FRIEDRICH-SCHILLER-UNIVERSITÄT JENA

Fakultät für Mathematik und Informatik

Bachelor of Science (B. Sc.)'s Thesis

Evaluation of Query Expansion Methods for Semantic Search over German Legal Norms

BY

Friedrich Tydecks

Born 09.10.1999 in Stuttgart

Supervisors:

Prof. Dr. Birgitta König-Ries

Marianne Jana Mauch

Leila Feddoul

Heinz Nixdorf Chair for Distributed Information Systems

Jena, December 2022

Abstract

Semantic search aims at understanding the information need of a users query and thus improve the effectiveness of the search. To our knowledge, free, public search systems in the German legal domain rely on simple lexical matching for document retrieval. In this work we present a prototype for the semantic search over German legal norms. We compare vector space models (word2vec, doc2vec and SBERT) on their ability to calculate document embeddings for similarity based ranking. Furthermore, we implement two query expansion techniques. The first technique is based on pseudo relevance feedback in which the users initial query is expanded by terms found in the relevant results retrieved by the initial query. The second expansion technique is based on external knowledge found in a thesaurus. Additionally, a vector space model is used to identify a relevant document during pseudo relevance feedback and for the word-sense disambiguation of candidate terms in both techniques. An experiment, which was conducted with a law expert, showed that semantic techniques have the potential to aid the search over German legal documents.

Zusammenfassung

Die semantische Suche zielt darauf ab, das Informationsbedürfnis eines Nutzers zu verstehen und so die Effektivität der Suche zu verbessern. Unseres Wissens nach nutzen freie, öffentliche Suchsysteme im deutschen Rechtsbereich eine einfache Volltextsuche für das Abrufen von Dokumenten. In dieser Arbeit stellen wir einen Prototyp für die semantische Suche über deutsche Rechtsnormen und Gesetzestexte vor. Wir vergleichen Vektorraummodelle (word2vec, doc2vec und SBERT) hinsichtlich ihrer Fähigkeit, Dokument-Embeddings für ein Ähnlichkeitbasiertes Ranking zu berechnen. Darüber hinaus implementieren wir zwei Techniken zur Expansion von Suchanfragen. Die erste Technik basiert auf einem pseudo-Relevanz-Feedback-Ansatz, bei dem die ursprüngliche Suchanfrage des Benutzers mit Begriffen erweitert wird, die in den relevanten Ergebnissen der ursprünglichen Anfrage gefunden wurden. Die zweite Erweiterungstechnik nutzt externes Wissen, welches in einem Thesaurus gefunden wird. Zusätzlich wird ein Vektorraummodell verwendet, um ein relevantes Dokument während des Pseudo-Relevanz-Feedbacks zu identifizieren und für die Sinn-Disambiguierung von Begriffen in beiden Techniken. Eine Evaluation, die mit einem Rechtsexperten durchgeführt wurde, hat gezeigt, dass semantische Techniken das Potenzial haben die Suche in deutschen juristischen Dokumenten zu unterstützen.

Contents

Abbreviations

AQE	Automatic Query Expansion
BERT	Bidirectional Encoder Representation from Transformers
CBOW	Continuous Bag of Words
DCG	Discounted Cumulative Gain
ELI	The European Legislation Identifier
IQE	Interactive Query Expansion
IR	Information Retrieval
JSON	JavaScript Object Notation
NLP	Natural Language Processing
NLTK	Natural Language Toolkit
PV-DBOW	Distributed Bag Of Words Model
PV-DM	Distributed Memory Model of Paragraph Vectors
SBERT	Sentence Bidirectional Encoder Representation from Transformers
TF-IDF	Term Frequency-Inverse Document Frequency
VSM	Vector Space Model

List of Figures

List of Tables

1 Introduction

This work was conducted in the context of the "Canaréno" (Computerunterstütze ANAlyse elektronisch verfügbarer REchtsNOrmen) project, which aims at supporting the analysis of legal norms using machine learning, natural language processing and information extraction techniques. This norm analysis is performed as an initial step for creating digital processes for administrative services. The German "E-Gesetzgebung"[1] (Elektronisches Gesetzugebungsverfahren des Bundes) initiative focuses on the electronical support of the process of creating legal norms. Currently, steps involved in the process of creating new legal norms are supported inconsistently by information technology and various software solutions are employed. A consistent electronic legislative process offers many advantages, such as reducing redundancy caused by media disruption, data becoming permanently reusable, and individual steps being digitally supported and simplified.

The goal of this work is to improve the searchability of German legal norms. The rapid expansion of data on the Web poses problems regarding the efficient retrieval of information [WF08]. Due to this issue, there has been a rising interest in extending conventional information retrieval to counteract information overload. It has become apparent that utilizing the contextual information of resources can improve retrieval performance significantly, either through semantically annotated content or other semantic techniques. To the best of our knowledge, existing platforms for searching over German legal norms implement a purely full-text search, relying on lexical matching of query and document words to retrieve relevant results. This requires users to exactly match the query words with the documents terminology. While there exists research which applies semantic search techniques on German court decisions and retrieval for laws written in other languages (cf. Chapter 4), to our knowledge, there exists no such work explicitly for German laws.

Within the scope of this thesis, we explore natural language processing and query reformulation techniques for information retrieval of documents in the German legal domain. Semantic search can enable non-expert users to find relevant documents for their information need.

[1] plattform.egesetzgebung.bund.de (accessed 22.09.2022)

We focused on building a prototype, which is ready for use and could be easily extended. The searchable dataset consist of over 6500 laws, containing over 100,000 legal articles.

Our research goal is to augment traditional keyword-based search by expanding an input query with related terms. To achieve that we (1) compare the query expansion aided by a general thesaurus, (2) evaluate the query expansion aided by a domain-specific thesaurus, and (3) assess the query expansion through a pseudo-relevance-feedback approach. To find terms for expansion, we retrieve synonyms, hyponyms and hypernyms from thesauri, find relevant terms for an original query by pseudo-relevance-feedback, and use word embeddings to disambiguate the terms. We further evaluate different vector space models on their ability to model and rank document relevance based on the similarity between queries and documents. All of this is done in an effort to improve the searchability of German legal norms beyond traditional full text search. The following research questions are addressed:

- Which machine-readable sources of German laws are publicly available?
- Which vector space models are well suited for the retrieval of German legal norms?
- To what extend can the embedding models aid query expansion?
- How well can query expansion improve the retrieval of German laws and how do different thesauri compare to each other?

This thesis is structured as follows: First, we introduce the reader to the main concepts of this work by providing background information in Chapter 2 and the related work in Chapter 3. Afterwards, we will describe our conceptual approach in Chapter 5 and present details of its implementation in Chapter 6. The results of our techniques are then evaluated and compared in Chapter 7. We conclude in Chapter 8 by summarizing our work and giving an outlook on future research directions.

2 Background

In this chapter we aim to provide an explanation of the core concepts and describe the technical terms used in this work. We will explain the steps involved in information retrieval from data pre-processing to evaluation. Furthermore, we will offer an insight into query expansion techniques and vector space models that can be used for ranking and word-sense disambiguation in information retrieval.

2.1 Information Retrieval

Information Retrieval (IR) [SG01] aims to find relevant material from an unstructured resources of data to satisfy an information need. Typically, a user submits a query which is received by the search system. The search system analyzes the query and ranks the documents in the collection of resources and retrieves the highest ranking documents. In the case of IR over text documents, information is made available in an unstructured form. This contrasts querying structured knowledge bases (e.g., database) using queries that were structured in a specific format (e.g., Structured Query Language), where the information need is precisely defined. However, in this case users are required to know the data scheme [BBH16].

The steps of a common IR system include pre-processing and indexing of documents, query handling, and ranking.

2.2 Pre-processing

In this section, we describe the relevant concepts for the pre-processing of data. This step attempts to improve the quality of data through its modification.

2.2.1 Tokenization

In preparation of Natural Language Processing (NLP) tasks, tokenization is an important step in document pre-processing [WK92]. Tokenization is the process of dividing a stream of characters into a list of individual tokens, which are more suitable to be utilised for further applications (cf. Table 2.1). Moreover, this task is rather language dependent, as it often relies on the use of delimiters to extract tokens, such as white space tokenization. This poses a problem in languages like Chinese, where word boundaries are not as clear as in English or German. Apart from words, punctuation, numbers, or other characters can also be considered as individual tokens. Some tokenizers also include the splitting of larger textual units like paragraphs or sentences [Hab+98]. Tokenization is an important step to enable the usability of individual query words for IR and to prepare data for machine learning models.

2.2.2 Stemming

Stemming is an important linguistic procedure in IR and NLP applications, which aims at reducing words to their word roots. Different morphological forms of a word are assumed to have the same semantic meaning [Jiv11]. Furthermore, words with different meanings but similar word form should be kept separated. For IR, this means that the number of retrieved documents is increased, due to words in the search query being independent from their grammatical form, while also shortening the retrieval time, due to a lower number of distinct words in the indexed documents. Two problems that arise when stemming are (1) under-stemming, which occurs when two terms that are supposed to be stemmed to the same root are not, and (2) over-stemming, which occurs when two words that should be stemmed to different stems are stemmed to the same one. A common stemming approach is to remove any attached pre- and suffixes from terms and checking the existence of the stem using a dictionary. Other stemmers are corpus based or perform statistical analysis before removing affixes. Stemming is an important step during query modification, since the querys and the documents grammatical word forms are independent from each other. Popular English stemmers include Porters Stemmer [Por80] and Lovins Stemmer [Lov68]. A list of stemmers and their exemplary outputs on a German sentence can be found in Table 2.1.

2.2.3 Lemmatization

Stemming and lemmatization are often used synonymously, although they are not the same. Similar to stemming, the goal of lemmatization is to find the normalized form of a word [PLM04]. However, the normalized form of a word can differ from its stem. Hence the normalization of a word is usually referred to as its dictionary form, whereas stemming can

Table 2.1: Tokenizers, Stemmers and Lemmatizers applied to an example sentence

Input
'Anton ist Häuser besichtigen gegangen'

Tokenizer	**Output**
NLTK [Lop04]	['Anton', 'ist', 'Häuser', 'besichtigen', 'gegangen']
SoMaJo [PU16]	['Anton', 'ist', 'Häuser', 'besichtigen', 'gegangen']
Stemmer	**Output**
Porter [Por80]	'anton ist häuser besichtigen gegangen'
Lancaster [Pai90]	'anton ist häus besichtig gegang'
Snowball [Por01]	'anton ist haus besicht gegang'
Lemmatizer	**Output**
WordNet [Fel00]	'Anton ist Häuser besichtigen gegangen'
HanTa [War19]	'Anton sein Haus besichtigen gehen'
spaCy [KP+21]	'Anton sein Haus besichtigen gehen'
Simplemma [Bar22]	'Anton sein Haus besichtigen gehen'

reduce a word to a root form, which is not naturally occurring [Liu+12].Though, as the name suggests, lemmatization tries to obtain the lemma of a word, which implies the reduction of a word to its root form through knowledge about the part-of-speech or context of the words occurence [Jiv11]. Different grammatical forms of a word are mapped into the same lemma to be analyzed as a single item, thus decreasing the complexity of the analyzed text. A list of lemmatizers applicable to German can be found in Table 2.1.

2.2.4 Stop words

Stop words are the most common words of a natural language. They are important for the syntactic formation of a sentence, but carry only little significant semantic information [KR16]. Frequently occurring stop words are considered unimportant in NLP applications like text summarization and IR. Due to this, it is a common practice to remove stop words to improve performance [Now11]. Typical stop word categories include articles, conjunctions and prepositions. A common dictionary based approach to remove stop-words is to tokenize a target document and afterwards compare the documents individual words to a stop-word list. Different ways to generate stop-word lists include the use of frequency statistics [Zou+06].

2.3 Indexing

Indexing is the process that converts the given information into a format that allows for a simplified and more efficient data retrieval [Cel+06]. Information organised this way allows for much faster look-up times, since without indexing, it would be necessary for the search engine to scan every element in the document corpus. Indexing a document corpus consists of four steps: managing the documents, tokenizing the documents, the processing of terms, and finally building the index [KG16].

Different types of indexing have been proposed. Forward or direct indexing refers to the mapping of a document to the terms it contains. The direct index stores the terms and the frequencies of the terms present in the document. The inverted index, which is employed by search systems like Apache Lucene [BMI12], maps a term to its documents. It can be regarded as a list that stores the document IDs and term frequencies for each document in which the term is present. Looking up only the documents that are mapped to the relevant keywords results in an even shorter search duration. Other types of indices include the document index or the lexicon [KG16].

Different techniques for indexing have been proposed. During single-pass indexing, term posting lists are kept, written to disk if memory is exhausted, and finally merged to form the lexicon. No direct index is constructed. Two-pass indexing is slower than one-pass indexing, but the direct index is constructed. In the first-pass, a vocabulary is built and in the second pass an inverted index is created. If there is a need to store positional information of terms, block indexing can be used. During block indexing, text is split up into blocks and the direct or inverted index stores the block information [KG16].

2.4 Search

In this section, we explain the differences between traditional IR and semantic search, as well as provide information on different methods employed in semantic search.

