases, then the DF and the IDF should become more dissimilar because the IDF is the inverse of the DF.

In contrast to the previous pairs of vectors, the TF and the DF vectors tend to become similar under certain conditions. The DF is a fraction of the TF. The DF vectors become more similar to the TF vectors when the local statistics of DF approximate corpus-wide TF statistics. This can be the case for large

documents. The more text of the corpus is encoded or when tokens occur many times, i.e., the ratio of the total number tokens to the vocabulary size over several documents increases, then the significance scores of an XSTM analysis can become more similar. This can explain the empirical results obtained in the previous section.

It can be speculated that the WE and IDF vectors share characteristics, too. It was shown that down-sampling of frequently occurring terms is very useful for the training of word2vec word embeddings models. To some degree, the IDF and the logarithm of IDF also down-sample frequently occurring tokens.

These considerations can explain the previously obtained empirical results to some degree. However, no formal proof is given. A more in-depth investigation of word embeddings with IDF weighting, WE-IDF, that has been proposed and explored by Ferrero et al. (2017), Nagoudi et al. (2017) and Júnior et al. (2017) could further underline the rationale but is left to future research.

6.4. Discussion

In this chapter, the XSTM Approach is presented. The XSTM Approach transfers the idea of conducting a sensitivity analysis from machine learning classifiers to text similarity measures. The XSTM Approach can be used to assess the significance of tokens to text similarity and to compare different vector space models. The XSTM Approach removes tokens one after another from two text segments and the text similarity is re-calculated. The outputs of an XSTM analysis are significance scores for tokens, i.e., a score of the significance of tokens to the text similarity of two text segments.

The XSTM Approach can be applied to a single match of a Semantic Text Matching problem instance, for example, when vector space models and text similarity measures are used. The significance scores of the individual tokens can be aggregated over several matches of a Semantic Text Matching problem. In the case that the text segments are shared between two vector space models, then an XSTM analysis can be used to compare vector space models. The WE-DF and TF-IDF vector space models are compared empirically. Further, the WE and TF-IDF vector space models are compared analytically.

The significance scores of the tokens are aggregated over several matches and ranked lists of the tokens are derived by sorting by the aggregated significance scores. The ranked lists of two vector space models can be compared with Spearman's rank correlation coefficient. Different potential influence factors on the similarity of the two vector space models are investigated. The ratio of the total number

Characteristic	WE-DF	TF-IDF
Base vectors (corpus component)	Token-token co-occurrence statistics (WE)	Term-frequency (TF)
Document weighting (document component)	Document-frequency (DF)	Inverse document-frequency (IDF)
Text segment vector	**WE * DF**	**TF * log(IDF)**
Similarity measure	cosine similarity	cosine similarity

Table 6.2.: Analytical comparison of WE-DF and TF-IDF vector space models.

of tokens and vocabulary size has the potential to be a useful indicator of the similarity between WE-DF and TF-IDF vector space models.

The primary result of this chapter is that WE-DF and TF-IDF vector space models can become similar under certain circumstances. Both vector space models can be considered as complex statistics to represent documents. The two vector space models are compared analytically. Both text representations are weighting schemes of corpus wide and document-specific statistics. Eventually, the WE-DF and TF-IDF vector space models can become similar. For example, when large documents are included, the DF vectors can become similar to the TF vectors to some degree.

The results of this chapter tend to confirm the results of Chapter 5 and the results presented in Landthaler et al. (2016). The WE-DF text representation is most useful to represent small text segments, while larger documents (and corpora) are better represented with the TF-IDF text representation because for larger text segments, word embeddings based vector space models lose their ability to conduct implicit query expansion and the results that can be achieved by the representations of the two vector space models can become similar for documents of medium size. For even larger documents, the TF-IDF vector space models starts to be better than the WE-DF vector space model. However, it remains an open question what the tipping point of document sizes is.

The empirical investigations are limited in generalizability due to the small number of data points. A solution could be to apply data augmentation approaches, for example, by using many randomly sampled selections of matches according to the evaluation sets. Moreover, no formal proofs are carried out. As a consequence, the research presented in this chapter is not complete and should be considered as a blueprint of how the XSTM Approach can be used. However, it remains an open question, how the results of a sensitivity analyses can be a reasonable tool for users.

The XSTM Approach could also be used to investigate the significance of the different dimensions of the word embeddings vectors. Schütze (1992) analyzed the dimensions of word representations of count-based DSMs. The result is that all dimensions of (count-based) word embeddings vectors contribute to the word-to-word similarity. The result was obtained from a word sense disambiguation task. For an XSTM analysis of word embeddings models dimensions, a similar result could be expected due to the strong relationship of word representations calculated from count-based and predictive DSMs.

Another possibility to use the XSTM Approach is to compare other text similarity measures such as the Levenshtein distance or the Word Movers Distance to derive further insights from text similarity measures. Other text representations and vector space models have been reported or can be constructed. A comparison of IDF-weighted word embeddings vectors to represent text segments and TF-IDF could show an even stronger similarity than the WE-DF and TF-IDF text representations and vector space models but is left to future work.

Any intelligent fool can make things bigger and more complex… It takes a touch of genius – and a lot of courage to move in the opposite direction.

Albert Einstein

CHAPTER 7

Conclusion

7.1. Summary

Legal research is a highly relevant activity in the legal domain. Nowadays, much legal knowledge is available in digital form. As a consequence, legal information retrieval, i.e., supporting legal experts when conducting legal research with technical solutions, is highly relevant. The legal language has many specific characteristics, for example, synonymy is used a lot. Moreover, legal documents are often long, of different types and constitute large corpora. Relevant information is typically scattered among several documents. Query expansion is one way to address these challenges of legal information retrieval. This thesis investigates different word embeddings based approaches to support and improve query expansion for legal information retrieval in the German legal domain.

A common approach to query expansion is to maintain an ontology, for example, a thesaurus, to expand the terms of a search query. The life-cycle of a thesaurus involves several business activities such as the initial creation and maintenance. In this thesis, the specific aspect of extending an existing thesaurus using word embeddings is investigated. Moreover, the WE-DF vector space model is investigated for its capability to conduct an implicit form of query expansion.

The following paragraphs answer the derived research questions. The main research question is answered after the derived research questions are addressed.

> ***Research question 1:*** What word embeddings algorithms exist and what semantic aspects do they encode?

Word embeddings are predictive DSMs. Five contemporary word embeddings algorithms have been identified. Three word embeddings algorithms, in particular, word2vec, FastText and GloVe, are considered in depth.

Two currently prominent representatives of predictive DSMs are the word2vec and the FastText algorithms. Both are sub-sampling approaches and calculate co-occurrence statistics of atomic units in a corpus. For the word2vec algorithm, the atomic units are tokens. The FastText algorithm extends the word2vec algorithm's approach to character n-grams of tokens as atomic units. Thus, FastText imitates syllable embeddings that can be accumulated to word embeddings vectors. Sub-sampling approaches can be thought of to approximate the results of matrix factorization approaches, i.e., count-based DSMs[1]. The GloVe algorithm can be considered as the most recent representative of count-based DSMs but is commonly called a word embeddings algorithm. The most recent word embeddings algorithms ELMo (Peters et al. (2017)) and BERT (Devlin et al. (2018) are deep learning algorithms and have not been considered in this thesis.

A major semantic aspect that is encoded in the word embeddings models calculated with the word2vec, FastText and GloVe word embeddings algorithms is synonymy. The different word embeddings algorithms lead to results of varying types, quality and quantity. These insights are not derived from the literature and, thus, revealed by answering the next research question.

> ***Research question 2:*** How can word embeddings be used to extend legal thesauri and how good are the results that can be obtained with such approaches?

The thesaurus extension task is different from the standard task of thesaurus reconstruction that is usually considered in research because the technology-assisted thesaurus extension requires a qualitative evaluation. Qualitative evaluations require great effort.

The creation and extension of a thesaurus are challenging tasks. A thesaurus in this thesis is considered to consist of synsets. The scope of a synset can be semantically broad or narrow. The synonymy relation among two terms can be judged quite differently by different domain experts. On top, the boundary between two or more synsets is hard to determine even for humans.

In order to answer the research question, one traditional count-based DSM (JoBimText Approach) and five word embeddings based approaches are compared qualitatively and where appropriate also quantitatively. All experiments regarding the thesaurus extension are conducted on a German tax law corpus and an existing thesaurus that is specifically maintained for the text corpus.

All word embeddings based approaches except the Label Propagation Approach use the cosine similarity to rank terms of a word embeddings model's vocabulary for a query vector. For each query vector, a candidate term list is calculated. In this thesis, only fixed-length candidate term lists are considered. All word embeddings based thesaurus extension approaches can be combined with any word embeddings algorithm.

For the Single Word Approach, the query vectors are individual word embeddings vectors that represent terms of the existing synset. The Single Word Approach is used to compare the JoBimText Approach to a word embeddings based approach because the results of the other word embeddings based approaches are less comparable to the JoBimText Approach. However, the results of the Single Word Approach can be compared to the results of the Synset Vector Approach to some degree.

Two approaches use synset embeddings: The Synset Vector Approach and the Intersection Approach.

[1] Predictive DSMs are superior to count-based DSMs because predictive DSMs are able to store information in "slack variables" and thus are more flexible in storing information while count-based DSMs allow only for "one" solution.

For the Synset Vector Approach, the synset embeddings vector is calculated as the mean of the word embeddings vectors of all terms of an input synset. The synset embeddings serve as the query vector. The Intersection Approach combines several candidate term lists calculated with the Synset Vector Approach but that have been calculated with different hyper-parameter configurations of the word embeddings algorithm.

The Label Propagation Approach constructs graphs from word embeddings models. The graphs are the input to label propagation algorithms that spread labels from labeled nodes to unlabeled nodes. When adequately modeled, then a partition of a word embeddings model's vocabulary into synsets is calculated.

A commonly known handicap of word embeddings is that the quality of the word embeddings vectors cannot be measured directly but only indirectly assessed by evaluating a downstream NLP task. All word embeddings algorithms and thesaurus extension approaches come with a large number of hyper-parameters that affect the quality of the resulting word embeddings model. A comparison of the RP-Score and the standard MAP shows that, in general, the two relevancy measures lead to comparable results regarding the evaluation of hyper-parameters. However, the RP-Score is particularly well suited for assessing the quality of word embeddings algorithms for the thesaurus extension task because more relations can be exploited and a train-/test-set split of the thesaurus is not required. The Iterations, Min-Count and Model Architecture hyper-parameters are identified as the most important hyper-parameters of the word2vec and FastText word embeddings algorithms[2].

The different word embeddings algorithms show very different results. The GloVe algorithm (that should be classified as a count-based DSM but is usually called a word embeddings algorithm), shows significantly worse results than the word2vec and FastText algorithms that are predictive DSMs. The FastText algorithm tends to suggest mostly syntactically similar terms, while the word2vec algorithm mainly suggests semantically related terms that are not necessarily syntactically similar. Depending on the use case, an adequate word embeddings algorithm should be chosen.

In general, word embeddings based approaches constitute a major leap forward when compared to a traditional count-based DSM. In contrast to word embeddings, count-based DSMs favor frequently occurring tokens. Also, the GloVe algorithm tends to favor frequently occurring tokens but less strong than the traditional count-based DSM algorithm used in the JoBimText tool suite. From another perspective, word embeddings are significantly better at representing terms that occur less frequently.