2.4.1 Traditional Information Retrieval

Traditional IR relies almost solely on the occurrence of words in documents [GMM03]. Given a search request, the IR system tries to find all documents that match the query words. Queries can be conjunctive, where each additional word narrows down the retrieved documents since query words are connected with a logical ”AND” operation, or they may be disjunctive, in which case additional words enlarge the pool of retrieved documents because query words are connected with a logical "OR" operation. A pure full-text search does not take into account the semantic meaning of the words it searches for (e.g., the conjunctive query "street laws" will return all documents containing both "street" and "laws" and not documents containing only the word "car"). Doing so can increase the effectiveness of the IR [RP14].

2.4.2 Semantic Search

Semantic search is defined as the search process which utilizes semantic information, such as concept relations, during the steps of query construction, the execution of the search, and the presentation of the results [HJH07]. Bast et al. [BBH16] distinguish between keyword-based and question answering-based semantic search. Furthermore, they differentiate between text, knowledge bases, and combined data as data resources. Common techniques in a semantic search system include ranking (e.g., through machine learning vector space models), indexing, query modification, and ontology based matching, merging, and reasoning.

A semantic search engine is a system that enables users to express their search intent and search a data corpus more efficiently [DHC08]. It tries to understand the semantic meaning of the document corpus and the user query to increase search effectiveness. Generally, a search engine adapts to the domain it was designed for. Different types of search engines include Extensive-Markup-Language-based search engines [Luk+02], ontology based search engines [Mae+03], or hybrid approaches. When given a user query, a search engine retrieves documents in a ranked list, containing the most relevant results, often in descending order [Bif+05]. There are many methods to perform ranking [JBB14]. Among these are entity ranking, relationship ranking and semantic document ranking. A common approach to rank documents relies on the use of vector space models (cf. Subsection 2.7.5).

2.5 Query Modification

Ineffectiveness of IR is often caused by query inaccuracy, resulting from imprecise keywords. This occurs when users are unsure about the nature of the content and cannot describe the context of their information need adequately [Ooi+15]. Vocabulary mismatch is a common problem in IR. Here, either different words in natural language have the same concept, or the same word can describe different meanings. These phenomena are known as polysemy and synonymy, and may lead to inaccurate queries. The new terms that are added to the query during query expansion have to fit the querys concept. Word-sense disambiguation is defined as the task of finding the correct meaning of a word given its context. It ensures that the meaning of the added terms are fitting the meaning of the query, given a specific context [ZN12]. Word-sense disambiguation techniques can be classified into knowledge-based, supervised and unsupervised approaches [PS15]. Different techniques for query modification have been proposed in response to these problems in order to improve the quality of the query.

Query expansion is a type of query modification which aims at adding meaningful terms to the original query to reduce the ambiguity of natural language, thus increasing the chances of discovering relevant documents. In order to find terms that actually enhance IR effectiveness, different semantic properties (e.g., taxonomic relations) can be exploited [Man07]. In addition, we distinguish between automatic, manual or user-assisted approaches [Ooi+15].

One typical user-assisted approach is based on **relevance feedback**. It adds terms to the query from the most relevant documents retrieved using the initial query. In the relevance feedback loop, users are initially required to enter a query, which is used to retrieve results. Then, the user judges the retrieved documents based on their relevance. Afterwards, terms are extracted from the relevant documents and scored (e.g., through a formula). Usually the user is then shown a list of the highest scoring terms and he or she decides which terms are deemed relevant. Finally, the relevant terms are added to the query. The effectiveness of this approach depends on the methods used for the selection and weighting of new terms [BMS07]. An alternative to using users judgement is pseudo-relevance-feedback, which assumes the top ranked documents to be relevant for term selection and query expansion.

Another established approach for query expansion utilizes the information stored in knowledge bases. Knowledge bases can be either corpus dependent, meaning the knowledge was extracted only from our document corpus, or corpus independent. Such corpus independent knowledge structures include ontologies and lexical thesauri [Raz+19].

Gruber defines the term **ontology** as an explicit specification of a conceptualisation [Gru92]. At its core, an ontology is a structured collection of information which describes concepts and the relationships between them. Ontologies further set rules regarding classes of objects rather than instances of them. They range from describing only general concepts to being very domain specific [SDK]. Approaches using knowledge-bases add terms to a query prior to its submission. To overcome vocabulary mismatch, meaning the incorrect matching of query and document terms, a common first step is to annotate the documents with the concepts contained in the ontology through named entity recognition and indexed via the annotated triples [NC18]. Afterwards, the query terms are mapped to the concepts of the ontology and related terms are added to the query [BMS07]. The design and construction of domain specific ontologies is a difficult, time consuming, and labour intensive process [Kas99]. Therefore, different automatic and semi-automatic methodologies for ontology construction have been proposed [Zho07].

A **thesaurus** (which is often regarded as a low level ontology) can be described as a structured vocabulary that groups words with similar meanings together [KY00] and depicts the taxonomic relations between the terms. Thesauri can be hand-crafted or built automatically, and can, similar to ontologies, be domain specific or contain more general terms. Expanding a query with related terms from a thesaurus can improve performance [IS09]. Some of the most important conceptual relations between terms in a thesaurus include:

- **Synonym** Two words, X and Y are synonyms, if they generally have the same meaning.
- **Hyponym** X is a hyponym of Y, if the meaning of X is included in the meaning of Y.
- **Hypernym** X is a hypernym of Y, if the meaning of X includes the meaning of Y.

Query suggestion is another approach to solve query inaccuracy, where the system improves the effectiveness of a query by providing suggestions to the user by analyzing the users past actions [Cao+08]. Such personalized suggestions can be made based on all the queries that were submitted during the same session. During a session, it is assumed that the user searches for the same topics and changes the query to receive better results [He+09]. Another way to create suggestions is to keep track of a users click-through for each query.

2.6 Ranking

Ranking in IR is the process of determining the most relevant documents from a data resource. Given a query, the search system is asked to rank all documents based on their relevance to the query's information need and return a set amount of documents that scored the highest [Dat10].

Different theoretical models to perform ranking in IR have been proposed. One of the earliest methods for IR utilized the Boolean model. Within this model, query's are submitted in the form of a Boolean algebra. This approach is rather easy, but it ignores partial matches in documents and usually no ranking is performed. Probabilistic models rank documents based on the probability of the documents relevance to a search query.
Another common approach is to use machine learning models such as vector space models for document ranking. This approach addresses the problems of the Boolean method, mainly that it considers partial matches and scores documents. Vector space models for ranking are further explained in the next section [Dat10].

2.7 Vector Space Models

Vector Space Models (VSM) have been used in the context of IR for a long time. The idea behind VSMs is to represent each document and query as a vector. One of the earliest approaches, which assumed that the inverse space density could be used to find the best indexing vocabulary, was suggested by Salton et al. [SWY75]. It is a solution to the problem of applying analytical methods to text data. Vector spaces enable useful operations like similarity calculation through simple measures e.g., cosine-similarity for ranking documents [AX19].

One of the more common methods for the formation of word vectors is one-hot encoding [Rod+18]. This representation encodes each word in the corpus with a different vector, where the size of the vector is the same as the number of distinct words in the corpus. All values of that vector are equal to zero except for one position, in which the value is one. This value differentiates the words from one another. This method has two drawbacks: First, the size of the vocabulary defines the size of the vectors, thus limiting flexibility and increasing the risk of high dimensionality. Secondly, it captures neither the semantic nor syntactic meaning of a word, but simply differentiates between them.

Statistical Language Modeling describes the process of building a probabilistic model of the distribution of words in natural language. It can be used to predict the occurrence of a word, based on the sequence of words immediately preceding it. Language models learn based on example texts and are used in various NLP tasks. The models can be used in IR to rank documents and word-sense-disambiguation. In this section we aim to provide an understanding of the classical TF-IDF vector space model, word embeddings, and contextual models, which are models that represent words or paragraphs into real-value vectors.

2.7.1 TF-IDF

TF-IDF stands for Term Frequency-Inverse Document Frequency and is a weighted scheme which reflects the importance of a word in the document. TF-IDF assumes that a word which appears infrequently in the document corpus is more informative than a word which appears frequently in the corpus. Due to this this assumption, stop words are given lower importance. In TF-IDF, the value of a word increases with the number of appearances it has in the same document. Because of this, it is widely used in text summarization and categorization applications [CAS16].

2.7.2 Word Embeddings

A word embedding is a representation of a word in form of a vector [Xio+19]. The continuous representation of word embeddings solves the drawbacks of one-hot encoding. Vector sizes do not increase as quickly in proportion to the vocabulary size. Furthermore, words with a similar semantic meaning are located closely to one another in the vector space. Thus, a pair of similar words has a smaller distance than a pair of dissimilar words. The vectors also allow for the addition and subtraction of word relations (e.g., $king - man + woman = queen$). Initially, embeddings were generated by neural network language models. Later, log-bilinear models (Mnih and Hinton [MH07]) and negative sampling were introduced to increase performance. Algorithms like word2vec [Mik+13], FastText [Boj+17] and GloVe [PSM14] create word embedding vectors. Each algorithm has been trained on a huge, pre-processed corpus of exemplary text and the embeddings represent individual tokens in the training corpus' vocabulary. The similarity of two word vectors reflects the relatedness of their words meaning.

In 2013, Mikolov et al. [Mik+13] have introduced two log-linear models for learning word embeddings. The Continuous Bag-Of-Words (CBOW) and Skip-gram model are used as the main algorithms for techniques like word2vec or fastText. Both algorithms use surrounding words to calculate word embeddings.

2.7.2.1 Continuous Bag-Of-Words

Bag-of-Words models assume that the grammar and order of words in a document can be ignored, thus they disregard it. The CBOW algorithm predicts the center word by using a fixed window size of the context while trying to maximize the probability in Equation 2.1 [Xio+19], where T is the number of training words, $w_1, w_2, ..., w_T$ are training words, and c is the window size. A visualization of the algorithm can be found in Figure 2.1.

$$\frac{1}{T}\sum_{t=1}^{T}\sum_{-c\leq j\leq c, j\neq 0}\log p(w_t|w_{t+j}) \tag{2.1}$$

2.7.2.2 Skip-gram

Skip-gram is an approach similar to CBOW. Instead of predicting the center word based on its surrounding words, Skip-gram tries to predict the word context given the distributed

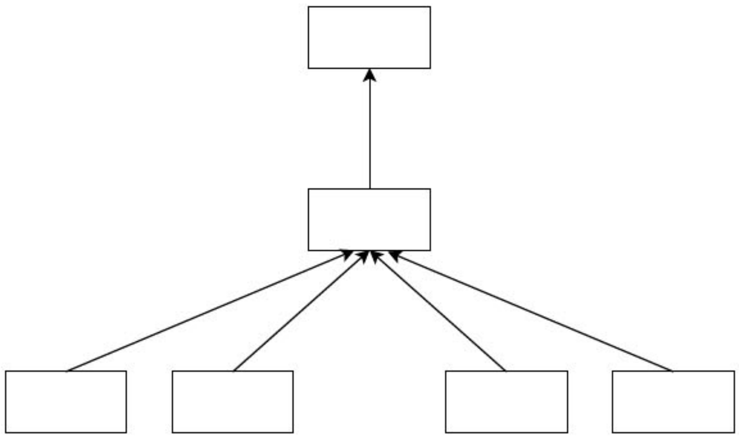


Figure 2.1: The CBOW learning algorithm [Xio+19]

representation of the input word. Note that Skip-gram is the inverted version of CBOW. Skip-gram tries to maximize the probability in Equation 2.2, where T is the number of training words, $w_1, w_2, ..., w_T$ are training words, and c is the window size. A visualization of the algorithm can be found in Figure 2.2.

$$\frac{1}{T}\sum_{t=1}^{T}\sum_{-c\leq j\leq c, j\neq 0}\log p(w_{t+j}|w_t) \tag{2.2}$$

The formulation of $p(w_{t+j}|w_t)$ is defined using the following softmax function (cf. Equation 2.3), where v_w is the input vector representation of w_t, W is the vocabulary size, and $v'_{w_{t+j}}$ and v'_{w_i} are the output representations of w_{t+j} and w_i:

$$p(w_{t+j}|w_t) = \frac{exp((v'_{w_{t+j}})^T v_{w_t}}{\sum_{i=1}^{W} exp((v'_{w_i})^T v_{w_t})} \tag{2.3}$$

2.7.3 Document Embeddings

Document embeddings represent entire paragraphs as one continuous vector. The disadvantage of Bag-of-words architectures for document embeddings is that different sentences containing the same words can have the same representation, even if their word order is distinct from one another. Due to this observation, Le and Mikolov have introduced the

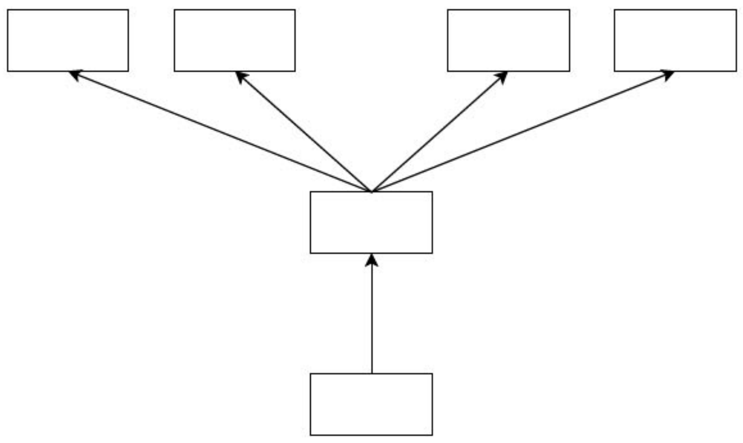


Figure 2.2: The Skip-gram learning algorithm [Xio+19]

unsupervised learning algorithm doc2vec, which forms paragraph tokens while learning word vectors to represent text segments or documents [LM14]. The Distributed Bag Of Words Model (PV-DBOW, cf. Figure 2.3) and Distributed Memory Model of Paragraph Vectors (PV-DM, cf. Figure 2.4) extend the word2vec architectures Skip-gram and CBOW [Lan20].

In PV-DM paragraph tokens can be thought of as another word and act as the contextual memory of the paragraph to help predict missing words. To do this, word and paragraph vectors are concatenated or averaged. PV-DBOW ignores context words in the input. Given a window size, the paragraph vector is trained to predict a number of words equal to it. In each iteration step of the gradient descend the text window and a random word in it are sampled and a classification task given the paragraph vector is formed.

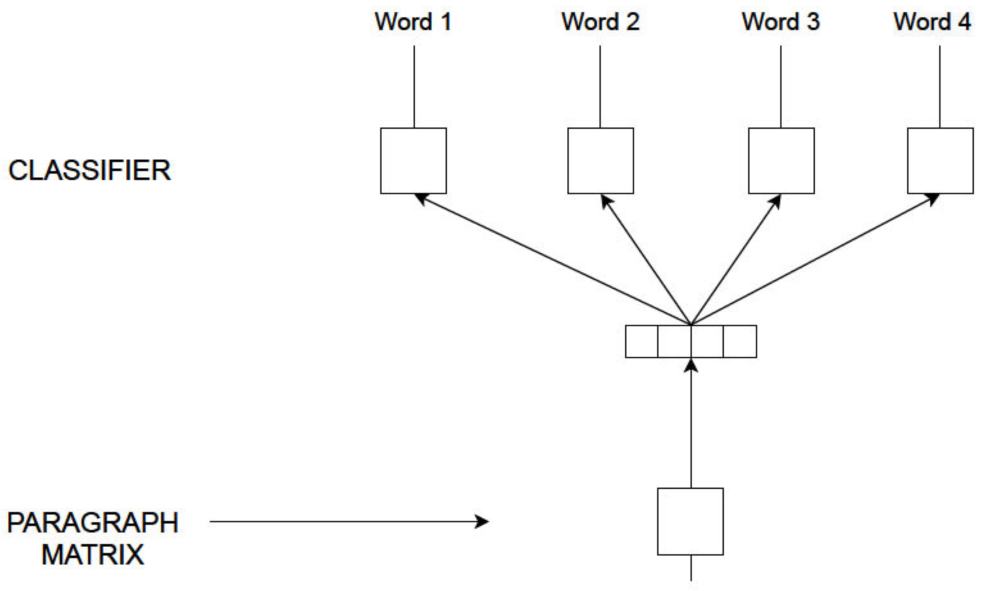


Figure 2.3: The PV-DBOW learning algorithm for document embeddings [Liu+19]

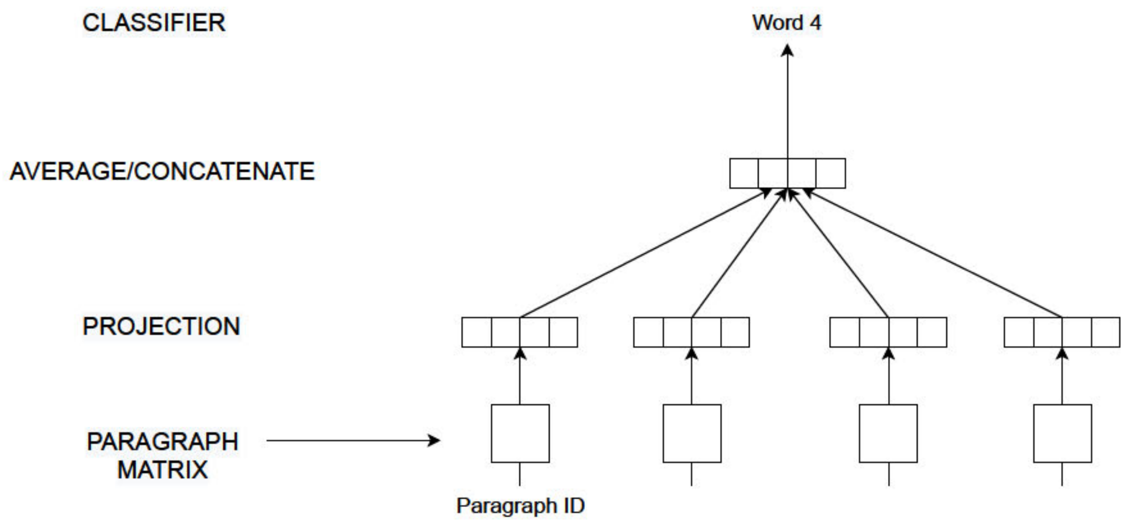


Figure 2.4: The PV-DM learning algorithm for document embeddings [Liu+19]

2.7.4 Contextual Embeddings and Transformers

Similar to the document embeddings, the contextual representation of a word also depends on its surrounding context. However, the contextual embeddings that are relevant for us use the Transformer architecture [Vas+17]. Instead of reading only a window of words at a time, Transformers read the entire sequence of tokens at once. In doing so, Transformers are able to represent a word while considering not only its immediate context. Thus, problems like word ambiguity and polysemy are handled.

ELMo, one of the earliest contextual encoders, was published by Peters et al. [Pet+18] in 2018. After that, there has been a steep competition between different models. In the next year, Devlin et al. [Dev+18] introduced the BERT (Bidirectional Encoder Representation from Transformers) model [XWD20]. This model was designed to pre-train contextual representations from unlabeled text. The bidirectional Transformer enables the processing of text from beginning to end and from end to beginning. Pre-trained BERT models can be used for a range of tasks, such as language inference and question answering. Although BERT is designed to be applied to different tasks without substantial modifications, different BERT models have been developed which are well suited for an asymmetric semantic search. The query in an asymmetric search is smaller in size than the documents that will be retrieved. SBERT (Sentence-BERT) utilizes siamese and triple network structures to compute sentence embeddings [RG19]. SBERT finds the most similar sentence pairs in a much shorter time than BERT while maintaining its accuracy. These sentence embeddings are especially suited for cosine-similarity comparison (cf. Subsection 2.7.5). A common step in applying a pre-trained language representation is fine tuning. During fine tuning, specific parameters are used to train the model on domain specific data.

2.7.5 Similarity Measurement

Vector representations enable the use of simple numerical operations for similarity computation. These measures reflect the semantic relatedness of words. Common measures include the cosine similarity, euclidean distance, and the dot product. Vectors of semantically related words or paragraphs tend to have a smaller angle, thus the cosine of the angle between two vectors of related words or paragraphs is higher than that of two unrelated terms. The calculation of the cosine similarity between two vectors $\vec{q}$ and $\vec{p}$, each containing float numbers, is depicted in Equation 2.4, where $\vec{q}_i$ and $\vec{p}_i$ are the float components of the vectors, and t the dimension size [RKA12]. The dot product works in similar way, but increases with the length of the vectors. The euclidean distance measures the distance between ends of two vectors, thus the euclidean distance between similar vectors is low compared to the distance between dissimilar vectors [Sid+14].

$$Sim(\vec{q},\vec{p}) = \frac{\vec{q}\cdot\vec{p}}{\|\vec{q}\|\|\vec{p}\|} = \frac{\sum_{i=1}^{t}\vec{q}_i\cdot\vec{p}_i}{\sqrt{\sum_{i=1}^{t}\vec{q}^2}\cdot\sqrt{\sum_{i=1}^{t}\vec{q}^2}} \tag{2.4}$$

2.8 Evaluation

Evaluation of IR systems is crucial, as it makes their performance measurable [Elb+15]. However, the lack of standardized evaluation has posed a problem for further advancements in the field of semantic search [Bla+13]. In general, we are not only concerned about the performance of an approach overall, but also about how it performs given different types of search queries. An evaluation study's success is dependent on choosing representative tasks that test a range of functions in an IR system [Ure+10]. One way to categorize IR evaluation is to differentiate between system-centric and user-centric evaluation [ST10].

System-centric evaluation compares IR systems ability to retrieve and rank relevant documents based on the user's information needs. Typically, IR focuses on maximizing precision, recall and other evaluation metrics e.g., the harmonic mean (F-measure). These metrics adopt the assumption of binary relevance and retrieval. A document is either relevant or irrelevant to a specific query and is either retrieved or not [Mea+06]. Precision is defined as the ratio of relevant and retrieved results, known as true positives (TP), to all retrieved results and can be calculated using Equation 2.5. False positives (FP) refers to the number of documents that were incorrectly marked as relevant. Recall is defined as the ratio of true positives to all relevant results and can be calculated using Equation 2.6, where true negatives (TN) refers to the number of correctly marked irrelevant documents. The F-measure is defined as the harmonic mean of precision and recall (cf. Equation 2.7) [SO17]. A set of queries can be evaluated using the mean average precision of the queries [Kor+11]. It can be calculated using Equation 2.8, where Q is the number of queries, and $AveP(q)$ the average precision of a query q. Furthermore, it has been argued that other factors such as complexity of user interaction or indexing and searching methods need to be considered as well, since they can affect retrieval performance [Gao+04].

$$\text{precision} = \frac{TP}{TP + FP} \tag{2.5}$$

$$\text{recall} = 2 \cdot \frac{TP}{TP + TN} \tag{2.6}$$

$$\text{F-score} = 2 \cdot \frac{\text{recall} \cdot \text{precision}}{\text{recall} + \text{precision}} \tag{2.7}$$

$$\text{Mean Average Precision} = \frac{\sum_{q=1}^{Q} \text{AveP}(q)}{Q} \tag{2.8}$$

However, using a binary relevance scale that determine a document to be either relevant or irrelevant is not always the best choice. The relevance of retrieved documents often resides in between relevance and irrelevance. The Discounted Cumulative Gain (DCG) [Jär+08] measures the usefulness of a document that was graded on a relevance scale, also taking into account its relative position in the results list. The DCG can be calculated using Equation 2.9, where i is the documents position in the results list, rel_i is the relevance score the document was ranked, and p is the number of documents in the results list.

$$\text{DCG} = \sum_{i=1}^{p} \frac{rel_i}{\log_2(i+1)} \tag{2.9}$$

One common practice involves the comparison of the retrieved results to a human expert crafted gold standard, which represents the ideal set of search results. Human experts usually represent only a relatively small user group, that have professional knowledge in e.g., a specific domain of interest. Due to this, another approach to evaluation is crowd-sourcing based, which already implies that instead of the judge being a small expert group, a large group of non-experts complete a higher number of assessments [Bla+13]. Both practices share a dependency on the human judges completing the evaluation task.

User-centric evaluation aims to assess the likelihood of an IR systems adoption and usage [ST10]. As with any sort of software, these factors are reliant on user satisfaction. User satisfaction depends on various aspects and is not necessarily captured by traditional evaluation metrics alone. Factors affecting a successful search include IR duration [WF08][WBB08], system and interface interaction [GJH07], and effort expectancy [Ven+03].

3 Related Work

In this chapter we survey published work in related research areas. We give an overview of developed query expansion approaches. Furthermore, we present some semantic search systems working specifically on the legal domain as well as other domains.

3.1 Query Expansion

Query Expansion has been known as a technique to improve retrieval effectiveness for some time. In 1988, Harman [Har88] proposed to use relevance feedback for query expansion. Given a query, the top ten documents are retrieved. Then, documents are marked as either relevant or irrelevant by a user. Afterwards, all non-common terms from the relevant documents are stored and sorted based on a statistical technique. Subsequently, 20 terms from the top of that list are added to the query. Harman observed significant performance improvements by making use of this method.

Few years later, Sanderson [San94] explained the importance of automatic word-sense disambiguation in IR and described techniques to realize it (e.g., utilizing machine readable corpora). Before that, the majority of sense disambiguation systems was based on manually crafted rules. Terms should only be added, if they fit the query's context [BMS07].

Efthimiadis [Eft96] investigated the process and effectiveness of interactive query expansion. Overall, he found that user satisfaction and precision were greatly increased by the proposed system. Another conclusion which was drawn was that users mainly selected narrower terms for query expansion. The question arose, if the repeated usage of the system acted as an iterative learning process for the users and if they chose concepts based on the process.