Word embeddings models are trained on the context of words. If not enough context samples are provided for a particular word, then the quality of the particular word embeddings vector of the words is likely of low quality. A standard solution to mitigate this problem is to remove words from a word embeddings model (for example, by setting the Min-Count hyper-parameter of a word embeddings algorithm to a value larger than zero[3]). A downside to this solution is that words that do not occur frequently enough cannot be considered in downstream tasks. This can be seen as an instance of the out-of-vocabulary problem[4]. The Intersection Approach investigated in this thesis can be seen as an alternative approach to setting the Min-Count hyper-parameter to a value larger than one. Through the intersection of candidate lists for synsets calculated from word embeddings models trained with

[2] for the considered use case and dataset

[3] The investigated word embeddings algorithms remove the tokens before the training procedure starts.

[4] The out-of-vocabulary problem is more general and also encompasses cases where new or unseen documents are processed and therefore, words occur that do not have a word embeddings vector.

different hyper-parameters, unstable word embeddings vectors are filtered. The Intersection Approach leads to slightly improved overall results in the qualitative evaluation and reasonable and also less frequently occurring terms are identified. A large enough value of the Min-Count hyper-parameter results in stable word embeddings models and the Intersection Approach will not work in this case.

Label propagation algorithms are used in other domains for ontology extension and good results are reported. However, for all label propagation algorithms investigated in this thesis, the results of both quantitative and qualitative evaluations are worse than the results of the Synset Vector Approach.

The Synset Vector Approach is a reasonable trade-off in terms of the quality of the results and the complexity of its application. The Synset Vector Approach performs significantly better for synsets with more than one input term than the Single Word Approach. Taking the mean of the word embeddings vectors of all terms in input synsets significantly helps to capture complex semantics of the input synsets.

Despite impressive results in general NLP applications, the signals of current word embeddings algorithms are too noisy to enable full automation. This applies particularly to the legal domain, where often both high precision and high recall are crucial for legal information retrieval. The effect is multiplied by the problem of low-quality word embeddings vectors for infrequently occurring words. Nevertheless, word embeddings can successfully be used to calculate a significant amount of correct suggestions that can be presented to human experts for review. The experiments conducted in this thesis suggest that word embeddings based approaches can calculate, on average, around 20 to 50 percent correct synonym candidates in the first ten calculated candidate terms per synset that are not present in the existing thesaurus yet.

The good results of the qualitative evaluation cannot be derived from the results of the quantitative evaluation of a thesaurus reconstruction task. Word embeddings based approaches calculate additional correct synonyms even for a large existing thesaurus. From another point of view, correct synonyms are not present in the existing thesaurus, i.e., a "gold standard" thesaurus does not exist for real-world datasets because a thesaurus is never "complete". Thus, a quantitative evaluation using an existing thesaurus will not evaluate the candidate terms calculated with word embeddings based approaches in a proper way. However, an existing thesaurus can still be used to optimize the hyper-parameters of the algorithms involved.

> ***Research question 3:*** How could word embeddings be used to conduct query expansion without maintaining a (legal) thesaurus and how good are the results that can be obtained with such approaches?

The WE-DF text representation can be used to represent, for example, sentences, paragraphs or documents by accumulating word embeddings vectors. Used in this way and when used together with a text similarity measure such as cosine similarity, the WE-DF text representation constitutes an alternative vector space model to other vector space models such as TF-IDF. The WE-DF vector space model implicitly conducts query expansion.

Several problem classes where WE-DF based representations could be used are identified. Problems that can be framed as Semantic Text Matching problems are suited best to use WE-DF based text representations regarding computational performance.

A use case in German tenancy law is investigated. The use case can be framed as a legal information

retrieval task, where tenancy template contract clauses serve as input to a natural language search query. Chapters from legal comments are retrieved for the queries. A dataset with six template tenancy contracts and three legal comments is constructed.

The CHAPTER and the SENT Approaches that use the WE-DF vector space model are explored. The CHAPTER Approach encodes full legal comment chapters with the WE-DF text representation. The SENT Approach represents legal comment chapters as a set of sentences that are encoded with the WE-DF text representation. The TF-IDF Approach uses the TF-IDF vector space model and serves as a baseline approach. However, two different TF-IDF implementations are considered. The gensim implementation of TF-IDF and Elasticsearch's *More like this* functionality.

For a quantitative evaluation of the approaches, an evaluation set is required. Contract clauses and corresponding legal comment chapters on the topic of cosmetic repairs have been identified. In order to reduce the labeling effort, a method that uses tags as an intermediary between contract clauses and legal comments is applied. Two variants of the evaluation set are used to compare the approaches (and TF-IDF implementations) that reflect a broad and a narrow scope of user intentions.

Different pre-processing options for the approaches are investigated. However, the impact of the different pre-processing options tends to be small. In contrast to the results of the thesaurus extension task, terms that occur only a few times still improve the results of the WE-DF vector space model based approaches. A reason can be the tiny training corpus.

The WE-DF vector space model does conduct an implicit form of query expansion. However, the performance of the CHAPTER Approach is worse than the performance of traditional TF-IDF vector space models. The Elasticsearch implementation of TF-IDF performs significantly better than the gensim implementation of TF-IDF. The SENT Approach performs best for the top-ranked results.

Another conclusion can be drawn from the results. The WE-DF text representation is best suited to encode small text segments like sentences. Due to the comparably low number of dimensions, the WE-DF text representation of large text segments superposes the individual semantics encoded in word embeddings vectors. Therefore, larger documents are better represented with traditional vector space representations such as TF-IDF except in cases where it is reasonable to represent larger documents as a set of smaller text segments.

An interesting result is that blindly adding more training data to the training corpus does not lead to better, or, in parts, dramatically worse results in the investigated use case. Eventually, this should be considered in general for applications in the (German) legal domain.

> ***Research question 4:*** How do lawyers judge natural language search that uses WE-DF vector space models in comparison to traditional keyword search?

A web application and a Microsoft Word AddIn that implement the use case have been developed. A web application offers greater flexibility while a text processor AddIn integrates deeper with workflows that are nowadays typical for lawyers. For both legal information retrieval tools, two search methods have been included. One method, the Selection Search, enables users to select a text segment of interest in a contract as input to a search query. The Selection Search is a form of a natural language search. The other method is a traditional Keyword Search that leverages Elasticsearch's search functionality and serves as a baseline. For both search methods, chapters from legal comments are retrieved for the search queries.

The Microsoft Word AddIn, the search methods and the different vector space models are evaluated with a user study. The results of a System Usability Score analysis shows that both search methods Selection Search and Keyword Search are evaluated as "good". The results suggest that lawyers that are experienced in a law domain prefer more control. The traditional Keyword Search allows for more control than the Selection Search. The Selection Search can, however, be a useful tool for lawyers inexperienced in a law domain. The quality of the search results is identified as the most important aspect of a legal information retrieval system. The lawyers were asked to judge the different approaches that deliver search results, i.e., the CHAPTER, SENT and TF-IDF Approaches. The lawyers could not identify significant quality differences among the approaches. The preference for one of the two search methods is polarized. One-half of the participants prefer the Selection Search, while the other half prefers the traditional keyword search. Thus, the Selection Search can serve as a complementary search method to the traditional keyword search for legal information retrieval. The participating lawyers stated that they would like to use the Word AddIn in the future.

> ***Research question 5:*** How similar or different are WE-DF vector space models in comparison to traditional TF-IDF vector space models?

A major trend in the artificial intelligence research community that recently has been picked up by the AI&Law research community is XAI. This thesis discusses an adaption of a sensitivity analysis approach to similarity measures. The XSTM Approach successively removes words from pairs of text segments and the text similarity is recalculated. This leads to a significance score for each word. The significance scores can be used as a rough measure of how vector space models weigh words. Moreover, the significance scores can be aggregated over several text similarity comparisons.

The XSTM Approach can be used to assess the differences and commonalities of the WE-DF and the TF-IDF vector space models. The results of both empirical and analytical investigations suggest that the WE-DF and TF-IDF vector space models can become similar under certain conditions, for example, when larger documents are encoded. However, due to only a few datapoints, the research should be seen as an extended outlook and as a blueprint how XSTM analyses can be used to compare text representations, text similarity measures and vector space models.

> ***Main research question:*** Do word embeddings models capture semantic aspects that can be used to improve semantic search for German legal information retrieval?

A major aspect that is encoded in word embeddings models calculated with the word2vec, FastText and GloVe algorithms is synonymy. Thus, word embeddings are particularly well suited to improve query expansion for legal information retrieval.

In the case that a thesaurus exists and a high level of control over the query expansion is desirable, word embeddings are well suited to extend a thesaurus and a significant amount of correct and new synonym candidates can be calculated. Word embeddings are significantly better suited to extend thesauri than traditional count-based DSMs. If it is important to identify syntactically similar tokens, then the FastText algorithm is the best choice. If the goal is to identify more semantically similar but syntactically dissimilar tokens, then the word2vec algorithm is the best choice. However, the signals encoded in word embeddings models are too noisy to enable full automation and human reviewers are required to review the results of word embeddings based approaches.

In the case that small text segments are present or large documents can be split into smaller text

segments, and less control on the query expansion is required, then the WE-DF vector space model can be used to conduct an implicit form of query expansion. Otherwise, for larger documents, the traditional TF-IDF vector space model is often a better solution. The WE-DF vector space model can further be used to construct a natural language search that is preferred by up to 50% of lawyers in comparison to a traditional keyword search.

A big challenge is the large degree of freedom in composing the natural language. For example, a phrase of 3 tokens and a vocabulary size of 10 allows for 1000 possible combinations. A large subset of the 1000 possible combinations constitutes potentially correct phrases. The noisy signals encoded in word embeddings models and the large degree of freedom of natural language demand for unsupervised approaches that can leverage additional information resources or exploit existing information better. An important aspect of legal information retrieval is that legal experts would like to have explanations on why and how legal information retrieval systems determine relevant documents.

7.2. Limitations

The research described in this thesis can be classified as applied science. The research is, for the most part, inductive through empirical experiments. This is a common approach in data science. However, the generalizability of the results and derived rationales is limited due to the limited number of datasets and experiments.

The technology zoo available to scientists nowadays is vast and continuously growing. The sheer amount makes it infeasible to consider all possible applicable technologies. In addition to that, data science technologies typically come with many hyper-parameters. The identification of an optimal set of hyper-parameters is a broad field on its own and due to the curse of dimensionality, heuristics need to be applied. In this thesis, each hyper-parameter is considered on its own. The large number of hyper-parameters makes it infeasible to investigate each hyper-parameter in exhaustive depth.

There are many life-cycle activities around thesauri. In this thesis, only the use case of thesaurus extension is considered. A limitation of the investigations in this thesis is that the effect of extending thesauri on legal information retrieval has not been investigated. The thesis does not cover the identification of synsets, i.e., the initial creation of a thesaurus. However, legal publishers, i.e., organizations that maintain large legal corpora, typically log the search queries of their users. A good starting point to identify synsets relevant for a thesaurus for query expansion is to start with terms that frequently occur in search queries. Furthermore, the integration of synonym suggestions into thesaurus maintenance tools has not been considered. Neither pre-trained word embeddings models nor extensions of the training corpus are considered for the thesaurus extension use case. Many use case challenges that have been identified are not addressed specifically or exhaustively. For example, open compound words constitute a huge challenge, especially for NLP applications on the German language. The identification of multi-token terms in the training corpus has not been addressed in greater detail. The research is limited to the identification of synonym relations. Other relation types, such as antonyms, are not covered by this thesis.