Kanaan et al. [Kan+08] compared interactive query expansion and automatic query expansion based on their retrieval effectiveness. For both systems they initially ranked documents using TF-IDF and extracted the most relevant terms from documents for relevance feedback. These terms were added to the query either automatically or interactively by a user. The techniques have been evaluated on a collection of Arabic documents. They concluded that interactive query expansion has the potential to be more stable and effective than automatic expansion, but requires the user to select either optimal or at least very fitting expansion terms.

General ontologies have been used for query expansion for a long time as well. Especially the lexical database WordNet [Mil95], which could be considered as a light-weight ontology[1], has been widely adopted for automatic query expansion. Vorhees [Voo94] uses the taxonomically related terms in WordNet to manually expand a query. Smeaton et al. [SKO95] determined the abstraction level of each term in the original query. If the term describes a broader meaning, a narrow term is added and vice versa. For terms in between both broader and narrower terms are added. Each added term completely replaces the original term of the query. The evaluation was conducted using shortened TREC-3 topic descriptions as queries. However, this approach performed quite poorly, which the authors attribute to an over simplistic weighting scheme.
Navigli and Velardi [NV03] argued that expansion terms should have the same level of abstraction as the words in the original query. An expansion with synonyms and hypernyms contained in general ontologies, such as WordNet, yields in only small performance improvements. Furthermore, terms for expansion should reside in the same semantic domain as the original query's terms to properly handle the problem of ambiguity. The system was evaluated using TREC2001 queries submitted to the web.
Due to this, domain specific ontologies are preferred over general ontologies for narrower search tasks [BMS07]. Fu et al. [FJA05] proposed a mixed approach for query expansion that aims at improving queries containing spatial terms through the utilization of two different ontologies that are incorporated in the SPIRIT system. When a user enters a query, spatial terms, which are terms that refer to a geographical position, are filtered and mapped a geographical ontology. The rest of the non-spatial or fuzzy spatial terms are mapped to a tourism domain ontology. The query is then expanded based on its geographical footprint, which was derived from the spatial terminology of the query. The evaluation was carried out on 4 exemplary queries each containing the same geographical positions but different relative terms. They concluded that their system can improve search significantly if a fuzzy spatial relationship (e.g., "north-of" or "near") is mentioned in the query.
Nilsson et al. [NHO05] developed SUiS, the Stockholm University Information system, which is an answering system for predefined questions. SUiS does not allow free queries, but guides users through predefined question types (e.g., who, what, when and where). Queries are expanded with synonyms and hyponyms found in the Stockholm University Ontology. They conducted a user evaluation which showed satisfactory results.

[1]Its creators do not call WordNet an ontology however, it is sometimes referred to be an ontology

Sihvonen and Vakkari [SV04] analyzed how users with different levels of expertise in the pedagogy domain interacted with terms extracted from the Education-Resources-Information-Center[2] thesaurus to expand a query. A pre- and post-search interview was carried out and the search logs of users were recorded. Their results show that the effectiveness of the expanded queries depended on the users level of expertise. While the expert group was able to select the correct type and number of expansion terms to improve retrieval, no such correlations between the thesaurus based query expansion and an improvement in search effectiveness could be made for queries submitted by the novice group.
Qiu and Frei [QF93] construct a similarity thesaurus based on the way terms are indexed by documents. A similarity thesaurus reflects knowledge about the collection it has been constructed from and is used to expand queries with similar terms. Unlike other approaches, the entire query concept is taken into account for query expansion (rather than single words). A term is added to the query if it is similar to the whole query concept, rather than the query's individual words. The experiment results showed that for different document collections the average precision improved by 18.31% - 29.12%.

Zhang et al. [Zha+16] proposed a two stage approach for query expansion to reduce time cost. In the first step, it is determined whether a query is suited for query expansion or not. In the second step, for all queries that were deemed to be suitable, a Rank-Support-Vector-Machine (RankSVM) model based query expansion is performed. The evaluation was conducted on three academic and one industry corpus. It showed that the proposed architecture could reduce the time cost of the query expansion substantially.

Kuzi et al. [KSK16] use word2vec's CBOW algorithm to expand a user query. The model was trained on TREC datasets. To find terms for expansion, Two methods were proposed. The centroid method aims to find the query's term vectors centroid and adds terms based on if their vectors are close to the centroid. The fusion based method regards each query term individually, and finds a list of candidate terms for each query term. Then, the final expansion terms are found through a normalized softmax function which yields their probabilities. Furthermore, these query expansion methods were integrated with a pseudo-relevance-feedback approach.

[2]https://eric.ed.gov/ (accessed 22.09.2022)

3.2 Semantic Search

3.2.1 Legal Domain

Eigenmann et al. [ES18] developed a prototype for the retrieval of German court decisions. Given an entire text as query, the prototype retrieves the five most similar court decisions. They used Elasticsearch to index their dataset, which consisted of 100,000 German court decisions. To calculate word embeddings for document ranking, the FastText [Boj+17] library was used and they achieved the best results using the Skip-gram algorithm with a vector dimension of 300. The evaluation was conducted with a law expert who rated each query and court decision pair. The ranking was based on the similarity of the pair and scaled on six levels, ranging from identical to completely different.

Landthaler et al. [Lan20] have analyzed different word embedding algorithms like FastText, GloVe, and word2vec. Afterwards, the embeddings were used to extend a German tax law legal thesaurus [Lan+17] and for query expansion. They found that the analyzed word embedding models are especially well suited to encode synonymity. Furthermore, the so called WE-DF (Word Embedding-Document Frequency) vector space model is used for text representation. It represents paragraphs by accumulating the word vectors of the paragraph. Users can interact with the search system by selecting text segments in documents as input for the IR system.

Yeung et al. [Yeu19] explored the effects of fine tuning different BERT models on OpenLegalData[3], which contains over 100,000 German court decisions. The evaluation was conducted on different tasks such as classification, regression, and retrieval based on similarity. They observed that for the similarity task, the fine tuned German legal BERT was able to capture legal information better than other VSMs, like TF-IDF and FastText. The German legal BERT had the highest mean average precision with a value of 0.2565, followed by TF-IDFs 0.2141 and German Bert with a value of 0.2025.

Altamarino Sainz [Alt15] assessed users requirements and built a user-centered, interactive search system based on the findings. The system employed a query reformulation tool, which suggests users terms that are related to the original query words. Candidate terms were either hyponyms, hypernyms, or sibling terms to the original query words. Users are able to interact with the a candidate list of terms to narrow down or broaden their queries. GermaNet was chosen as a lexical database and searches were performed over German law documents. Furthermore, an interview was conducted with an expert, which showed that the query reformulation tool provided supported for an effective search.

[3]http://openlegaldata.io (accessed 22.09.2022)

Schweighofer et al. [SG07] expanded user queries with terms found in a legal ontology to improve the results of legal information systems. The legal ontology was built contained around 5500 terms that were mostly reused from the Lexical-Ontologies-for-legal-Information-Sharing (LOIS) database [Din+05]. Furthermore, their proposed system uses relevance feedback to include the search context information in query expansion. The legal ontology was generated by performing text mining on a manually generated document collection.

3.2.2 Other Domains

Sistriani et al. [SBK20] developed a retrieval system for review articles in the tourism domain. The search system consists of two main parts. The first component performs search and ranking and is based on the TF-IDF VSM for representing documents and queries. The document and query vectors are compared through cosine similarity. The second component is the query modification. The terms of a query are matched with the concepts found in a tourism domain ontology and all terms that are somehow related to the concept are added to the query. The system was evaluated by users on 50 exemplary queries, for each of which the top ten documents were retrieved and labeled as either relevant or irrelevant. Sistriani et al. concluded that the query expansion could improve the mean average precision from a value of 0.315 to 0.563.

Chauhan et al. [Cha+13] proposed a domain ontology based query expansion system for semantic search over the sports domain. The introduced query expansion consist of two steps. First, the synonyms of query terms are obtained via WordNet. Afterwards, a second expansion based on the semantic similarity of concepts found in a sports domain ontology is conducted. If the similarity of a query concept is higher than the concrete threshold of 0.4, it is added to the set of candidate concepts. They evaluated their system on ten exemplary queries. Overall, the average precision increased from 0.622 without query expansion to 0.87 with query expansion and the average recall from 0.5 without query expansion to 1 with query expansion.

Löffler et al. [Löf+17] expand input queries with synonyms, and scientific and commons names from the GFBios [Die+14] open access Terminology Service. In a previous study they observed that scholars of the biological domain are unfamiliar with all taxonomic terms. Thus given a users query, datasets containing broader or narrower terms were not considered relevant by the users (e.g., when searching for "butterflies" datasets containing a subclass term like "Vanessa atalanta" were considered less relevant). The expanded terms are connected to the original search term via a logical OR. Search terms are connected with a logical AND operation. Only if there are no results for this default query, an OR search is carried out.

4 Analysis of the Legal Domain

In this chapter we analyze legal standards and systems for searching German legal norms. We further provide information on the advancements that have been made in the field of generating machine readable knowledge sources in the legal domain in the form of ontologies and thesauri. This analysis was done in an effort to find resources we could base our work on. On the one hand, this includes the data source we chose, while on the other hand it includes our choice of expansion methods and German legal thesaurus.

4.1 Standards

Akoma Ntoso[1] (Architecture for Knowledge-Oriented Management of African Normative Texts using Open Standards and Ontology) is an United-Nations-initiative which strives to develop standards for e-Parliament services in Africa [DGA17]. It was inspired by the European Committee for Standardisations Metalex [BHW02] and its purpose is to define a common document format, metadata scheme, and ontology for parliamentary, legislative, and judiciary documents. To achieve this goal, Akoma Ntoso was standardised by OASIS[2] (Organization for the Advancement of Structured Information Standards) and extended to the XML-based format LegalDocML[3], which describes the content of legislative documents. Palmirani et al. [Pal+11] have created an extension of RuleML [BTW01], a XML based rule standard used to model documents, called LegalRuleML, which captures constitutive and prescriptive statements to model legal norms. When they are used in combination, these models can be viewed as the literal and logical legal content. The German "E-Gesetzgebung"[4] project has adopted and adapted LegalDocML to meet the requirements of the German legal domain. LegalDocML.de is still in development at the moment. The first version covers the structure of German regulation drafts.

[1]http://akomantoso.org (accessed 22.09.2022)
[2]https://www.oasis-open.org/ (accessed 22.09.2022)
[3]https://www.oasis-open.org/committees/tc_home.php?wg_abbrev=legaldocml (accessed 22.09.2022)
[4]plattform.egesetzgebung.bund.de (accessed 22.09.2022)

4.2 German legal Search

The Federal Ministry of Justice of Germany provides almost all of the current laws in their applied form at "Gesetze im Internet"[Bun]. It also offers a keyword-based search service, allowing either for a logical "AND" operation, which retrieves the documents that contain all the query words, or a logical "OR" operation, which retrieves the documents containing at least one of the query words. Users may additionally decide between searching through only law titles or both, title and body of the text.

In this context, an alpha version of a "rechtsinformationsportal" [Tec] was developed by tech4germany. Its aim is to improve the browsing of laws and to increase usability. They worked together with users focusing on aspects such as advanced filtering options, better use- and readability of laws, and providing an overall easier to understand search to enhance user satisfaction. During the project they created an API[5] which offers various functionalities (e.g., listing and bulk download of laws) and is based on the law data found at "Gesetze im Internet". The API allows users to retrieve the information of single articles. Users can either input the exact article slug or a query string to retrieve articles. Another API is provided by "OffeneGesetze.de"[6], a nonprofit project of the Open Knowledge Foundation Deutschland e.V.[7]. The APIs[8] purpose is to enable access to German Federal Law Gazettes ("Bundesgesetzblätter"). Law Gazettes are published by the federal ministry of justice to update the public about newly published laws and all changes made to existing laws. However, while both approaches allow searching over legislative documents, to our knowledge, they both rely on full text or metadata search for document retrieval.

Beck-online[9] is a priced German legal database. It is organized in about 150 modules that cover different legal fields and occupation groups. Furthermore, it provides different publication types, such as legal norms, books, and newspapers. Beck-online can be personalized to fit a users needs. Buzer.de[10] is a German legal database that contains all publications in the federal law gazettes since 2006. This includes all applied and former legal norms. It offers a search system over all documents, in which it is simplified to find connections between norms. Buzer.de provides a full text search as well as title or abbreviation search for entire laws or single articles and paragraphs. The free legal information portal dejure.org[11] contains about 300 laws and over 1,900,000 court decisions. Although the law document count is

[5]https://api.rechtsinformationsportal.de/v1 (accessed 22.09.2022)
[6]https://offenegesetze.de (accessed 22.09.2022)
[7]https://okfn.de (accessed 22.09.2022)
[8]https://offenegesetze.de/daten (accessed 22.09.2022)
[9]https://beck-online.beck.de (accessed 22.09.2022)
[10]https://www.buzer.de/ (accessed 22.09.2022)
[11]https://dejure.org/ (accessed 22.09.2022)

relatively low, they note that it covers around 90 percent of the research need. It provides a full text search over the documents. Legal-Tech-In-Germany offers a list of the most common legal databases and search systems in the German legal domain[12].

4.3 Ontologies

Several models for legal ontologies have been proposed. First, we will briefly cover early theoretical approaches with a focus on already published, usable legal ontologies. Note that we disregard ontologies which only model parts of the legal domain, e.g. policies or licenses but focus on approaches that do not model a specific domain within the legal field but act as a more generic conceptualisation.