The evaluation set that has been constructed for the Semantic Text Matching use case does not cover all granularities of users' search intentions. The task of text segmentation that is a preliminary requirement of addressing Semantic Text Matching problems is not addressed in depth. Other text rep-

resentation technologies such as doc2vec, Skip-thought vectors or sent2vec have not been explored. Many other approaches to query expansion that have been reported in the literature are not compared to the approaches considered in this thesis.

The results of the user study are limited due to the small number of participants that is not representative of the population of German lawyers. The construction of user studies is complex and many detailed decisions need to be made. This opens the space for a vast amount of different possibilities to construct user studies and many other ways to design the user study could have been considered.

The XSTM Approach is limited in its applicability and in the interpretation of the results. In particular, no obvious way of how users can leverage the results of an XSTM analysis is reported. The empirical results that use the XSTM analysis are limited in many ways. Only two vector space models are compared. In particular, the WE-IDF text representation and vector space model have not been investigated. The small number of datapoints do not allow generalizable results. For the analytical comparison of the WE-DF and TF-IDF vector space models, no formal proof is given. However, the empirical investigations are declared as a blueprint of how the XSTM Approach can be used to compare vector space models and as an extended outlook.

7.3. Outlook

The limitations listed in Section 7.2 are good starting points for future research. An important task for future research is the initial creation of thesauri. The synonym candidates suggested with word embeddings based approaches contain a significant amount of relevant candidates, but the signals are too noisy to enable full automation so far. Stronger filtering of the results might increase the precision of the results, for example, filtering for specific POS-tags. The sense2vec (Trask et al. (2015)) approach could be used. The sense2vec approach appends a delimiter character and a POS-tag to tokens. The training corpus could be tagged in this way, which would enable filtering for POS-tags in the candidate lists. However, POS-tags are a comparably simple grammatical attribute. Thus, additional linguistic attributes like semantic role labels could be used, too. Integrating additional linguistic and semantic resources seems to be a promising starting point to improve the automated creation and extension of thesauri, for example, to tackle the challenges of open compound words.

The Semantic Text Matching problem is present in many non-legal but also particularly in legal applications, namely argumentation mining and information retrieval. On the one hand, the results indicate that legal information retrieval functionality could be integrated better into legal workflows. On the other hand, the results also point out how important (but also how difficult) an understanding of the user's intention is for legal research. Further research will be required to better understand legal professionals and their needs in order to optimize legal information retrieval systems. From a technical point of view, text similarity measures are heuristics but also scalable solutions and, therefore, are used a lot in information retrieval. However, in comparison to word embeddings, even standard and straightforward approaches like TF-IDF do not lead to good "explanations" for the search results. Further research is inevitable to understand word embeddings better. A fruitful direction of research will be the construction of legal information retrieval technologies that are accompanied by some rationale that "explains" search results to users.

In general, AI&Law research often focuses on applying new or advanced technologies to previously

published use cases and/or identifies new use cases. Typically, a "gold standard" dataset is used to evaluate approaches. Most of the time, the creation of a "gold standard" is a manual, time-consuming and challenging task. The different results of the quantitative and qualitative evaluation of the thesaurus extension could give rise to question relying on "gold standards". It is rarely the case that legal experts are asked to evaluate the proposed approaches afterward. AI&Law research could benefit from user studies. Put more drastically, "legal-expert-centered" research could open up new, promising research dimensions to AI&Law research.

The pace of the technological development of the word embeddings technology is fast. The word2vec algorithm marks a milestone approach that led to a substantial scientific interest in word embeddings. A major success factor is the shallow neural network (one hidden layer) used for training that enables an efficient calculation of word embeddings models on large corpora. Facing currently a hype of deep learning technologies, the development of word embeddings at the time of writing experiences a trend to train word embeddings models (again) with deep neural networks. Hyped approaches include, for example, ELMo (Peters et al. (2017)) and BERT (Devlin et al. (2018), published as a pre-print on arXiv). However, only small improvements seem feasible at the moment by using deep learning concepts in NLP or even less good results than those obtained by simple approaches, see, for example, Pagliardini et al. (2018) where the accumulation of vectors works better than training deep learning algorithms for sentence representations. This gives rise to question the high effort of deep learning technologies for the small improvements possible so far. It would be sensible to investigate the effects of the overfitting of deep learning technologies in more depth in order to classify deep learning technologies on a more abstract level. Arguing with Occam's razor, Mikolov's approach might be a good trade-off of simplicity and complexity in comparison to deep learning concepts for training word embeddings models. However, word embeddings will be an exciting field of research to stay.

Appendix

A. Processing Environment

Hardware

All experiments where runtime is captured are conducted on a machine with the following specification:

- **Processor:** Intel(R) Xeon(R) CPU E5-2650, 16 x 2.00GHz
- **Main Memory:** 16 GB
- **Hard Disk Drive:** 200 GB

Software

- **Operating System:** Ubuntu 16.04.5 LTS (Xenial Xerus), Kernel version: 4.4.0-146-generic
- **word2vec:** version 0.1c, https://github.com/tmikolov/word2vec
- **GloVe:** version 0.2, http://github.com/stanfordnlp/glove
- **FastText:** version 0.1.0, https://fasttext.cc/
- **gensim:** version 3.7.1, https://radimrehurek.com/gensim/
- **spacy:** version 2.0.5, https://spacy.io/
- **pandas:** version 0.24.0, https://pandas.pydata.org/
- **bleach:** version 2.1.2, https://pypi.org/project/bleach/
- **flask:** version 0.12.2, http://flask.pocoo.org/

- **numpy:** version 1.16.1, https://www.numpy.org/
- **scipy:** version 1.2.0, https://www.scipy.org/
- **seaborn:** version 0.9.0, https://seaborn.pydata.org/
- **nltk:** version 3.2.5, https://www.nltk.org/
- **scikit-learn:** version 0.20.2, https://scikit-learn.org/stable/
- **nmslisb:** version 1.7.2, https://github.com/nmslib/nmslib

B. Hyper-parameter Configurations

The following tables describe the utilized hyper-parameter configurations of word embeddings algorithms. Default values are marked with an asterisk. Hyper-parameter ranges are indicated with dashes. Subsampling is denoted by brackets. For example, 1-59 (:2) means that the hyper-parameter range ranges from one to 59 and every second value is considered, i.e., $1, 3, 5, \ldots, 59$. All word embeddings models are calculated with 16 threads.

The following hyper-parameter configurations of the word2vec algorithm are used:

Configuration	*Model Architecture*	*Iterations*	*Min-Count*	*Window Size*	*Vector Size*	*Negative Samples*	*Sample Threshold*
W01	CBOW*	43	5*	5*	400	5*	1e-3*
W02	CBOW*	40	5*	5*	400	5*	1e-3*
W03	Skip-gram	59	5*	5*	400	5*	1e-3*
W04	CBOW*	59	5*	5*	400	5*	1e-3*
W05	CBOW*	59	5*	5*	400	5*	1e-3*
W06	CBOW*	59	5*	5*	400	5*	1e-3*
W07	Skip-gram	1-59 (:2)	1	5*	400	5*	1e-3*
W08	CBOW*	1-59 (:2)	1	5*	400	5*	1e-3*
W09	Skip-gram	1-59 (:2)	5*	5*	400	5*	1e-3*
W10	CBOW*	1-59 (:2)	5*	5*	400	5*	1e-3*
W11	CBOW*	40	5*	1-30 (:1)	400	5*	1e-3*
W12	CBOW*	40	5*	5*	50-600 (:50)	5*	1e-3*
W13	CBOW*	40	5*	5*	400	1-10 (:1)	1e-3*
W14	CBOW*	40	5*	5*	400	5*	0 - 5e-3 (:1e-4)
W15	CBOW*	40	5*	8	150	8	5e-4
W16	CBOW*	43,41,39	5*	5*	400	5*	1e-3*
W17	CBOW*	43,41,39	1	5*	400	5*	1e-3*
W18	CBOW*	46-35(:2)	1	5	400	5*	1e-3*
W19	CBOW*	100	1	5*	300	5*	1e-3*
W20	CBOW*	100	5*	5*	300	5*	1e-3*

Table B1.: word2vec algorithm hyper-parameter configurations.

The Alpha hyper-parameter is not investigated and set to the default value.

The following hyper-parameter configurations of the GloVe algorithm are used:

Configuration	*Iterations*	*Min-Count*	*Window Size*	*Vector Size*
G01	1-59	1	5	400
G02	1-59	5	5	400
G03	40	5	1-30 (:1)	400
G04	40	5	5	50-600 (:50)
G05	40	5	23	200

Table B2.: GloVe algorithm hyper-parameter configurations.

The hyper-parameters Symmetric, Max-Vocab, Distance Weighting, Alpha, Eta, X-Max are not investigated and set to default values.

The following hyper-parameter configurations of the FastText algorithm are used:

Configuration	*Model Architecture*	*Iterations*	*Min-Count*	*Window Size*	*Vector Size*	*Negative Samples*	*Sample Threshold*	*MinN*	*MaxN*
F01	Skip-gram	40	5*	5*	400	5*	1e-3*	3*	6*
F02	Skip-gram	1-59 (:4)	5*	5*	400	5*	1e-3*	3*	6*
F03	CBOW	1-59 (:4)	5*	5*	400	5*	1e-3*	3*	6*
F04	Skip-gram	5*	1-30 (:1)	5*	400	5*	1e-3*	3*	6*
F05	Skip-gram	5*	5*	5*	50-600 (:50)	5*	1e-3*	3*	6*
F06	Skip-gram	5*	5*	5*	400	1-9	1e-3*	3*	6*
F07	Skip-gram	5*	5*	5*	400	5*	0 - 5e-3 (:1e-4)	3*	6*
F08	Skip-gram	8	5*	5*	400	5*	1e-3*	1-3	1-6
F09	Skip-gram	60	5*	5*	400	5*	1e-3*	3*	6*
F10	Skip-gram	100	5*	5*	400	5*	1e-3*	3*	6*
F11	Skip-gram	100	5*	5*	400	5*	1e-3*	3*	6*

Table B3.: FastText algorithm hyper-parameter configurations.

The LrUpdateRate hyper-parameter is not investigated and set to default value.

C. User Study Questions

The following questions are used in the online questionnaire of the user study. For each question, the answer options are listed in angle brackets and English translations are provided.

Q01 Wie viele Jahre sind Sie bereits als Rechtsanwalt bzw. Rechtsanwältin tätig? / How many years of professional experience as a lawyer do you have? <0-3 Jahre / 0-3 years; 4-6 Jahre / 4-6 years; 7-10 Jahre / 7-10 years; Mehr als 10 Jahre / More than 10 years>

Q02 Sind Sie Berufsträger bzw. Berufsträgerin? / Are you an attorney at law? <Ja / Yes; Nein / No>

Q03 Wie gut kennen Sie sich im deutschen Mietrecht aus? / How well do you know German tenancy law? <Grundkenntnisse / basic knowledge; Gelegentliche Beratung / Seldom consultation; Häufige Beratung / Frequent consultation; Schwerpunkt meiner Beratung / Focus of consultation>

Q04 In welchen Rechtsgebieten sind Sie hauptsächlich tätig? Bitte wählen Sie bitte maximal 3 aus. / <Arbeitsrecht / Labor law; Bank-und Kapitalmarktrecht / Banking law; Bau- und Architektenrecht / Building law; Erbrecht / Inheritance law; Familienrecht / Family law; Rechtsschutz / Legal protection; Handels- und Gesellschaftsrecht / Industrial property protection; IT-Recht / IT law; Insolvenzrecht / Insolvency law; Internationales Recht / International Law; Medizinrecht / Medical law; Miet- und Wohnungseigentumsrecht / Tenancy Law; Migrationsrecht / Migration law; Sozialrecht / Social law; Steuerrecht / Tax law; Strafrecht / Criminal law; Transport- und Speditionsrecht / Transport law; Urheber- und Medienrecht / Copyright and entertainment law; Vergaberecht / Public procurement law; Verkehrsrecht / Traffic law; Versicherungsrecht / Insurance law; Verwaltungsrecht / Administration law; Andere / Others>

Q05 Wenn Sie sich ein typisches, häufiges Mandat Ihrer Kanzlei vorstellen: Wie hoch ist der Anteil der Recherche von Fachinformationen (Kommentare und Handbücher, Rechtsprechung, Gesetzte, Muster etc.) im Durchschnitt an Ihrer Arbeitszeit? / If you consider a typical brief in your chancery: How large is the fraction of legal research (Legal comments and books, court decisions, legislation, patterns? <Weniger als 1% / Less than 1%; 1-5%; 5-10%; 10-20%; 20-30%; 30-40%; 40-50%; Mehr als 50% / More than 50%>

Q06 Mit welchem Fachinformationsmittel (Medium) beginnen Sie in der Regel Ihre Recherche? / What legal resource is the common entry point of your legal research? <Buch oder Zeitschrift / Book or journal; Bezahlte Fachwissendatenbank / Paid online service; Internetsuche / Free online service; Andere, und zwar / Others, namely>

Q07 Welche Fachinformationsmittel (Medien) nutzen Sie regelmäßig im Laufe einer Recherche? (Mehrfachnennungen möglich) / What legal resources (mediums) do you use regularly during a legal research? (Multiple choices possible) <Buch oder Zeitschrift / Book or journal; Bezahlte Fachwissendatenbank / Paid online service; Internetsuche / Free online search; Andere, und zwar / Others, namely>

Q08 Ich denke, dass ich die Kontext-bezogene Suche gerne häufig benutzen würde. / I think that I would like to use the Selection Search frequently. <Stimme gar nicht zu / Strongly disagree; Stimme zu / Agree; Neutral / Neutral; Stimme zu / Agree; Stimme stark zu / Strongly agree>

Q09 Ich fand die kontext-basierte Suche unnötig komplex. / I found the Selection Search unnecessarily

complex. <Stimme gar nicht zu / Strongly disagree; Stimme zu / Agree; Neutral / Neutral; Stimme zu / Agree; Stimme stark zu / Strongly agree>

Q10 Ich fand die Kontext-bezogene Suche einfach zu benutzen. / I thought the Selection Search was easy to use. <Stimme gar nicht zu / Strongly disagree; Stimme zu / Agree; Neutral / Neutral; Stimme zu / Agree; Stimme stark zu / Strongly agree>

Q11 Ich glaube, ich würde die Hilfe einer technisch versierten Person benötigen, um die Kontext-bezogene Suche benutzen zu können. / I think that I would need the support of a technical person to be able to use the Selection Search. <Stimme gar nicht zu / Strongly disagree; Stimme zu / Agree; Neutral / Neutral; Stimme zu / Agree; Stimme stark zu / Strongly agree>

Q12 Ich fand, die verschiedenen Funktionen in der Kontext-bezogenen Suche waren gut integriert. / I found the various functions in the Selection Search were well integrated. <Stimme gar nicht zu / Strongly disagree; Stimme zu / Agree; Neutral / Neutral; Stimme zu / Agree; Stimme stark zu / Strongly agree>

Q13 Ich denke, die Kontext-bezogene Suche enthielt zu viele Inkonsistenzen. / I thought there was too much inconsistency in the Selection Search. <Stimme gar nicht zu / Strongly disagree; Stimme zu / Agree; Neutral / Neutral; Stimme zu / Agree; Stimme stark zu / Strongly agree>

Q14 Ich kann mir vorstellen, dass die meisten Menschen den Umgang mit der Kontext-bezogenen Suche sehr schnell lernen. / I imagine that most people would learn to use the Selection Search very quickly. <Stimme gar nicht zu / Strongly disagree; Stimme zu / Agree; Neutral / Neutral; Stimme zu / Agree; Stimme stark zu / Strongly agree>

Q15 Ich fand die Kontext-bezogene Suche sehr umständlich zu nutzen. / I found the Selection Search very awkward to use. <Stimme gar nicht zu / Strongly disagree; Stimme zu / Agree; Neutral / Neutral; Stimme zu / Agree; Stimme stark zu / Strongly agree>

Q16 Ich fühlte mich bei der Benutzung der Kontext-bezogenen Suche sehr sicher. / I felt very confident using the Selection Search. <Stimme gar nicht zu / Strongly disagree; Stimme zu / Agree; Neutral / Neutral; Stimme zu / Agree; Stimme stark zu / Strongly agree>

Q17 Ich musste eine Menge lernen, bevor ich anfangen konnte die Kontext-bezogene Suche zu verwenden. / I needed to learn a lot of things before I could get going with the Selection Search. <Stimme gar nicht zu / Strongly disagree; Stimme zu / Agree; Neutral / Neutral; Stimme zu / Agree; Stimme stark zu / Strongly agree>

Q18 Insgesamt, würde ich die Benutzerfreundlichkeit der Kontext-bezogenen Suche bewerten als: / Overall, I would rate the user-friendliness of the Keyword Search as: <Schlimmstmöglich / Worst imaginable; Schrecklich / Awful; Schlecht / Poor; Ausreichend / OK; Gut / Good; Sehr gut / Excellent; Bestmöglich / Best imaginable>

Q19 Ich denke, dass ich die Volltext-Suche gerne häufig benutzen würde. / I think that I would like to use the Keyword Search frequently. <Stimme gar nicht zu / Strongly disagree; Stimme zu / Agree; Neutral / Neutral; Stimme zu / Agree; Stimme stark zu / Strongly agree>

Q20 Ich fand die Volltext-Suche unnötig komplex. / I found the Keyword Search unnecessarily com-

plex. <Stimme gar nicht zu / Strongly disagree; Stimme zu / Agree; Neutral / Neutral; Stimme zu / Agree; Stimme stark zu / Strongly agree>

Q21 Ich fand die Volltext-Suche einfach zu benutzen. / I thought the Keyword Search was easy to use. <Stimme gar nicht zu / Strongly disagree; Stimme zu / Agree; Neutral / Neutral; Stimme zu / Agree; Stimme stark zu / Strongly agree>

Q22 Ich glaube, ich würde die Hilfe einer technisch versierten Person benötigen, um die Volltext-Suche benutzen zu können. / I think that I would need the support of a technical person to be able to use the Keyword Search. <Stimme gar nicht zu / Strongly disagree; Stimme zu / Agree; Neutral / Neutral; Stimme zu / Agree; Stimme stark zu / Strongly agree>

Q23 Ich fand, die verschiedenen Funktionen in der Volltext-Suche waren gut integriert. / I found the various functions in the Keyword Search were well integrated. <Stimme gar nicht zu / Strongly disagree; Stimme zu / Agree; Neutral / Neutral; Stimme zu / Agree; Stimme stark zu / Strongly agree>

Q24 Ich denke, die Volltext-Suche enthielt zu viele Inkonsistenzen. / I thought there was too much inconsistency in the Keyword Search. <Stimme gar nicht zu / Strongly disagree; Stimme zu / Agree; Neutral / Neutral; Stimme zu / Agree; Stimme stark zu / Strongly agree>

Q25 Ich kann mir vorstellen, dass die meisten Menschen den Umgang mit der Volltext-Suche sehr schnell lernen. / I imagine that most people would learn to use the Keyword Search very quickly. <Stimme gar nicht zu / Strongly disagree; Stimme zu / Agree; Neutral / Neutral; Stimme zu / Agree; Stimme stark zu / Strongly agree>

Q26 Ich fand die Volltext-Suche sehr umständlich zu nutzen. / I found the Keyword Search very awkward to use. <Stimme gar nicht zu / Strongly disagree; Stimme zu / Agree; Neutral / Neutral; Stimme zu / Agree; Stimme stark zu / Strongly agree>

Q27 Ich fühlte mich bei der Benutzung der Volltext-Suche sehr sicher. / I felt very confident using the Keyword Search. <Stimme gar nicht zu / Strongly disagree; Stimme zu / Agree; Neutral / Neutral; Stimme zu / Agree; Stimme stark zu / Strongly agree>

Q28 Ich musste eine Menge lernen, bevor ich anfangen konnte die Volltext-Suche zu verwenden. / I needed to learn a lot of things before I could get going with the Keyword Search. <Stimme gar nicht zu / Strongly disagree; Stimme zu / Agree; Neutral / Neutral; Stimme zu / Agree; Stimme stark zu / Strongly agree>

Q29 Insgesamt, würde ich die Benutzerfreundlichkeit der Volltext-Suche bewerten als: / Overall, I would rate the user-friendliness of Keyword Search as: <Schlimmstmöglich / Worst imaginable; Schrecklich / Awful; Schlecht / Poor; Ausreichend / OK; Gut / Good; Sehr gut / Excellent; Bestmöglich / Best imaginable>

Q30 Wie beurteilen Sie die Qualität der Ergebnisse von "Methode 1" im Durchschnitt? / How do you rate the quality of the results of "Method 1" on average? <Mangelhaft / E; Ausreichend / D; Befriedigend / C; Gut / B; Sehr gut / A>

Q31 Wie beurteilen Sie die Qualität der Ergebnisse von "Methode 2" im Durchschnitt? / How do you

rate the quality of the results of "Method 2" on average? <Mangelhaft / E; Ausreichend / D; Befriedigend / C; Gut / B; Sehr gut / A>

Q32 Wie beurteilen Sie die Qualität der Ergebnisse von "Methode 3" im Durchschnitt? / How do you rate the quality of the results of "Method 3" on average? <Mangelhaft / E; Ausreichend / D; Befriedigend / C; Gut / B; Sehr gut / A>

Q33 Wie beurteilen Sie die Qualität der Ergebnisse der "Volltext-Suche" im Durchschnitt? / How do you rate the quality of the results of the Keyword Search on average? <Mangelhaft / E; Ausreichend / D; Befriedigend / C; Gut / B; Sehr gut / A>

Q34 Welche Suchmethode hat im Rahmen dieser Aufgabenstellung aus Ihrer Sicht die besten Suchergebnisse geliefert? / In your opinion, what search method delivered the best results? <Methode 1 / Method 1; Methode 2 / Method 2; Methode 3 / Method 3; Volltext-Suche / Keyword Search>

Q35 Was ist für Sie persönlich der wichtigste Grund, der für eine der Suchmethoden spricht? / What is the most important factor of a search method? <Qualität der Ergebnisse / Quality of results; Bedienbarkeit / Usability; Steuerbarkeit/Kontrolle / Controlability; Anderer Grund, und zwar / Other reason, namely>

Q36 Fallen Ihnen bei "Methode 1" Besonderheiten bei der Qualität der Suchergebnisse auf, z. B. dass lange Text-Passagen tendenziell zu schlechteren Ergebnissen führen oder Ähnliches? / Do you recognize patterns regarding the quality of search results, for example, that long text segments lead to worse results or similar for "Method 1"? <Freitext-Kommentar / Free-text comment>