Valente et al. [VB94] have proposed the functional ontology of law, a connection of primitive categories of legal knowledge (e.g., Normative Knowledge, Responsibility Knowledge). The specification of their structure and relations between the categories aids legal comprehension. They assume the legal system to have an internal structure that aims at accomplishing a function. The principles of their core ontology are thought to be reused for other specifications [Val+96].
In 1995, van Kralingen has introduced the concept for a frame based ontology for law [Kra97]. van Kralingen focused on determining primitives to model the legal domain. The main elements are the norm-, act-, and concept-frame. They are used in order to create the structure of the model.

The European Legislation Identifier (ELI) intends to lower the entry barrier it takes to model the properties that describe European legislation [Fra+19]. It was created with the goal of including legislation as a part of the Web of data. Legislative metadata is viewed as a graph of connected entities and described based on the FRBRoo ontology[13]. ELI also proposes the use of URI (Uniform Resource Identifier), which serve as links to legislation and can be specified by publishers to adapt to different legal systems. So far, according to the European Union, a group of 20 countries have implemented ELI, which Germany is not a part of.

Eurovoc [Pub] is a light-weight ontology which was developed and is managed by the European Union. Its core function is to index documents published by the EU. Eurovoc is organized in 21 sectors and 127 sub-sectors. Each sector is linked to a conceptual field of competence of the EU. Within Eurovoc, concepts are lexicalized by a set of terms. One term of that set, called the preferred term, indexes the concept. For example, the German term for

[12]https://www.legal-tech-in-deutschland.de/juristische-datenbanken/ (accessed 22.09.2022)
[13]https://cidoc-crm.org/frbroo/ (accessed 22.09.2022)

”corruption” which resides in the sub-sector of criminal law belonging to the sector of law indexes all documents that are associated with the concept indexed by that term. Furthermore, terms in Eurovoc are interconnected through semantic relations, such as hierarchical and associative relations [LCV19].

Gandon et al. present an extension of the LegalRuleML ontology to model norms and rules [GGV17]. Normative requirements are represented as Linked Data and deontic operators organized in a hierarchical structure. The core concepts are formalized in OWL [LCV19].

Within the Lynx project [DP20] a legal knowledge graph was created from multilingual legal documents. Furthermore, a service oriented architecture was built for various tasks, such as summarization of documents, question answering and search, and terminologically related services. Key findings of the project were, that there is a gap between official and commercial sources when it comes to legal information provision, and that there is a need for uniform standards for legal information.

4.4 German Thesauri

Legivoc [VJP13] is a collection of multilingual legal vocabularies from member states of the EU. It aims at connecting laws from different legal systems, building on the foundation of Eurovoc (cf. Section 4.3) and extending it to allow for the linking of national law systems. Its vocabulary system contains legal terms in several European languages and it defines semantic relationships between them, namely synonyms, hypernyms, hyponyms, and related concepts. They also offer a SPARQL based API.

Wolters Kluwer German labor law thesaurus [Bas] contains vocabulary on legal issues, supplemented by terms of neighboring fields. However, as the name already suggests, it mainly contains words connected to the legal field of labor law. The labor law thesaurus holds 1728 terms and is listed in the Legivoc thesaurus.

Jurivoc [BüH98] is a thesaurus which was published in three different languages, German, French, and Italian by the Federal Supreme Court of Switzerland. It was designed to aid the indexing of legal documents. In both thesauri, terms are thematically structured and describe the same semantic relationships as Legivoc. However, Jurivoc is monohierarchical, meaning that a term is always assigned to a single hypernym.

4.5 Findings

Based on the knowledge we gained we drew several conclusions for the realization of our search system:

- There are APIs and search systems available which enable browsing or downloading of German legal norms. However the approaches do not make use of semantic techniques to improve search.
- The found ontologies do not describe the actual content of the legal document but focus on describing their structural content or use another concept abstraction level. Thus concepts and entities of existing ontologies and Knowledge graphs could not be used to semantically annotate documents in an effort to enhance document search and interpretation.
- Wolters Kluwer German labor law thesaurus can be used for testing domain specific query expansion.

Our system is similar to the ”rechtsinformationsportal” system. We use the API provided by them to download the same document dataset. However, in contrast to their system, we allow for an equal search of law articles and entire laws. Furthermore, we make use of semantic techniques, mainly the thesaurus and relevance feedback based query expansion, and embedding models, with an aim to improve search effectiveness.

5 | Concept

In this chapter, we will shortly emphasize the goal of this work and present the architecture of our proposed search system while explain its underlying concepts.

Traditional IR finds relevant results based on lexical matching. Thus, users are required to be familiar with the vocabulary of the search domain. To the best of our knowledge, current search systems of documents in the German legal domain do not go beyond mere string matching. Our approach focuses on improving the effectiveness of law retrieval by leveraging machine learning and semantic search techniques. Our goal is to offer a semantic search system that enables users to retrieve relevant documents and satisfy their information need without relying on exact matching of terms. We aim at lowering the entry barrier for users without legal expertise. For this purpose, we implement features that automatically help the user in trying to formulate his information need. However, to a certain degree there is a trade-off between automatic query expansion (AQE) and interactive query expansion (IQE). Research suggests that giving an experienced user more options to manually interact with a search system through IQE has bigger potential and can yield to better results [Ruto3]. Nonetheless, AQE is usually the preferred approach for users that do not have any domain knowledge as it does not require them to make further decisions beyond entering an initial query. For now, our approach focuses on AQE and thus aims at increasing the overall user satisfaction by presenting more suitable results.

The proposed approach is depicted in Figure 5.1. It uses "Gesetze im Internet" as a source for the law documents, which are pre-processed before embeddings calculation is performed for each document. The embedding is appended to the document and the documents, which now contain law information in the form of text and the appended embedding, is indexed by the search engine. The user interacts with the system by submitting keyword queries, that are first enriched using various AQE techniques. First, the query is expanded by using a pseudo relevance feedback approach. Afterwards, for each query word taxonomically related terms are retrieved from a thesaurus and added to the query. Query embeddings are calculated for the expanded query by the same model used for document embedding calculation. Finally, the query is submitted to the search engine, which performs search and returns relevant results to the user. In the following sections we will describe each step in more detail, as well as explain the query expansion via a representative example query in Table 5.2.

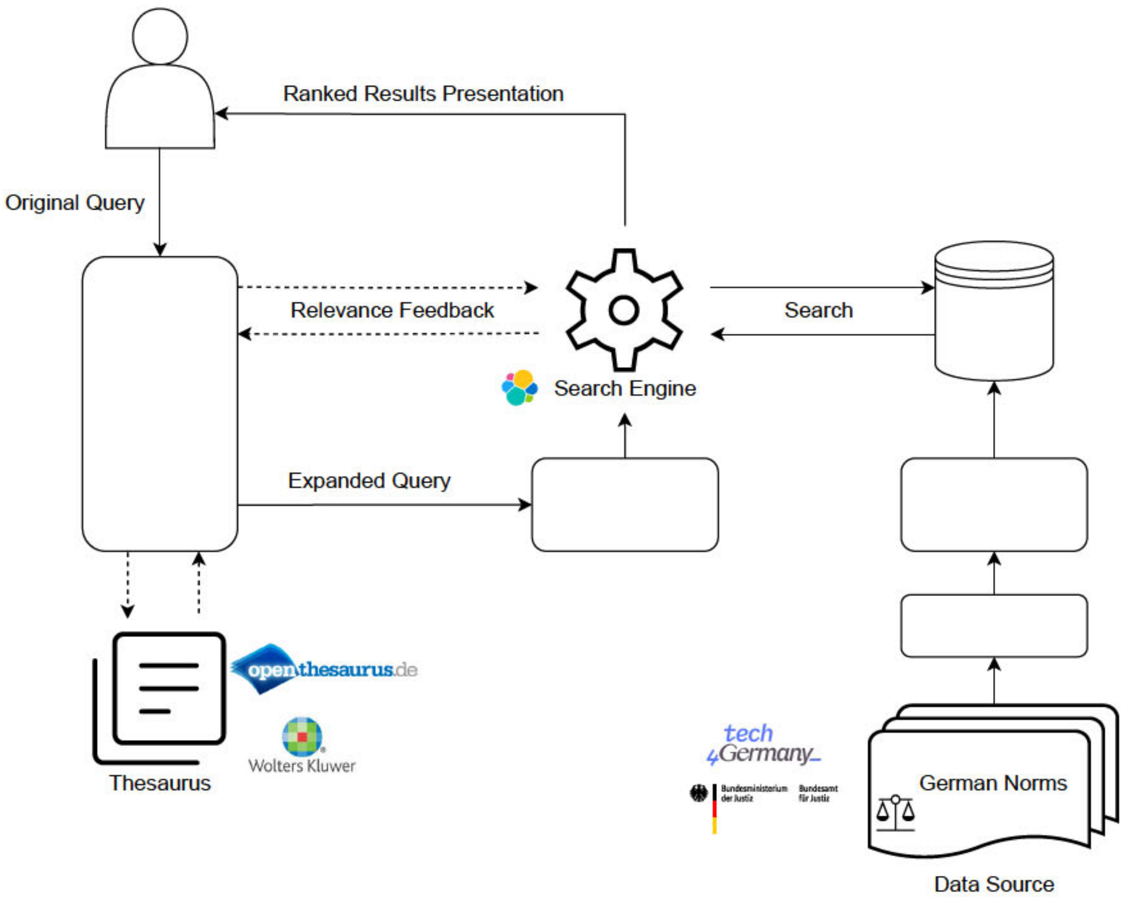


Figure 5.1: The proposed System Architecture [Tec] [Bun] [Ela] [Lan] [Wol]

5.1 Data Source

The Source of our data is "Gesetze im Internet" and it is accessed through the "Rechtsinformationsportal" API (cf. Chapter 4). It enables the bulk download of all laws in the JavaScript Object Notation (JSON) format. Each law can contain a number of articles, headings, and heading articles. The document types and their respective occurrences in the corpus is shown in Table 5.1. Our system treats all legal document types as equal, thus laws, articles, headings, and heading articles are saved as individual entries in the JSON data.

Table 5.1: Document Type, Definition, and Occurrence Count

Document Type	Definition	Occurrence Count
law	Collections of articles, possibly with levels of subsections.	6871
article	"Paragrafen" or "Artikel" that contain the textual content. Can belong to a heading they appear under.	104861
heading	Structure laws into subsections.	13983
headingArticle	Textual content of headings.	96
Total Count		125811

5.1.1 Pre-processing

Before using the law data, it needs to be prepared for the unsupervised learning algorithm. As a first step, the JSON fields that are relevant for our task are identified and accumulated. All other fields which were received from the rechtsinformationsportal are removed. Afterwards, additional irrelevant information is removed. This step depends on the embedding model the data is prepared for. In the case of the doc2vec and word2vec models, we remove stopwords, punctuation, and numbers. Furthermore the text is lemmatized, tokenized, all words are lowercased, and umlauts are replaced (e.g., "ä" is replaced by "ae"). Some of the steps are not applied when the data is prepared for the SBERT model. Figure 5.2 illustrates distribution of the number of pre-processed documents for each word count.

5.2 Embeddings

We have explored different VSMs to calculate query and document embeddings. We first calculate embeddings for each document in the collection. Then, for each user keyword query the corresponding embedding is calculated using the same model. This leaves us with high dimensional vectors, which enable us to use simple mathematical operations to rank each document after the retrieval process. In our case, we compute the cosine similarity between each document and query. The higher the similarity, the higher the score and ranking of the document is.