Q37 Fallen Ihnen bei "Methode 2" Besonderheiten bei der Qualität der Suchergebnisse auf, z. B. dass lange Text-Passagen tendenziell zu schlechteren Ergebnissen führen oder Ähnliches? / Do you recognize patterns regarding the quality of search results, for example, that long text segments lead to worse results or similar for "Method 2"? <Freitext-Kommentar / Free-text comment>

Q38 Fallen Ihnen bei "Methode 3" Besonderheiten bei der Qualität der Suchergebnisse auf, z. B. dass lange Text-Passagen tendenziell zu schlechteren Ergebnissen führen oder Ähnliches? / Do you recognize patterns regarding the quality of search results, for example, that long text segments lead to worse results or similar for "Method 3"? <Freitext-Kommentar / Free-text comment>

Q39 Fallen Ihnen bei der "Volltext-Suche" Besonderheiten bei der Qualität der Suchergebnisse auf, z. B. dass lange Text-Passagen tendenziell zu schlechteren Ergebnissen führen oder Ähnliches? / Do you recognize patterns regarding the quality of the search results, for example, that long text segments lead to worse results or similar for the Keyword Search? <Freitext-Kommentar / Free-text comment>

Q40 Haben Sie Kritik/Anregung/Verbesserungsvorschläge? / Do you have criticism/suggestions/proposals for improvement? <Freitext-Kommentar / Free-text comment>

Q41 Würden Sie eine Kontext-bezogene Suche in Zukunft nutzen wollen? / Would you like to use the Selection Search in the future? <Ja / Yes; Nein / No>

D. Example Candidate Term Lists

a) JoBimText Approach compared to Single Word Approach with word2vec

Existing synset terms	Rank	JoBimText Approach		Single Word Approach (word2vec)	
		Candidate terms	⌀ Rating	Candidate terms	⌀ Rating
steuerfestsetzungsverfahrens (taxing procedure's)	1	verfahren (procedure)	0,5	festsetzungsverfahrens (fixation procedure's)	0,5
steuerfestsetzungsverfahren (taxing procedure)	2	betriebsaufspaltung (cleavage of company)	0	veranlagungsverfahrens (investment procedure's)	0,5
	3	vergleich (settlement)	0	feststellungsverfahrens (finding procedure's)	0,5
	4	prüfung (examination)	0	verwaltungsverfahrens (administration procedure's)	0,5
	5	betriebsverpachtung (lease of company)	0	billigkeitsverfahrens (justness procedure)	0,5
	6	außenprüfung (external examination)	0	lohnsteuerabzugsverfahrens (income tax deduction procedure's)	0,5
	7	überprüfung (audit)	0	steuererhebungsverfahrens (tax collection procedure's)	1
	8	auseinandersetzung (dispute)	0	abrechnungsverfahrens (calculation procedure's)	0,5
	9	berechnung (calculation)	0,75	rechtsbehelfsverfahrens (relief procedure's)	0,5
	10	anschaffung (purchase)	0	steuerschuldverhältnisses (tax liability rate's)	0,5

Existing synset terms	Rank	JoBimText Approach		Single Word Approach (word2vec)	
		Candidate terms	⌀ Rating	Candidate terms	⌀ Rating
lohnberechnung (wage calculation)	1	abrechnung (accounting)	0,5	lohnabrechnungen (wage accounting)	1
vergütungsberechnung (payment calculation)	2	berechnung (calculation)	0,75	teilzeitkräfte (part-time workers)	0
entgeltberechnung (salary calculation)	3			stundennachweise (work performance records)	0,25
bezügeberechnung (earnings calculation)	4			branchenbesonderheiten (sector properties)	0
	5			einzelbezahlung (single pay)	0
	6			schichtzuschläge (shift allowances)	0,5
	7			interviewtätigkeit (interview activity)	0
	8			verrichtet (performed)	0
	9			trainingsprogramme (training programs)	0
	10			stundenlöhne (hourly wages)	0,5

b) Single Word Approach with word2vec compared to Synset Vector Approach with word2vec

Existing synset terms	Rank	Single Word Approach (word2vec)		Synset Vector Approach (word2vec)	
		Candidate terms	⌀ Rating	Candidate terms	⌀ Rating
firmen-pkw (company auto)	1	fahrzeug (car)	0,75	fahrzeug (car)	0,75
firmenwagen (company wagon)	2	einsatzfahrzeug (utility car)	0,75	dienst-pkw (office auto)	1
betriebs-pkw (enterprise auto)	3	fahrer (driver)	0,5	firmenwagens (company wagon's)	1
betriebsfahrzeug (enterprise car)	4	privatflugzeug (private plain)	0,5	firmenfahrzeuge (enterprise cars)	1
dienstfahrzeug (office car)	5	nutzungsverbot (ban on the use)	0	dienstwagens (office wagon)	1
dienstwagen (office wagon)	6	dienstfahrten (office tour)	0,5	privatfahrten (private tour)	0,5
"+7"	7	flugzeug (airplane)	0	dienstkraftwagen (office automobile)	1
	8	dienstkraftwagen (office automobile)	1	firmenfahrzeuges (company car's)	1
	9	betriebshof (depot)	0	dienstfahrten (office tours)	0,5
	10	firmeneigenes (company's)	0,25	fahrzeuge (cars)	0,75

Existing synset terms	Rank	Single Word Approach (word2vec)		Synset Vector Approach (word2vec)	
		Candidate terms	⌀ Rating	Candidate terms	⌀ Rating
gasthaus (guest-house)	1	hotel (hotel)	0,25	restaurant (restaurant)	1
gaststätte (pub)	2	restaurant (restaurant)	1	hotel (hotel)	0,25
	3	privathaus (private home)	0,25	kantine (canteen)	0,5
	4	bordell (bordel)	0	gastwirtschaft (restaurant)	1
	5	zimmer (room)	0	apotheke (pharmacy)	0
	6	fachgeschäft (specialized store)	0	gaststätten (restaurants)	1
	7	sozialraum (social environment)	0	gasthof (inn)	1
	8	gästehaus (guest house)	1	nachtbar (neighbor)	0,25
	9	bauernhaus (farm house)	0,25	restaurants (restaurants)	1
	10	gasthof (inn)	1	tankstelle (gas station)	0

Figure D1.: Example candidate terms for the JoBimText, Single Word and Synset Vector Approaches.

a) Synset Vector Approach with GloVe, word2vec and FastText word embeddings algorithms

Existing synset terms	Rank	Synset Vector Approach (GloVe)		Synset Vector Approach (word2vec)		Synset Vector Approach (FastText)	
		Candidate terms	⌀ Rating	Candidate terms	⌀ Rating	Candidate terms	⌀ Rating
informationsschrift (document for information)	1	seitige (-sided)	0,25	rundschreiben (circular letter)	1	informationsbroschüren (brochures for information)	1
informationsbrief (letter for information)	2	gereichte (brought about)	0	broschüre (brochure)	1	informationsblatts (page for information's)	1
informationsblatt (page for information)	3	est-kommission (income tax comission)	0	informationsschreiben (information letter)	1	informationsblätter (pages for information)	1
informationsbroschüre (brochure for information)	4	rechtsbehelfsheftung (rights of legal appeal)	0,75	merkblatt (handout)	1	informationsschreibens (pages for informations')	1
	5	ab-nr (ab-nr.)	0	merkblatts (handout's)	1	informationsschreiben (paper for information)	1
	6	urteilsabschrift (transcript of court decision)	0,5	anschreiben (cover letter)	0,75	informationsschriften (documents for information)	1
	7	zustvst ("tax authory law")	0	hausmitteilung (internal memo)	0,5	informationsheftes (booklet for information)	1
	8	herausgegebene (released)	0,25	bericht (report)	0,75	broschüre (brochure)	1
	9	einspruchshefter (file of appeal)	0,25	verkaufsprospekt (sales brochure)	0,5	informationsstelle (information office)	0,5
	10	ötbmf ("tax authority in Austria")	0	bestätigungsschreiben (confirmationn letter)	0,5	informationsprogramm (information program)	0,5

Existing synset terms	Rank	Synset Vector Approach (GloVe)		Synset Vector Approach (word2vec)		Synset Vector Approach (FastText)	
		Candidate terms	⌀ Rating	Candidate terms	⌀ Rating	Candidate terms	⌀ Rating
nachschusspflicht (reserve liability)	1	nachschußpflicht (reserve liability)	1	nachschußpflicht (reserve liability)	1	nachschusspflichten (reserve liabilities)	1
nachschußverpflichtung (duty of reserve liability)	2	nachschussverpflichtung (duty of reserve liability)	1	nachschussverpflichtung (duty of reserve liability)	1	nachschusspflichtig (liable to reserve)	1
	3	erwerbspflicht (duty of income)	0	außenhaftung (external liability)	0	nachschussverpflichtung (duty of reserve liability)	1
	4	haftungsfreistellung (indemnification)	0	verlustteilnahme (participation of loss)	0,25	nachschussverpflichtungen (duties of reserve liability)	1
	5	gesellschaftsvertragliche (company agreement's)	0,25	pflichteinlage (obligatory deposit)	0,5	nachschuss (reserve)	0,5
	6	reelle (real)	0,25	einlageverpflichtung (duty of obligatory deposit)		nachschußpflicht (reserve liability)	1
	7	verschuldensunabhängige (independent of debt)	0,25	verlustbeteiligung (loss sharing)	0,25	nachschußpflichten (reserve liabilities)	1
	8	empfangsbevollmächtigung (confirmation of authorization to receive)	0	haftsumme (amount of liability)	0	nachschuß (reserve)	0,5
	9	abnahmeverpflichtung (obligation to take delivery)	0	kommanditeinlage (deposit for limited partnerships)	0,25	nachschusses (reserve's)	0,5
	10	haftungsbeschränkung (limitation of liability)	0	haftungsbeschränkung (limitation of liability)	0	zuschusspflicht (duty of grant)	0,5

b) Synset Vector Approach with word2vec compared to Intersection Approach with word2vec with different hyper-parameter configurations

Existing synset terms	Rank	Synset Vector Approach (word2vec)		Intersection Approach (word2vec), I=10, K=10		Intersection Approach (word2vec), I=10, K=20	
		Candidate terms	⌀ Rating	Candidate terms	⌀ Rating	Candidate terms	⌀ Rating
speditionsunternehmen (forwarding enterprise)	1	transportunternehmen (carrying business)	1	transportunternehmen (carrying business)	1	transportunternehmen (carrying business)	1
spedition (forwarding agent)	2	fabrik (factory)	0	fabrik (factory)	0	auslieferungslager (depot)	0,5
spediteur (haulier)	3	speditionen (forwardin agents)	1	speditionen (forwarding agents)	1	fabrik (factory)	0
speditionsfirma (forwarding company)	4	verkaufsbüro (sales office)	0	großhandel (wholesale)	0,25	speditionen (forwarding agents)	1
speditionsbetrieb (forwarding firm)	5	fuhrunternehmen (carrying companies)	1			frachtführer (carrier (person))	0,5
speditionsgeschäft (forwarding business)	6	großhandel (wholesale)	0,25			großhandel (wholesale)	0,25
	7	auslieferungslager (depot)	0,5			speditions (forwarding agent's)	1
	8	speditions (forwarding agent's)	1			kraftfahrzeugzubehör (automobile accessories)	0
	9	gießerei (casting company)	0				
	10	kraftfahrzeugzubehör (automobile accessories)	0				

Existing synset terms	Rank	Synset Vector Approach (word2vec)		Intersection Approach (word2vec), I=10, K=10		Intersection Approach (word2vec), I=10, K=20	
		Candidate terms	⌀ Rating	Candidate terms	⌀ Rating	Candidate terms	⌀ Rating
storno (reversal)	1	storniert (canceled)	1	storniert (canceled)	1	storniert (canceled)	1
stornieren (cancel)	2	stornierungen (cancellations)	1	stornierungen (cancellations)	1	stornierungen (cancellations)	1
stornierung (cancellation)	3	datenabsender (data transmitter)	0			eingegebene (entered)	0
	4	eingabedaten (input data)	0			auslesen (readout)	0
	5	buchungssatzes (accounting record's)	0			datenbestand (data stock)	0
	6	korrigieren (correct)	0,5			korrigieren (correct)	0,5
	7	übermittelter (information conveyed)	0				
	8	auslesen (readout)	0				
	9	eingegebene (entered)	0				
	10	manuell (manual)	0				

Figure D2.: Example candidate terms for word embeddings algorithms for the Synset Vector and Intersection Approaches.

Abbreviations

AI&Law Artificial Intelligence & Law

AP Average Precision

BOW Bag-of-Words

CBOW Continuous Bags of Words

DF Document Frequency

doc2vec Paragraph Vector

DSM Distributional Semantic Model

GCC German Civil Code

GDPR General Data Protection TODO

FN false negatives

FP false positives

FTSE Full Text Search Extension

NLP Natural Language Processing

GDPR General Data Protection Regulation

HAL Hyperspace Analogue to Language

IDF Inverse document frequency

LDA Latent Dirichlet Allocation

LMI Lexicographer's Mutual Information

LSI Latent Semantic Indexing

MAP Mean Average Precision

MI Mutual Information

PMI Point-wise Mutual Information

P-PMI Positive Point-wise Mutual Information

PCA Principal Component Analysis

POS part-of-speech

OPOSN only POS-nouns

PV-DBOW Distributed Bag Of Words Model of Paragraph Vectors

PV-DM Distributed Memory Model of Paragraph Vectors

RNN Recurrent Neural Network

RP-Score Ranking Position Score

SG Skip-gram

SP Standard pre-processing

SR Stopwords removal

ST Stemming

SUS System Usability Score

SVD Singular-Value Decomposition

TF-IDF Term Frequency - Inverse Document Frequency

TN true negatives

TP true positives

WE-DF Word Embeddings - Document Frequency

WE-IDF Word Embeddings - Inverse Document Frequency

XAI eXplainable AI

XSTM eXplainable Semantic Text Matching

Glossary

Candidate term list A candidate term list is a list of candidate terms that has been calculated by an algorithm. Typically candidate term lists are ordered and limited in size so that not all terms in a vocabulary are proposed as candidate terms for a synset.

Candidate term A candidate term is a term that has been identified by algorithms as a candidate for inclusion in a synset. Candidate terms usually need to be reviewed by human experts.

Distributional semantic model Distributional semantic models use statistical information to infer semantic aspects of texts and language elements. Baroni et al. (2014) distinguishes two classes of distributional semantic models. Count-based DSMs are traditional approaches that use static algorithms while predictive DSMs leverage machine learning algorithms and are capable of predicting words from a given sequence of text.

Keyword Search The Keyword Search is a traditional search method used in legal information retrieval systems.

Search technology In this thesis, a search technology means a technical approach for a (legal) information retrieval system that calculates search results for a user query.

Search method In this thesis, a search method is a human-computer interaction method, i.e., a way how users interact with a legal information retrieval system..

Selection Search The Selection Search is a "novel" human-computer interaction method where users select text fragments of an existing text as input to a (legal) information retrieval system.

Synset A synset is short for a synonym set, i.e., a set of terms that are considered as synonyms according to a thesaurus. Synsets are one type of sets that are collected in thesauri.

Synset embeddings A synset embeddings is dense, real-valued vector that analogue to word embeddings represents a synset..

Term A term is a semantic unit that can consist of several words or tokens, for example, one open compound word constitutes one term. Several terms can constitute a synset of a thesaurus.

Token A token are the technically derived semantically atomic unit from words through pre-processing.

Word A word means in this thesis a natural language word and is different than a term or a token.

Word embeddings model A word embeddings model encompasses all word embeddings vectors and unique set of tokens that are used to train a word embeddings model (vocabulary)..

Word embeddings model's vocabulary A word embeddings model's vocabulary is the set of unique tokens that are used to train a word embeddings model. Eventually, tokens are excluded from the training and the word embeddings model's vocabulary if they do not occur frequently enough. The minimum occurrence frequency of tokens in controlled via the Min-Count hyper-parameter of word embeddings algorithms.

Word embeddings vectors A word embeddings vector is a concrete vector that represents one token.

Word embeddings algorithm A word embeddings algorithm is a concrete algorithm or implementation of such an algorithm of the family of algorithms of word embeddings.

Word embeddings Word embeddings is a technology and a family of algorithms that represent tokens with dense, real-valued vectors.

Bibliography

Kolawole J. Adebayo, Luigi Di Caro, Livio Robaldo, and Guido Boella. Textual inference with tree-structured lstms. In *Proceedings of the 28th Benelux conference on Artificial Intelligence.*, 2016.

Laura Altamirano Sainz. Applying lexical knowledge to improve search quality for a german legal information database. Master's thesis, Technical University of Munich, Department of Informatics, Munich, Germany, 4 2015.

Ioannis Arapakis, Xiao Bai, and B Barla Cambazoglu. Impact of response latency on user behavior in web search. In *Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval*, pages 103–112. ACM, 2014.

Kevin D. Ashley. *Artificial Intelligence and Legal Analytics: New Tools for Law Practice in the Digital Age*. Cambridge University Press, Cambridge, 2017.

Hiteshwar Kumar Azad and Akshay Deepak. Query expansion techniques for information retrieval: a survey. *Information Processing & Management*, 56(5):1698–1735, 2019.

Aaron Bangor, Philip Kortum, and James Miller. Determining what individual sus scores mean: Adding an adjective rating scale. *J. Usability Studies*, 4(3):114–123, May 2009.

Marco Baroni and Alessandro Lenci. Distributional memory: A general framework for corpus-based semantics. *Computational Linguistics*, 36(4):673–721, 2010.

Marco Baroni, Georgiana Dinu, and Germán Kruszewski. Don't count, predict! a systematic comparison of context-counting vs. context-predicting semantic vectors. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 238–247, 2014.

Hannah Bast, Björn Buchhold, Elmar Haussmann, et al. Semantic search on text and knowledge bases. *Foundations and Trends in Information Retrieval*, 10(2-3):119–271, 2016.

Y Bengio, Olivier Delalleau, and Nicolas Le Roux. Label propagation and quadratic criterion. *Semi-Supervised Learning*, 09 2006.

Yoshua Bengio. Neural net language models. *Scholarpedia*, 3(1):3881, 2008.

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. A neural probabilistic language model. *Journal of machine learning research*, 3(Feb):1137–1155, 2003.

James Bergstra and Yoshua Bengio. Random search for hyper-parameter optimization. *J. Mach. Learn. Res.*, 13(1):281–305, February 2012.

Chris Biemann and Martin Riedl. Text: Now in 2d! a framework for lexical expansion with contextual similarity. *Journal of Language Modelling*, 1(1):55–95, 2013.

Jon Bing. Designing text retrieval systems for conceptual searching. In *Proceedings of the 1st international conference on Artificial intelligence and law*, pages 43–51. ACM, 1987.

David M Blei, Andrew Y Ng, and Michael I Jordan. Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan):993–1022, 2003.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. Enriching word vectors with subword information. *TACL*, 5:135–146, 2017.

J. Brooke. SUS: A quick and dirty usability scale. In P. W. Jordan, B. Weerdmeester, A. Thomas, and I. L. Mclelland, editors, *Usability evaluation in industry*. Taylor and Francis, London, 1996.

John A. Bullinaria and Joseph P. Levy. Extracting semantic representations from word co-occurrence statistics: A computational study. *Behavior Research Methods*, 39(3):510–526, Aug 2007.

John A Bullinaria and Joseph P Levy. Extracting semantic representations from word co-occurrence statistics: stop-lists, stemming, and svd. *Behavior research methods*, 44(3):890–907, 2012.

Frank Buschmann. *Pattern-orientierte Software-Architektur: Ein Pattern-System*. Pearson Deutschland GmbH, 1998.

Claudio Carpineto and Giovanni Romano. A survey of automatic query expansion in information retrieval. *Acm Computing Surveys (CSUR)*, 44(1):1, 2012.

Kenneth Ward Church and Patrick Hanks. Word association norms, mutual information, and lexicography. *Computational linguistics*, 16(1):22–29, 1990.

Jacob Cohen. A coefficient of agreement for nominal scales. *Educational and psychological measurement*, 20(1):37–46, 1960.

Ronan Collobert and Jason Weston. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th international conference on Machine learning*, pages 160–167. ACM, 2008.

Ido Dagan, Oren Glickman, and Bernardo Magnini. The pascal recognising textual entailment challenge. In *Machine Learning Challenges Workshop*, pages 177–190. Springer, 2005.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

Judith P Dick. Representation of legal text for conceptual retrieval. In *Proceedings of the 3rd international conference on Artificial intelligence and law*, pages 244–253. ACM, 1991.

Christian Dirschl. Thesaurus generation and usage at wolters kluwer deutschland gmbh. *Jusletter IT 18*, 2016.

Hai Dong, Farookh Khadeer Hussain, and Elizabeth Chang. A survey in semantic search technologies. In *2008 2nd IEEE international conference on digital ecosystems and technologies*, pages 403–408. IEEE, 2008.

John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12(Jul):2121–2159, 2011.

Christoph Erl. Semantic text matching of company policies and regulatory documents using text similarity measures. Master's thesis, Technical University of Munich, Department of Informatics, Munich, Germany, 2 2018.

Mohammad Hassan Falakmasir and Kevin D Ashley. Utilizing vector space models for identifying legal factors from text. In *JURIX*, volume 302 of *Frontiers in Artificial Intelligence and Applications*, pages 183–192. IOS Press, 2017.

Jérémy Ferrero, Laurent Besacier, Didier Schwab, and Frédéric Agnès. Using word embedding for cross-language plagiarism detection. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 415–421, 2017.

J. R. Firth. *A synopsis of linguistic theory 1930-55.*, volume 1952-59. The Philological Society, Oxford, 1957.

Debasis Ganguly, Dwaipayan Roy, Mandar Mitra, and Gareth J.F. Jones. Word embedding based generalized language model for information retrieval. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '15, pages 795–798, New York, NY, USA, 2015. ACM.

Yoav Goldberg. A primer on neural network models for natural language processing. *Journal of Artificial Intelligence Research*, 57:345–420, 2016.

Yoav Goldberg and Omer Levy. word2vec explained: deriving mikolov et al.'s negative-sampling word-embedding method. *arXiv preprint arXiv:1402.3722*, 2014.

Matthias Grabmair, Kevin D Ashley, Ran Chen, Preethi Sureshkumar, Chen Wang, Eric Nyberg, and Vern R Walker. Introducing luima: an experiment in legal conceptual retrieval of vaccine injury decisions using a uima type system and tools. In *Proceedings of the 15th International Conference on Artificial Intelligence and Law*, pages 69–78. ACM, 2015.

Gregory Grefenstette. *Explorations in automatic thesaurus discovery*, volume 278. Springer Science & Business Media, 1994.