Initially, we tested the doc2vec model to calculate document embeddings, as this seemed like a perfect fit for our document-like laws and articles. We believed that the paragraph vectors of the doc2vec model could capture the context of these documents well. The model was trained

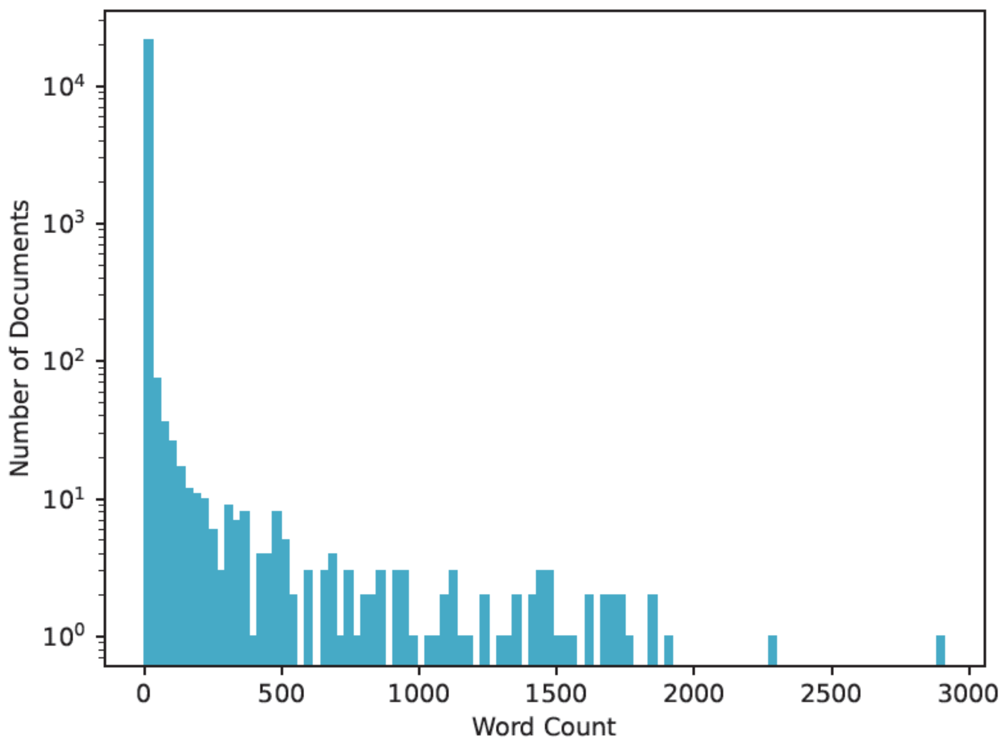


Figure 5.2: Number of pre-processed documents for each word count with logarithmic scale of Y-Axis

on the entire pre-processed document collection. Doc2vec calculates paragraph vectors for documents as well as word embeddings for every single word in the collection. The document embedding is calculated for each document and the vector information is appended to the document during the indexing process (cf. Section 5.3). The same doc2vec model is then used to calculate the embedding for user queries. However, during our testing it became apparent that choosing this model might not result in great IR performance. We believe this could be due to the nature of our search being an asymmetric one. This means that the input query is usually much smaller in size than the documents in the corpus. The paragraph vectors of the doc2vec model could have problems capturing the context of such a short query.

To address this issue, we refocused on single word embeddings to calculate the document and query vectors. The word2vec model was used to calculate word embeddings. The pre-processed documents undergo the same training procedure as with the doc2vec model. In order to enable the search via vector similarity, it is necessary to represent the entire document as a single vector. To achieve this, the mean of all vectors belonging to words that are present in the document is calculated. The same is done with the keywords of the user queries.

Another embedding model we explore is the BERT model. In particular, we chose a pre-trained multilingual SBERT model. The data pre-processing for SBERT models differs from the pre-processing for the other models. The SBERT model is a contextual model, which means that the sentences and paragraphs need to contain the largest amount of contextual information possible. To preserve this information, we do not entirely remove stopwords, punctuations, or numbers and we do not tokenize and lowercase each word. In fact, the pre-trained SBERT model contains a trained tokenizer, as tokenization for BERT models is different to simple word separation based tokenization. Like the other models, the SBERT model is used to map each document to a high dimensional vector, which is appended to the document and indexed by the search engine. The same is done for the user queries.

5.3 Search Engine

The search engine acts as the point of connection to our document data. We chose to work with the Elasticsearch platform [Ela], an Apache Lucene [BMI12] based search engine. The platform contains other components but we will solely use its search engine. In our architecture it serves two main functions. First, it indexes and stores the law documents in an index. Elasicsearch automatically creates a mapping for each index, which defines the field data type as text, keyword, geo-point, etc. and thus determines how an uploaded document will be processed by the search engine. Alternatively, mappings can also be defined manually. Elasticsearch is designed to work with documents that are stored as JSON files. Second, it accepts user queries, ranks documents based on a chosen metric, and returns them to the user. Elasticsearch provides support for embeddings through the dense_vector field type and pre-defined vector similarity based ranking functions. For our purpose, the query vector is passed on to the search engine and the documents are ranked based on the cosine similarity between the query and document vector. Elasticsearch then returns a ranked list of documents based on the similarity score back to the user in the JSON format. Okapi BM25 is the default scoring function of Elasticsearch. It is demonstrated in Equation 5.1, where q_i is the query term at position i, $IDF(q_i)$ is the inverse document frequency of q_i, $fieldLen$ is the length of the field that Elasticsearch searches (defined as the number of terms), $avgFieldLen$ is the field's average length, variable b influences the degree to which the $fieldLen$ compared to the $avgFieldLen$ effects the results, $f(q_i, D)$ is the number of occurrences of q_i in document D, and variable $k1$ determines how much a single query term can influence the score of a document.

$$\sum_{i}^{n} IDF(q_i)\frac{f(q_i, D)\cdot(k1+1)}{f(q_i, D)+k1\cdot(1-b+b\cdot\frac{fieldLen}{avgFieldLen})} \tag{5.1}$$

5.4 Query Expansion

Two techniques of query expansion are employed in our search system. The goal is to find terms that were not present in the original query, but are semantically related to the query context. This enriches the original query, since new terms could potentially give more insights about the users information need, that was not explicitly formalized in the query. However, it could also harm effectiveness, if the wrong terms are added to the original query. Our query expansion consists of two separate parts, namely a pseudo-relevance-feedback based approach and query expansion based on the use of thesauri. We assume the order by which these methods are applied to be highly relevant for the final results. Thus, the different combination variants are also evaluated in Chapter 7. The number of terms that should be added to the original query can be defined through a parameter. In our case, the parameter is set to three for the relevance feedback based method and to one for the thesaurus based technique. Table 5.2 shows the results of the expansion on an example query.

5.4.1 Relevance Feedback

We use a pseudo-relevance-feedback approach for query expansion. Relevance feedback is a technique which usually relies on user interaction to improve performance. Nonetheless, as we have mentioned earlier, we focus on AQE. First, the initial query is sent to the search engine and we retrieve the ten highest ranked documents through TF-IDF scoring. After that, the actually relevant documents are marked. This process is usually done by the user. In our case however, this is accomplished by calculating the cosine similarity between the document embeddings and query embeddings, which are generated using the SBERT model. All the stopwords, punctuation, and numbers are then removed from this new set of actually relevant documents. The left over text is tokenized and for each term t we calculate the wpq using Equation 5.2 [Kan+07], where N is the number of documents in the collection, R is the number of relevant documents retrieved by the intial query, rt is the number of relevant documents retrieved by the initial query which containt term t, and nt is the number of documents containing term t. The 15 terms with the highest wpq value are marked as candidates. Then, the word2vec embeddings are used to rank the terms one final time through their cosine similarity to the query. In our case, the three highest ranking terms are added to the query. Note that this number depends on the parameter defined beforehand.

$$\text{wpq(t)} = \log \frac{\frac{rt}{R-rt}}{\frac{nt-rt}{N-nt-R+rt}} \cdot \left(\frac{rt}{R} - \frac{nt-rt}{N-R}\right) \tag{5.2}$$

5.4.2 Thesaurus

We want to compare the expansion potential of a general thesaurus to that of a domain specific thesaurus. For the general thesaurus, we chose to work with OpenThesaurus [Nab05] and GermaNet [HF97] [HH10]. As of August 2022, OpenThesaurus contains 185,708 words and as of April 2022 GermaNet contains 205,000 lexical units. OpenThesaurus is a German Thesaurus which, unlike other wordnets, is not only managed by linguists, but follows an open collaboration principle. GermaNet is a lexical wordnet, which was developed in 1996 and is still maintained by a group of language and computer linguists. For the domain specific Thesaurus, we chose to work with Wolters Kluwer German labor law Thesaurus [Bas] (cf. Section 4.4).

The main steps of our thesaurus based query expansion are the same for every previously mentioned theaurus. For better comprehensibility we will refer to original query words as "words" and candidate words for an expansion as "terms". The results of a query should not depend on the grammatical forms of its words. In order to realize this, we lemmatize and tokenize the user query to make sure that the words are brought to their root form. After that we send a request to the thesaurus to find taxonomically related terms for each word in the original query. In response, we extract synonyms, hyponyms and hypernyms from the thesaurus. In the case of OpenThesaurus, we also retrieve substrings, which are subterms within the word, and similar terms, which are terms that are written similarly to the original query word. This could be very useful, if the user misspelled a word in the query. As a result we are left with a set consisting of terms which are each related to a query keyword. In the next step, we check for each term retrieved from the thesaurus whether or not it is contained in the document vocabulary. This vocabulary information is stored by the embedding model, which we have trained on the document collection earlier (cf. Section 5.2). This functions as a first filter to exclude out-of-vocabulary terms. Finally, for each of these found candidates we perform sense disambiguation. For this process, we calculate the terms embeddings and their similarity to the original queries embedding. This makes sure that the term is coherent with the querys context. In our approach, the terms with the highest similarity scores are then chosen as the expansion terms. The number of terms added this way is defined beforehand through the expansion parameter. However, since this could results in an irrelevant term being used for expansion, we make sure that the similarity value for all expansion terms is over a defined threshold. All the expansion terms are appended to the original query behind the word they were retrieved for.

Table 5.2: Modification and Expansion of an example Query

Original Query
'Arbeitsverhältnis Kündigung Ansprüche'

Modification Technique	**Modified Query**
Lemmatization	'Arbeitsverhältnis Kündigung Anspruch'
Umlaut Replacement	'Arbeitsverhaeltnis Kuendigung Anspruch'

Expansion Technique	**Expanded Query**
General Thesaurus *OpenThesaurus*	'Arbeitsverhaeltnis Beschaeftigungsverhaeltnis Kuendigung Ausscheiden Anspruch Recht'
Specific Thesaurus *Wolters Kluwer German labor law Thesaurus*	'Arbeitsverhaeltnis Beschaeftigungsverhaeltnis Kuendigung Entlassung Anspruch Vorschuss'
Relevance Feedback	'Arbeitsverhaeltnis Kuendigung Anspruch enden Arbeitgeber Arbeitnehmer'

6 Implementation

In this chapter we will provide a detailed explanation about our implementation. Note that improving the computational time was not of high importance during system development.

6.1 Data

As we mentioned before, the source of our law data is "Gesetze im Internet". We retrieve all current laws via the "Rechtsinformationsportal" API. It is updated every 24 hours with the latest laws and allows for a bulk download of all laws in a single JSON file. Nonetheless, the JSON file contains fields that are not important for our task at the moment (e.g., "release notes"). The relevant JSON fields need to be extracted. In our case, relevant JSON fields refer to the ones that contain actual legislative text. A list containing the fields that we deemed relevant for our purpose can be found in Table 6.1.

6.1.1 Pre-processing

The first step in the pre-processing of our data is to iterate over the original JSON file and extract the relevant fields. For that, all legal document types are taken into consideration. The fields are then appended to a new JSON file. This is important, since our chosen search engine is optimized for receiving a JSON input. All information extracted from the relevant fields is accumulated in a new "content" field. Note that for all document types except laws we also define a "belongs to" field that links a document to its law.

The documents have to be prepared differently to suit different embedding models.
In order to prepare them for the word2vec and doc2vec model, the German stop word list from the Natural Language Toolkit (NLTK) [Lop04] is loaded. Afterwards we replace all umlauts and use the list to remove all stop words from the document data. We additionally remove all punctuation and numbers. As a last pre-processing step, the text is lemmatized using the HanTa Lemmatizer (cf. Subsection 2.2.3), tokenized using the NLTK tokenizer, and

Table 6.1: Chosen JSON Fields and Descriptions

Field Name	Description
`type`	Indicates the type of the document. Can either be set to `law`, `article`, `heading`, or `headingArticle`
`titleShort`	Short version of the law title.
`titleLong`	Long version of the law title.
`abbreviation`	Abbreviations of the law title.
`notes`	Documentary footnotes of the law.
Article: `name`	Paragraph number of the document.
Article: `title`	The title of article, headings, or article headings
Article: `body`	Contains the main content of an article, heading, or heading article.

lowercased. Each document is processed this way and all documents are stored in a new JSON file.
The SBERT model requires the data to contain contextual information. This means, that the removal of stop words, removal of numbers, and lemmatization would negatively influence the results. Thus, pre-processing for the SBERT model consists only of replacement of umlauts and removal of punctuation (except for punctuation indicating sentence endings). Note that we are using a pre-trained SBERT model, which includes a pre-trained tokenizer.

6.2 Embeddings

For the embeddings of our documents and the query we compare three models: word2vec, doc2vec, and SBERT. The word2vec and SBERT model are additionally used by query expansion (cf. Section 6.4).

6.2.1 Doc2vec

The model was trained on the whole pre-processed document data using the PV-DM algorithm (cf. Subsection 2.7.3). The window parameter is set to 5, the epochs are set to 150, and the size of the embedding vectors is set to 500. Furthermore, the minimal word count is set to two, meaning that all words with a total frequency lower than two are ignored. The model also manages a vocabulary of all the words occurring in the document corpus. To extract a documents vector, doc2vecs ”`infer_vector`” function is called. For each document, a vector is extracted and appended to it in a new vector field. Finally, the documents containing title, legislative text, and the vector embedding are stored in a JSON file.

6.2.2 Word2vec

The model is trained on the whole pre-processed document data using the CBOW algorithm (cf. Subsection 2.7.2). To ensure a fair comparison, the additional parameters were set exactly like doc2vec's parameters. The word2vec model calculates embedding vectors for single words and stores the word vectors in a vocabulary. We need to represent the whole document as one single vector, rather then multiple vectors for our purpose. We calculate the mean vector of a document through using the contained single word vectors. Finally, the vector is appended to the document in a new vector field and all documents, which are made up of title, legislative text, and vector embedding, are written into a JSON file.

6.2.3 SBERT

We chose to work with the ”`distiluse-base-multilingual-cased-v1`” [Hug] pre-trained SBERT model. This choice was made based on our research that, to our knowledge, there is no German SBERT model. The model can be accessed through the ”Sentence-Transformer” Python framework [RG19]. This SBERT model is multilingual, meaning that a number of languages are supported (in this case 15), German being one of them. This model maps a document to a vector of size 512, which is comparable to the other models. Like the other models, the vector is appended to the document in a new vector field and all documents, which contain the same fields as the documents from the other embedding models, are stored in a JSON file.

6.3 Search Engine

The Search Engine represents the core of our system. It allows us to interact with the document corpus through queries. We worked with the Elastic Cloud [Ela]. Elasticsearch accepts and indexes JSON files as document inputs and allows the user to manually define how the JSON file should be stored through a mapping. The mapping contains a list of relevant fields in the document and their interpretation as Elasticsearch field types. In our case, it is used to set the field type of our JSON field "`vector`" to Elasticsearches "`dense_vector`" field type. Elasticsearch offers predefined similarity measures for the "`dense_vector`" field type like cosine similarity ranking. Furthermore, it provides other predefined ranking algorithms like Okapi BM25 [Cona] [Conb], an extension of TF-IDF. Okapi BM25 (cf. Section 5.3) is used during the pseudo-relevance-feedback based query expansion (cf. Subsection 6.4.1). By default, Elasticsearch sets the value of b to 0.75 and the value of $k1$ to 1.2. The documents are uploaded through the Elasticsearch Python client[1]. The structure of an example article document can be found in the following listing:

```
{
        "type": "article",
        "belongs to": "Verordnung zur Durchführung gemeinschaftsrechtlicher und ↩
            unionsrechtlicher Vorschriften über Maßnahmen zur Bekämpfung, Überwachung und ↩
            Beobachtung der Blauzungenkrankheit",
        "content": " § 4a Wildtieruntersuchung, weitergehende Maßnahmen (1) Die zuständige ↩
            Behörde kann zur Erkennung der Blauzungenkrankheit bei empfänglichen ↩
            Wildwiederkäuern Untersuchungen anordnen. ...",
        "vector": [
        0.019904183223843575,
        0.0068108271807432175,
        0.134660542011261,
                    ⋮
        -0.032890062779188156
        ]
}
```

To submit a query, search parameters have to be defined beforehand. The number of returned documents is set to ten and the fields that are returned in each document are the "`type`", "`content`", and the "`belongs to`" field (in case the document type is either article, heading, or heading article). For the vector based search, the documents are ranked using the cosine similarity (cf. Subsection 2.7.5) of the query vector and the document vector. In our case, we add one to the cosine score, since scores could fall below zero because the cosine similarity range is $[-1, 1]$. This would result in an Elasticsearch error due to the negative values. Thus the cosine similarity scores reside in the range of $[0, 2]$.

[1]https://elasticsearch-py.readthedocs.io/en/v8.4.3/ (accessed 08.10.2022)

6.4 Query Expansion

Two approaches to AQE have been implemented. For bot of them the resulting expanded query is lemmatized using the HanTa lemmatizer and converted into a vector by one of our embedding models (doc2vec, word2vec, or SBERT). The vector is then used as input for the search engine to retrieve relevant results.

6.4.1 Relevance Feedback

After the initial query has been entered, it is submitted to Elasticsearch and the ten highest ranked documents are retrieved using Okapi BM25. The documents are then pre-processed like the documents for the word2vec and doc2vec models. Afterwards, the SBERT model is used to calculate the embedding for every document and the original query. For each document, the cosine similarity to the query is computed using the PyTorch framework. If the similarity score is higher than the fixed threshold of 0.2, the document is marked as "relevant". Note that it is a common practice to leave the relevance marking to the user, however, we are focusing on an AQE approach. In the next step all documents marked as relevant are tokenized and for each term in the documents the wpq value is calculated (cf. Equation 5.2). In order to accumulate all necessary variables to calculate the value, a query is sent to Elasticsearch for each term in the relevant documents. The terms are then sorted by their wpq values in descending order. The 15 highest ranked terms are chosen as candidates for the expansion.

During the second step, the 15 terms are disambiguated using the word2vec model. First, the similarity between the word embeddings of each candidate term and query term are calculated. If the similarity value is below a threshold of 0.1, the term is discarded. Furthermore, we calculate the similarity between each candidate terms embedding and the entire query's embedding. We extract the entire query's embedding through the mean of its individual word embeddings. If the similarity between the candidate terms embedding and the entire queries embedding is higher than 0.1, the term is stored in a list with its similarity value to the query's embedding. Finally, the list is sorted in descending order by the same value. The 3 highest scoring terms are chosen as an expansion and appended to the original query[2].

[2]Note that the number of expansion terms, which we refer to as the expansion parameter, can be chosen by the user beforehand

6.4.2 Thesaurus

Within this work, we compare two types of thesauri: generic and domain specific. However, the underlying procedures are very similar. We access the generic thesaurus OpenThesaurus via the provided API. It allows us to submit a term and returns synonyms, hyponyms, and hypernyms of the term in a JSON file. The local GermaNet data source is accessed through their Python API. Wolters Kluwer German labor law thesaurus was manually downloaded from "bartoc.org" [Bas] beforehand and stored in a JSON file. When searching for a word, we check the entire file to find synonyms, hyponyms, and hypernyms.

The following expansion process is identical for both types of thesauri. First, the user query is tokenized. Afterwards, for each word in the query we retrieve synonyms, hyponyms, and hypernyms from the thesaurus. For every term retrieved that way, we check if the term is present in the document corpus. All terms which are not included are discarded. All terms that remain are stored in a list and disambiguated using the same disambiguation procedure as described in Subsection 6.4.1. This time the highest ranked term is chosen for expansion (Note that this number can be chosen by the user). In contrast to the pseudo-relevance-feedback based expansion, the expansion term is appended right after the original query's word which it was extracted for.

7 Evaluation

In this chapter we descibe the system's evaluation setup and we present and analyse the results. The evaluation was conducted together with a law expert. Example queries were chosen by the expert, for each of which we have applied our Semantic Search techniques. The retrieved documents were presented to the expert, who rated each document on a given scale. Furthermore, an interview was conducted with the law expert to gain further insights which could improve the system.

As a first step of this evaluation, ten queries were found together with the law expert. We want to point out that we did not know how well these queries would be processed by our system beforehand. Our focus was simply to try to cover various aspects of German law using the ten queries. Note that four of the queries fell into the field of labor law (Queries 3-6, cf. Table 7.3) and that we attempted to formulate them using words that non-expert users might use. This way we also have the opportunity to evaluate our expansion which is based on a domain specific thesaurus. We also attempted to formulate the information need for each query, since it is crucial for rating results. The law expert needs to know the information need for each query in order to assess if a given document result is relevant for the information need or not.

Each of the queries was then submitted to our search system. For relevance feedback we set the expansion parameter to three and for the thesaurus based query expansion we set the parameter to one. This means, that relevance feedback will append a maximum of three terms to the original query and the thesaurus based expansion will add one term for every word in the original query. The queries and their expansions are depicted in Table 7.3. For clarity, we only listed the terms that were added to the query. The system then retrieved the ten highest ranked documents for each query. The retrieved results were presented to the expert for scoring. At no point did the expert know which method he was currently evaluating. Furthermore, the order of the ten retrieved documents was mixed up. We chose a 3-point scale instead of a binary one (cf. Table 7.1). This is useful, since sometimes a retrieved document is neither irrelevant nor entirely relevant, but somewhere in between. Tables

Table 7.1: Rating Scores and Descriptions

Rating Score	Description
1 - Irrelevant	The document and query generally describe different areas of law.
2 - Relevant	The document or parts of it fit the information need of the query.
3 - Highly Relevant	The document satisfies the information need of the query completely.

containing the results of each query and method can be found in Table A.3 and Table A.2, where "GT" refers to "general thesaurus expansion", "ST" refers to "domain specific thesaurus expansion" and "RF" refers to "Relevance Feedback based Query Expansion". Furthermore, combined techniques are denoted with a "+" and the order can be read from left to right (e.g. "GT + RF" refers to "First GT is applied and afterwards RF is applied").

7.1 Embeddings

A key part of this evaluation is the comparison of different embedding models. Okapi BM25 is a ranking method based on TF-IDF but overcomes some of the issues TF-IDF has and serves as a baseline for this evaluation. Our system supports the calculation of embeddings for documents and queries using doc2vec, word2vec or SBERT. We want to compare which of these Embedding models performs best. In all three cases, the configurations are the same as described earlier in Chapter 6 and the queries were not expanded. The results of this comparison are visualized in Figure 7.1.

Okapi BM25 has an average discounted cumulative gain (DCG) of 7.833 over all ten queries. Word2vec has the highest score with an average DCG of 8.44. The SBERT model, which we suspected to score the highest since it is a contextual model, has a score of 7.316. As we expected, doc2vec scores the lowest with an average DCG of 4.857. As we can observe from Table A.3 and the per query results for each method that are visualized in Figure 7.2, the score of the same query can vary greatly between the different methods and between queries. For example, for queries 1, 4, and 7 SBERT scores better than Okapi BM25. Okapi BM25 scores 8.091 for the first query, while SBERT scores 11.24. However, SBERT only scores a 5.805 for query number 10, while Okapi BM25 scores 8.476. We also notice however, that doc2vec has

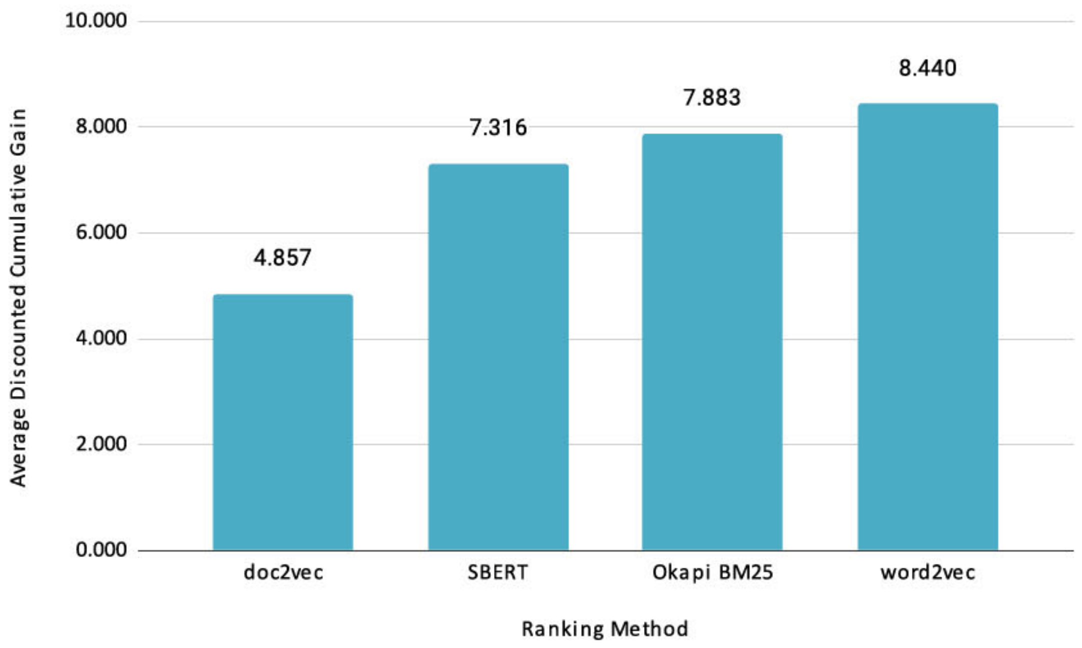


Figure 7.1: Average Discounted Cumulative Gain of Embedding Model

the lowest score for all queries among all four methods.

Generally, through the Okapi BM25 baseline we can observe that some of the queries formulated the information need better than other queries to begin with. Furthermore, the scores for the embedding based retrieval show improvements for some queries as well as a decrease of the average DCG for others. For example, query 5 scores an average DCG of 10.084 with the Okapi BM25 based ranking and a DCG of 5.805 with the SBERT based method, whereas query 1 has DCG score of 8.091 with the Okapi BM25 based ranking method and a DCG of 11.24 with the SBERT based embedding model.