Aditya Grover and Jure Leskovec. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 855–864. ACM, 2016.

Carole Diane Hafner. *An Information Retrieval System Based on a Computer Model of Legal Knowledge.* PhD thesis, University of Michigan, Ann Arbor, MI, USA, 1978.

Fritjof Haft and Eric Hilgendorf. *Strafrecht Besonderer Teil 1*, volume 9. Auflage. C.H. Beck, 2009.

Manfred Harner. Mietvertrag, eigentumswohnung, 2017. URL https://products2.haufe.de/#G:pi=PI17574&&;;D:did=HI1949297&&.

Zellig S Harris. Distributional structure. *Word*, 10(2-3):146–162, 1954.

Felix Hausdorff. Dimension und äußeres maß. *Mathematische Annalen*, 79(1):157–179, Mar 1918.

Djoerd Hiemstra. *Using Language Models for Information Retrieval*. PhD thesis, University of Twente, Enschede, Netherlands, 2001.

G. E. Hinton, J. L. McClelland, and D. E. Rumelhart. Parallel distributed processing: Explorations in the microstructure of cognition, vol. 1. In David E. Rumelhart, James L. McClelland, and CORPORATE PDP Research Group, editors, *Parallel distributed processing: explorations in the microstructure of cognition, vol. 1*, chapter Distributed Representations, pages 77–109. MIT Press, Cambridge, MA, USA, 1986a.

Geoffrey E Hinton, JL McClelland, and DE Rumelhart. Parallel distributed processing: Explorations in the microstructure of cognition, vol. 1. *Distributed representations*, pages 77–109, 1986b.

Geoffrey E Hinton et al. Learning distributed representations of concepts. In *Proceedings of the eighth annual conference of the cognitive science society*, volume 1, page 12. Amherst, MA, 1986c.

Perry R Hinton. *Guide for Social Science Students*. Routledge, 1995.

Thomas Hofmann. Probabilistic latent semantic analysis. In *Proceedings of the Fifteenth conference on Uncertainty in artificial intelligence*, pages 289–296. Morgan Kaufmann Publishers Inc., 1999.

Shoaib Jameel and Steven Schockaert. Entity embeddings with conceptual subspaces as a basis for plausible reasoning. In *Proceedings of the Twenty-second European Conference on Artificial Intelligence*, pages 1353–1361. IOS Press, 2016.

K. Sparck Jones. *Synonymy and Semantic Classification*. PhD thesis, University of Cambridge, Cambridge, United Kingdom, 1964.

Armand Joulin, Edouard Grave, Piotr Bojanowski, Matthijs Douze, Hérve Jégou, and Tomas Mikolov. Fasttext.zip: Compressing text classification models, 2016.

Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. Bag of tricks for efficient text classification. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 427–431, Valencia, Spain, 2017. Association for Computational Linguistics.

Edilson Anselmo Corrêa Júnior, Vanessa Queiroz Marinho, and Leandro Borges dos Santos. Nilc-usp at semeval-2017 task 4: A multi-view ensemble for twitter sentiment analysis. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 611–615, 2017.

Adam Kilgarriff, Pavel Rychly, Pavel Smrz, and David Tugwell. The sketch engine. *Information Technology*, 105:116, 2004.

Harald Kinne, Klaus Schach, and Hans-Jürgen Bieber. *Miet- und Mietprozessrecht*, volume 7. Auflage. Haufe Lexware, 2012.

Ryan Kiros, Yukun Zhu, Ruslan R Salakhutdinov, Richard Zemel, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. Skip-thought vectors. In *Advances in neural information processing systems*, pages 3294–3302, 2015.

Matt Kusner, Yu Sun, Nicholas Kolkin, and Kilian Weinberger. From word embeddings to document distances. In *International conference on machine learning*, pages 957–966, 2015.

J Richard Landis and Gary G Koch. The measurement of observer agreement for categorical data. *biometrics*, pages 159–174, 1977.

Jörg Landthaler. Image object and document classification using neural networks with replicated softmax input layers. Master's thesis, Technical University of Munich, Munich, Germany, 12 2011.

Jörg Landthaler, Bernhard Waltl, and Florian Matthes. Unveiling references in legal texts - implicit versus explicit network structures. In Erich Schweighofer, Franz Kummer, Walter Hötzendorfer, and Georg Borges, editors, *Networks, Proceedings of the 19th International Legal Informatics Symposium IRIS 2016*, volume 19, pages 71–78. Österreichische Computer Gesellschaft, 2015.

Jörg Landthaler, Bernhard Waltl, Patrick Holl, and Florian Matthes. Extending full text search for legal document collections using word embeddings. In *Legal Knowledge and Information Systems - JURIX 2016: The Twenty-Ninth Annual Conference*, volume 294 of *Frontiers in Artificial Intelligence and Applications*, pages 73–82. IOS Press, 2016.

Jörg Landthaler, Ingo Glaser, and Florian Matthes. Towards explainable semantic text matching. In *Legal Knowledge and Information Systems - JURIX 2018: The Thirty-first Annual Conference, Groningen, The Netherlands, 12-14 December 2018.*, volume 313 of *Frontiers in Artificial Intelligence and Applications*, pages 200–204. IOS Press, 2018a.

Jörg Landthaler, Elena Scepankova, Ingo Glaser, Hans Lecker, and Florian Matthes. Semantic text matching of contract clauses and legal comments in tenancy law. In Erich Schweighofer, Franz Kummer, Ahti Saarenpää, and Burkhard Schafer, editors, *Data protection / LegalTech, Proceedings of the 21st International Legal Informatics Symposium IRIS 2018*, volume 21, pages 73–82. Editions Weblaw, 2018b.

Jörg Landthaler, Bernhard Waltl, Dominik Huth, Daniel Braun, Christoph Stocker, Thomas Geiger, and Florian Matthes. Extending thesauri using word embeddings and the intersection method. In *Proceedings of the Second Workshop on Automated Semantic Analysis of Information in Legal Texts co-located with the 16th International Conference on Artificial Intelligence and Law (ICAIL 2017), London, UK*, volume 2143 of *CEUR Workshop Proceedings*. IOS Press, 2018c.

Jörg Landthaler, Ingo Glaser, Hans Lecker, and Florian Matthes. User study on selection search and semantic text matching in german tenancy law. In Erich Schweighofer, Franz Kummer, and Ahti Saarenpää, editors, *Internet of Things, Proceedings of the 22nd International Legal Informatics Symposium IRIS 2019*, volume 22, pages 367–368. Editions Weblaw, 2019.

Karl Larenz and Claus-Wilhelm Canaris. *Methodenlehre der Rechtswissenschaft*, volume 3. Auflage. Springer, 1995.

Steven A Lastres. Rebooting legal research in a digital age, 2015.

Quoc Le and Tomáš Mikolov. Distributed representations of sentences and documents. In *International conference on machine learning*, pages 1188–1196, 2014.

Omer Levy and Yoav Goldberg. Neural word embedding as implicit matrix factorization. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 2177–2185. Curran Associates, Inc., 2014.

Kevin Lund. Semantic and associative priming in high-dimensional semantic space. In *Proc. of the 17th Annual conferences of the Cognitive Science Society, 1995*, 1995.

Kevin Lund and Curt Burgess. Producing high-dimensional semantic spaces from lexical co-occurrence. *Behavior research methods, instruments, & computers*, 28(2):203–208, 1996.

B. B. Mandelbrot. An introduction to multifractal distribution functions. In H. Eugene Stanley and Nicole Ostrowsky, editors, *Random Fluctuations and Pattern Growth: Experiments and Models*, pages 279–291. Springer Netherlands, Dordrecht, 1988.

Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. *Introduction to Information Retrieval*. Cambridge University Press, New York, NY, USA, 2008.

Robert Meusel, Mathias Niepert, Kai Eckert, and Heiner Stuckenschmidt. Thesaurus extension using web search engines. In *International Conference on Asian Digital Libraries*, pages 198–207. Springer, 2010.

Tomáš Mikolov. *Statistical language models based on neural networks*. PhD thesis, Brno University of Technology, Brno, Sweden, 2012.

Tomáš Mikolov and Geoffrey Zweig. Context dependent recurrent neural network language model. In *2012 IEEE Spoken Language Technology Workshop (SLT)*, pages 234–239. IEEE, 2012.

Tomáš Mikolov, Martin Karafiát, Lukáš Burget, Jan Černockỳ, and Sanjeev Khudanpur. Recurrent neural network based language model. In *Eleventh annual conference of the international speech communication association*, 2010.

Tomáš Mikolov, Stefan Kombrink, Lukáš Burget, Jan Černockỳ, and Sanjeev Khudanpur. Extensions of recurrent neural network language model. In *2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5528–5531. IEEE, 2011.

Tomáš Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. In *Proceedings of the International Conference on Learning Representations (ICLR) Workshop*, ICLR'13, 2013a.

Tomáš Mikolov, Quoc V. Le, and Ilya Sutskever. Exploiting similarities among languages for machine translation. *CoRR*, 2013b.

Tomáš Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pages 3111–3119. Curran Associates, Inc., 2013c.

Tomáš Mikolov, Wen-tau Yih, and Geoffrey Zweig. Linguistic regularities in continuous space word representations. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 746–751, 2013d.

Scott Miller, Jethran Guinness, and Alex Zamanian. Name tagging with word clusters and discriminative training. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics: HLT-NAACL 2004*, 2004.

Andriy Mnih and Geoffrey E Hinton. A scalable hierarchical distributed language model. In *Advances in neural information processing systems*, pages 1081–1088, 2009.

Markus Mueller. Label propagation for tax law thesaurus extension. Master's thesis, Technical University of Munich, Department of Informatics, Munich, Germany, 11 2018.

Nona Naderi and Graeme Hirst. Argumentation mining in parliamentary discourse. In *Proceedings of the Computational Models of Natural Argument 2016, New York, United States*, 2016.

El Moatez Billah Nagoudi, Jérémy Ferrero, and Didier Schwab. Lim-lig at semeval-2017 task1: Enhancing the semantic similarity for arabic sentences with vectors weighting. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 134–138, 2017.

Yoshiki Niwa and Yoshihiko Nitta. Co-occurrence vectors from corpora vs. distance vectors from dictionaries. In *Proceedings of the 15th conference on Computational linguistics-Volume 1*, pages 304–309. Association for Computational Linguistics, 1994.

Bruno M Nogueira, Maria F Moura, M da S Conrado, Rafael G Rossi, Ricardo M Marcacini, and Solange O Rezende. Winning some of the document preprocessing challenges in a text mining process. In *Embrapa Informática Agropecuária-Artigo em anais de congresso (ALICE)*, 2008.

Arvid Österlund, David Ödling, and Magnus Sahlgren. Factorization of latent variables in distributional semantic models. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 227–231, Lisbon, Portugal, September 2015. Association for Computational Linguistics.

Matteo Pagliardini, Prakhar Gupta, and Martin Jaggi. Unsupervised learning of sentence embeddings using compositional n-gram features. In Marilyn A. Walker, Heng Ji, and Amanda Stent, editors, *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers)*, pages 528–540. Association for Computational Linguistics, 2018.

Raquel Mochales Palau and Marie-Francine Moens. Argumentation mining: the detection, classification and structure of arguments in text. In *Proceedings of the 12th international conference on artificial intelligence and law*, pages 98–107. ACM, 2009.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. Glove: Global vectors for word representation. In Alessandro Moschitti, Bo Pang, and Walter Daelemans, editors, *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1532–1543. ACL, 2014.

Lee F Peoples. The death of the digest and the pitfalls of electronic research: what is the modern legal researcher to do. *Law Library Journal*, 97:661, 2005.

James W Perry, Kent Allen, and Madeline M Berry. Machine literature searching x. machine language; factors underlying its design and development. *American Documentation (pre-1986)*, 6(4):242, 1955.

Matthew E Peters, Waleed Ammar, Chandra Bhagavatula, and Russell Power. Semi-supervised sequence tagging with bidirectional language models. *arXiv preprint arXiv:1705.00108*, 2017.

Philipp Pickel. Prototypical implementation and assessment of relatedness search in laws, judgments and commentaries. Master's thesis, Technical University of Munich, Department of Informatics, Munich, Germany, 12 2016.

Jay Michael Ponte and W Bruce Croft. *A language modeling approach to information retrieval*. PhD thesis, University of Massachusetts at Amherst, 1998.

Helen C Purchase. *Experimental human-computer interaction: a practical guide with visual examples*. Cambridge University Press, 2012.

M. Atif Qureshi and Derek Greene. Eve: explainable vector based embedding technique using wikipedia. *Journal of Intelligent Information Systems*, Jun 2018.

Nitin Ramrakhiyani, Sachin Pawar, and Girish Palshikar. word2vec or jobimtext?: A comparison for lexical expansion of hindi words. In *Proceedings of the 7th Forum for Information Retrieval Evaluation*, pages 39–42. ACM, 2015.

Sujith Ravi and Qiming Diao. Large scale distributed semi-supervised learning using streaming approximation. In *Artificial Intelligence and Statistics*, pages 519–528, 2016.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. "why should I trust you?": Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016*, pages 1135–1144, 2016.

Martin Riedl and Chris Biemann. Scaling to large3 data: An efficient and effective method to compute distributional thesauri. In *Conference on Empirical Methods on Natural Language Processing*, pages 884–890, 2013.

Ruty Rinott, Lena Dankin, Carlos Alzate, Mitesh M Khapra, Ehud Aharoni, and Noam Slonim. Show me your evidence–an automatic method for context dependent evidence detection. In *Proceedings of the 2015 Conference on Empirical Methods in NLP (EMNLP), Lisbon, Portugal*, pages 17–21, 2015.

Stephen Robertson, Hugo Zaragoza, et al. The probabilistic relevance framework: Bm25 and beyond. *Foundations and Trends® in Information Retrieval*, 3(4):333–389, 2009.

Werner Romberg. *Vereinfachte Numerische Integration*. Det Kongelige Norske Videnskabers Selskab Forhandlinger, Trondheim, Norway, 1955. Band 28.

Sascha Rothe and Hinrich Schütze. Autoextend: Combining word embeddings with semantic resources. *Computational Linguistics*, 43(3):593–617, 2017.

Herbert Rubenstein and John B. Goodenough. Contextual correlates of synonymy. *Communications of the ACM*, 8(10):627–633, October 1965.

David E Rumelhart, Geoffrey E Hinton, Ronald J Williams, et al. Learning representations by back-propagating errors. *Cognitive modeling*, 5(3):1, 1988.

Pavel Rychlý and Adam Kilgarriff. An efficient algorithm for building a distributional thesaurus (and other sketch engine developments). In *Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions*, ACL '07, pages 41–44, Stroudsburg, PA, USA, 2007. Association for Computational Linguistics.

Ali Sadeghian, Laksshman Sundaram, D Wang, W Hamilton, Karl Branting, and Craig Pfeifer. Semantic edge labeling over legal citation graphs. In *Proceedings of the workshop on legal text, document, and corpus analytics (LTDCA-2016)*, pages 70–75, 2016.

Magnus Sahlgren. *The Word-Space Model*. PhD thesis, Stockholm University, Stockholm, Sweden, 2006.

Magnus Sahlgren. The distributional hypothesis. *Italian Journal of Disability Studies*, 20:33–53, 2008.

Ruslan Salakhutdinov and Geoffrey Hinton. Semantic hashing. *International Journal of Approximate Reasoning*, 50(7):969–978, 2009.

Gerard Salton and Christopher Buckley. Term-weighting approaches in automatic text retrieval. *Information processing & management*, 24(5):513–523, 1988.

Ferdinand de Saussure. Course in general linguistics (trans. wade baskin). *London: Fontana/Collins*, page 74, 1916.

Ferdinand de Saussure. Course in general linguistics, translation. *R. Harris, London: Duckworth*, 1983.

Jaromir Savelka, Huihui Xu, and Kevin D Ashley. Improving sentence retrieval from case law for statutory interpretation. In *Proceedings of the Seventeenth International Conference on Artificial Intelligence and Law*, pages 113–122. ACM, 2019.

Hinrich Schütze. Dimensions of meaning. In *Supercomputing'92: Proceedings of the 1992 ACM/IEEE Conference on Supercomputing*, pages 787–796. IEEE, 1992.

Hinrich Schütze. Word space. In *Advances in neural information processing systems*, pages 895–902, 1993.

Erich Schweighofer, Anton Geist, et al. Legal query expansion using ontologies and relevance feedback. In *LOAIT*, pages 149–160, 2007.

Aleksander Smywiński-Pohl, Karol Lasocki, Krzysztof Wróbel, and Marek Strzala. Automatic construction of a polish legal dictionary with mappings to extra-legal terms established via word embeddings. In *Proceedings of the Seventeenth International Conference on Artificial Intelligence and Law*, ICAIL '19, pages 234–238, New York, NY, USA, 2019. ACM.

Richard Socher, Eric H Huang, Jeffrey Pennin, Christopher D Manning, and Andrew Y Ng. Dynamic pooling and unfolding recursive autoencoders for paraphrase detection. In *Advances in neural information processing systems*, pages 801–809, 2011.

Charles Spearman. The proof and measurement of association between two things. *American journal of Psychology*, 15(1):72–101, 1904.

Rudolf Stürzer and Michael Koch. *Vermieterlexikon*, volume 15. Auflage. Haufe Lexware, 2017.

Rudolf Stürzer, Michael Koch, Birgit Noack, and Martina Westner. *Vermieter-Praxishandbuch*, volume 8. Auflage. Haufe Lexware, 2015.

Keet Sugathadasa, Buddhi Ayesha, Nisansa de Silva, Amal Shehan Perera, Vindula Jayawardana, Dimuthu Lakmal, and Madhavi Perera. Legal document retrieval using document vector embeddings and deep learning. In Kohei Arai, Supriya Kapoor, and Rahul Bhatia, editors, *Intelligent Computing*, pages 160–175, Cham, 2019. Springer International Publishing.

Tokunaga Takenobu, Fujii Atsushi, Sakurai Naoyuki, and Tanaka Hozumi. Extending a thesaurus by classifying words. *Automatic Information Extraction and Building of Lexical Semantic Resources for NLP Applications*, 1997.

Andrew Trask, Phil Michalak, and John Liu. sense2vec-a fast and accurate method for word sense disambiguation in neural word embeddings. *arXiv preprint arXiv:1511.06388*, 2015.

Joseph Turian, Lev Ratinov, and Yoshua Bengio. Word representations: a simple and general method for semi-supervised learning. In *Proceedings of the 48th annual meeting of the association for computational linguistics*, pages 384–394. Association for Computational Linguistics, 2010.

Peter D Turney and Patrick Pantel. From frequency to meaning: Vector space models of semantics. *Journal of artificial intelligence research*, 37:141–188, 2010.

Naohiko Uramoto. Positioning unknown words in a thesaurus by using information extracted from a corpus. In *Proceedings of the 16th conference on Computational linguistics-Volume 2*, pages 956–961. Association for Computational Linguistics, 1996.

Marc Van Opijnen and Cristiana Santos. On the concept of relevance in legal information retrieval. *Artificial Intelligence and Law*, 25(1):65–87, 2017.

Cornelis Joost Van Rijsbergen. Foundation of evaluation. *Journal of documentation*, 30(4):365–373, 1974.

Ngoc Phuoc An Vo, Caroline Privault, and Fabien Guillot. Experimenting word embeddings in assisting legal review. In *Proceedings of the 16th edition of the International Conference on Articial Intelligence and Law*, pages 189–198. ACM, 2017.

Hugo P de Vos. Methods for automatically generating a legal thesaurus. Master's thesis, Radboud University, Center for Language Studies, Nijmegen, Netherlands, 9 2017.

Bernhard Waltl and Roland Vogel. Explainable artificial intelligence – the new frontier in legal informatics. *Jusletter IT 22*, 2018.

Bernhard Waltl, Jörg Landthaler, and Florian Matthes. Differentiation and empirical analysis of reference types in legal documents. In *JURIX*, volume 294 of *Frontiers in Artificial Intelligence and Applications*, pages 211–214. IOS Press, 2016.

Bernhard Waltl, Jörg Landthaler, and Florian Matthes. Differentiation and empirical analysis of reference types in legal documents. *Jusletter IT 17*, 2017a.

Bernhard Waltl, Jörg Landthaler, Elena Scepankova, Florian Matthes, Ingo Glaser, Christoph Stocker, and Christian Schneider. Automated extraction of semantic information from german legal documents. In Erich Schweighofer, Franz Kummer, Walter Hötzendorfer, and Christoph Sorge, editors, *Trends and Communities of Legal Informatics, Proceedings of the 20th International Legal Informatics Symposium IRIS 2017*, volume 20, pages 73–82. Österreichische Computer Gesellschaft, 2017b.

Bernhard Waltl, Georg Bonczek, Elena Scepankova, and Florian Matthes. Semantic types of legal norms in german laws: Classification and analysis using local linear explanations. *Artificial Intelligence and Law*, 2018.

Bernhard Ernst Waltl. *Semantic Analysis and Computational Modeling of Legal Documents.* dissertation, Technical University of Munich, 2018.

Gineke Wiggers, Suzan Verberne, Gerrit-Jan Zwenne, et al. Exploration of intrinsic relevance judgments by legal professionals in information retrieval systems. *Anne Dirkson, Suzan Verberne, Gerard van Oortmerssen & Wessel Kraaij*, page 5, 2018.

Ludwig Wittgenstein. *Philosophical investigations. Philosophische Untersuchungen.* Macmillan, 1953.

Yuto Yamaguchi and Kohei Hayashi. When does label propagation fail? a view from a network generative model. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, IJCAI'17, pages 3224–3230. AAAI Press, 2017.

Yuto Yamaguchi, Christos Faloutsos, and Hiroyuki Kitagawa. Omni-prop: Seamless node classification on arbitrary label correlation. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015.

Yuto Yamaguchi, Christos Faloutsos, and Hiroyuki Kitagawa. Camlp: Confidence-aware modulated label propagation. In *Proceedings of the 2016 SIAM International Conference on Data Mining*, pages 513–521. SIAM, 2016.

Hamed Zamani and W. Bruce Croft. Embedding-based query language models. In *Proceedings of the 2016 ACM International Conference on the Theory of Information Retrieval*, ICTIR '16, pages 147–156, New York, NY, USA, 2016. ACM.

Dengyong Zhou, Olivier Bousquet, Thomas N Lal, Jason Weston, and Bernhard Schölkopf. Learning with local and global consistency. In *Advances in neural information processing systems*, pages 321–328, 2004.

Xiaojin Zhu and Zoubin Ghahramani. Learning from labeled and unlabeled data with label propagation. Technical report, Carnegie Mellon University, 2002.