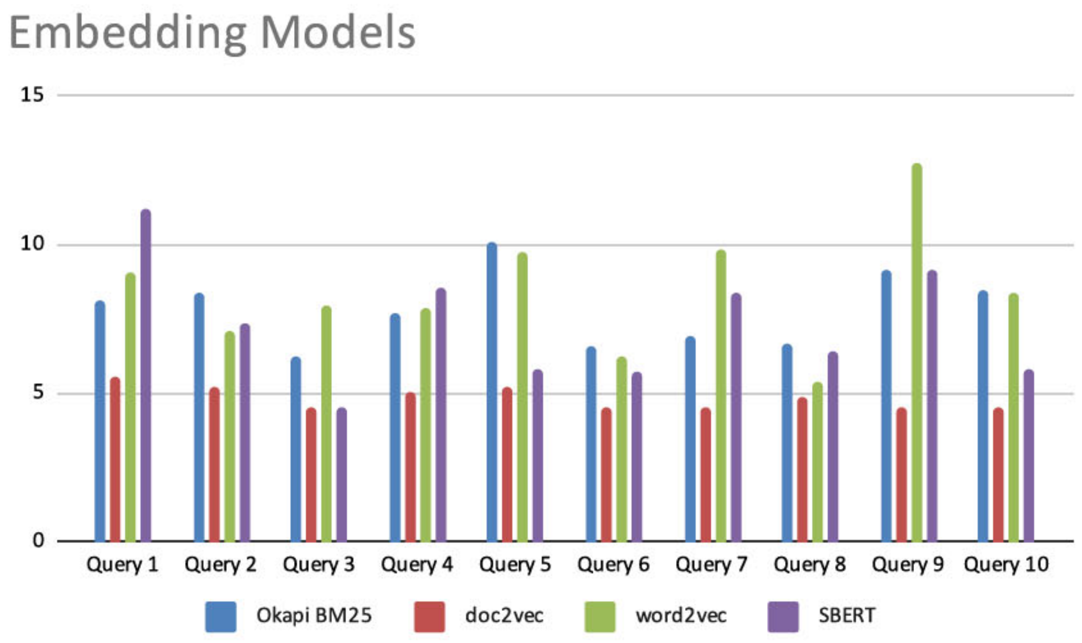


Figure 7.2: Discounted Cumulative Gain of each Query and Embedding Model

7.2 Query Expansion

The other main part of this evaluation is the comparison of our system's AQE techniques. Here, we compare each individual technique, as well as their results when they are used in combination. Another factor is the order in which the techniques are applied, if they are used in combination. The expansion results as well as their computational time can be seen in Table A.2 and Table A.3, and Table 7.2. The techniques are further combined with either word2vec or SBERT, as we assumed them to be the best performing embedding models based on the previous analysis. Due to time limitations, we chose to only evaluate the expansion using OpenThesaurus as a general thesaurus. The expansion of our queries using GermaNet can be found in Table A.1.

For the comparison of individual query expansion methods we evaluated them using SBERT as model for embedding calculation and baseline. A future evaluation should include an evaluation of the individual methods in combination with the other embedding models. SBERT was chosen as the sole model for this part due to time limitations. The results are visualized in Figure 7.3. The general thesaurus based expansion (SBERT + GT) improves the average DCG slightly with a score of 7.523 and an improvement of 0.207 compared to the baseline. The specific thesaurus expansion (SBERT + ST) scores slightly lower with a score of 6.961 and a decrease of 0.355 compared to the SBERT. The best individual technique in our evaluation is the relevance feedback based approach with an average DCG of 7.861, which is an improvement of 0.545 compared to the baseline. Note that even with expansion the SBERT model did not reach the performance of the word2vec model.

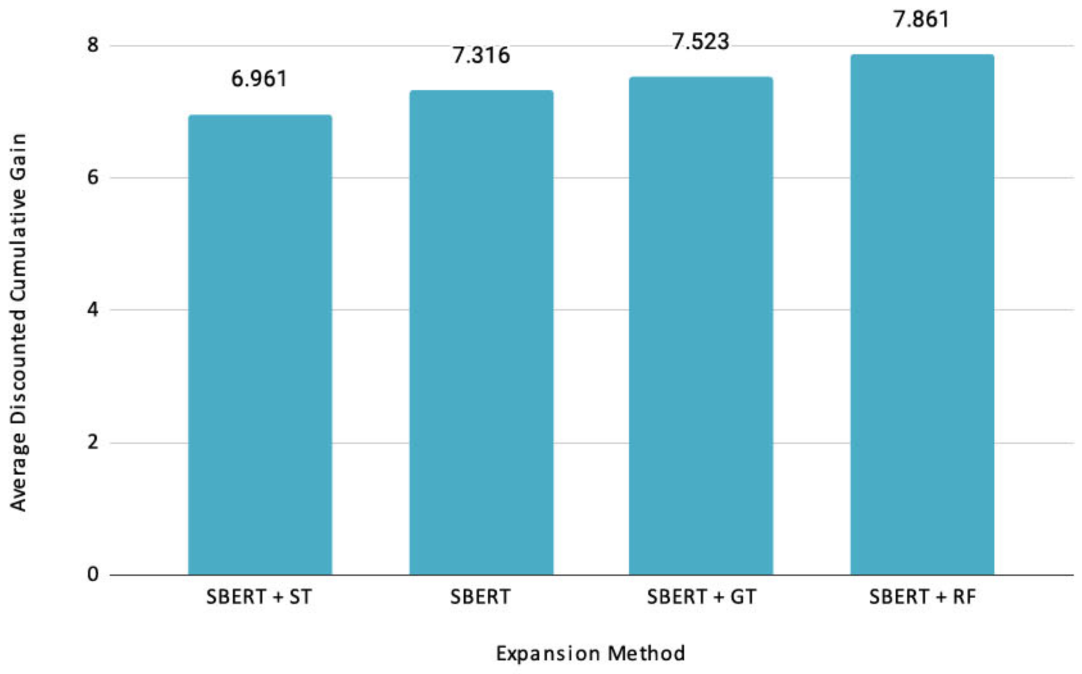
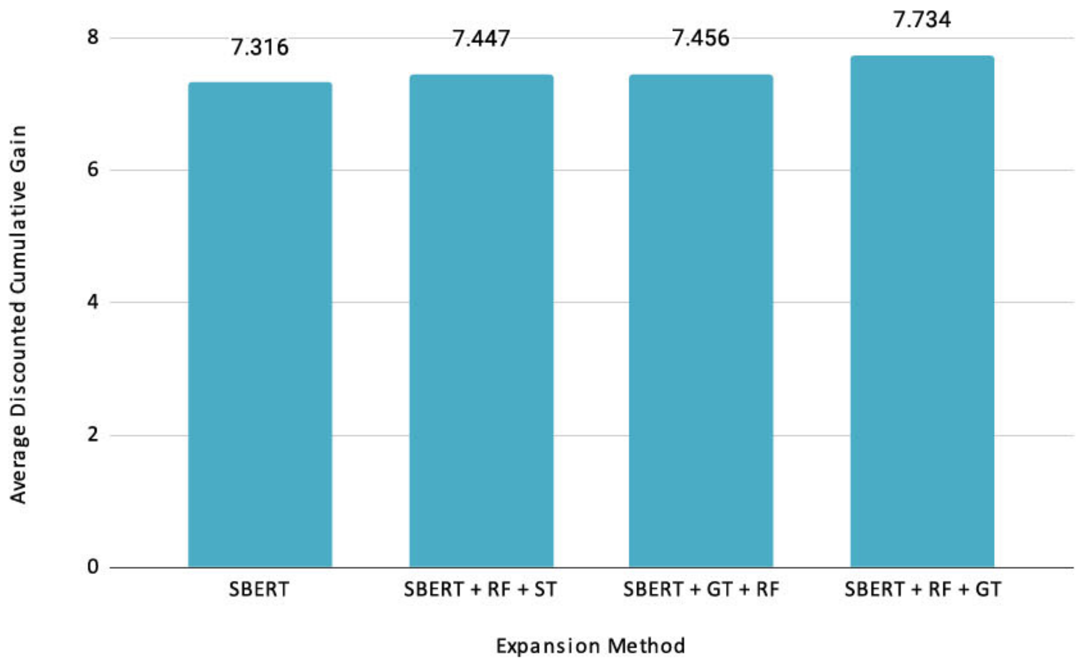


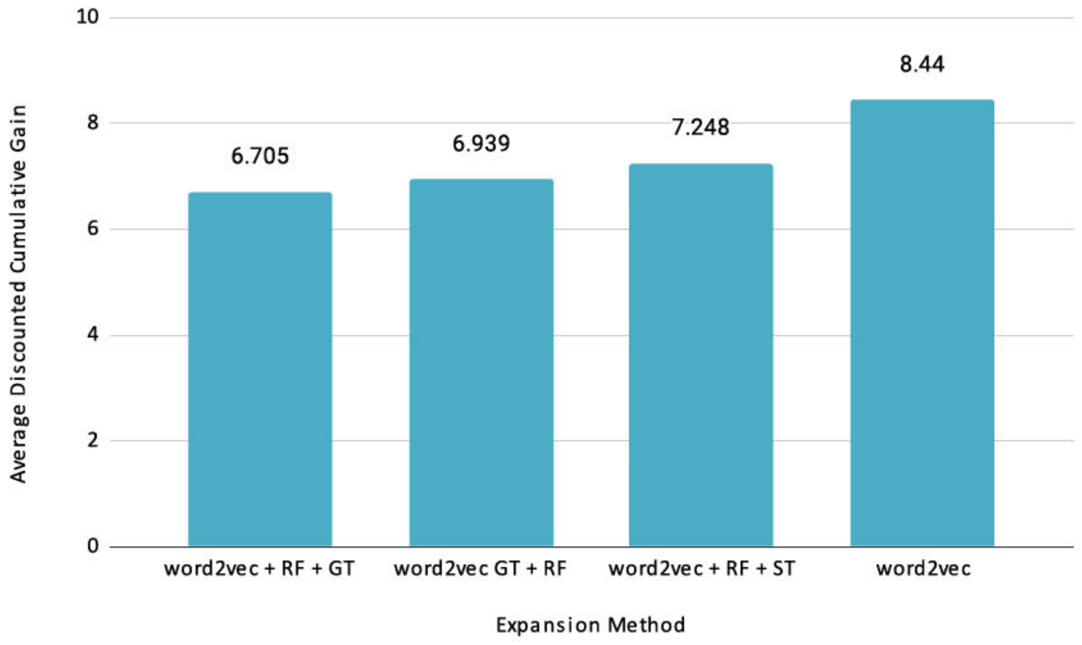
Figure 7.3: Average Discounted Cumulative Gain of individual Expansion Methods

The comparison of combined techniques was conducted with SBERT and word2vec as embedding models and baselines. The opposite order was only evaluated for the GT + RF combination due to time limitations. The results of the combined techniques with SBERT as the embedding model are visualized in Figure 7.4a and the results of the combined techniques with word2vec are depicted in Figure 7.4b. In general, we notice that the SBERT model profits more from the combined expansion than the word2vec model. SBERT + RF + GT scores the highest with a DCG of 7.734, an improvement of 0.418 to the baseline, followed by SBERT model with the opposite order of expansion with an average DCG of 7.456, which is an improvement of 0.14 and the SBERT + RF + ST with a DCG of 7.447, which is improvement of 0.131. Word2vec scores 6.705 when expanded by relevance feedback followed by the general thesaurus and 6.939 in the opposite order. The combination of word2vec + RF + ST scores the highest with an average DCG of 7.248. In this evaluation, word2vec scores lower when combined with expansion techniques as opposed to without it.

Overall, SBERT + RF is the best performing expansion technique, followed by SBERT + RF + GT and its opposite order, SBERT + GT + RF. Note that word2vec without expansion still performs the best.



(a) Average Discounted Cumulative Gain of Combined Expansion Methods and SBERT



(b) Average Discounted Cumulative Gain of Combined Expansion Methods and word2vec

Figure 7.4: Average Discounted Cumulative Gain of Combined Expansion Methods

The results of each query for each expansion technique are visualized in Figure 7.5. For queries 1 and 9, which score on the higher end of all queries , the GT- and RF-based expansion mostly add related terms to the query. In the case of query 9 the GT-expansion adds the terms

"Person" ("person") and "Verpflichtung" ("obligation") and the RF-based expansion adds terms such as "Eltern" ("parents") or "Personensorge" ("personal custody" or "care for the person") (cf. Table 7.3). However, the expansion terms that are added to query 1 result in a lower DCG. RF adds the term "Halt" ("stop"), which may be the reason why for this query RF scored lower than the original query without expansion using the SBERT embedding model. Note that this term is very close the German word "halten" ("to keep").
For one of the lower scoring queries, query 10, the RF-based technique added terms ("abs", "Massnahmen" ("measures"), "Behörde" ("authority")) that were not relevant since they did not relate to the concept of the query. However, the expansions for query 8, which scores the lowest overall with an average DCG of 5.304, are relevant additions to the query. We also observe that for queries 3-6, which were queries chosen to retrieve law documents of the labor law field, the ST-based expansion results in no significant improvements. Of these 4 queries, the terms that are added to query 4 (e.g., "Arbeitsentgelt" ("pay"), "Mindestarbeitsbedingung" ("minimum working condition")) seem to result in the best improvements as they are relevant to the query concept. In the case of query 6, no relevant terms for an expansion were found in the thesaurus. This could either be due to the reason that the domain thesaurus is too small, or because the original query word "HARTZ4" does not appear in the legal vocabulary of our corpus. In the case of query 8 the GT-based expansion seems to add the relevant terms "Sicherstellung" ("guarantee" or "assurance"), "Versicherung" ("insurance"), and "Erzeugnis" ("product"), but the expansion results in a decrease of its DCG.

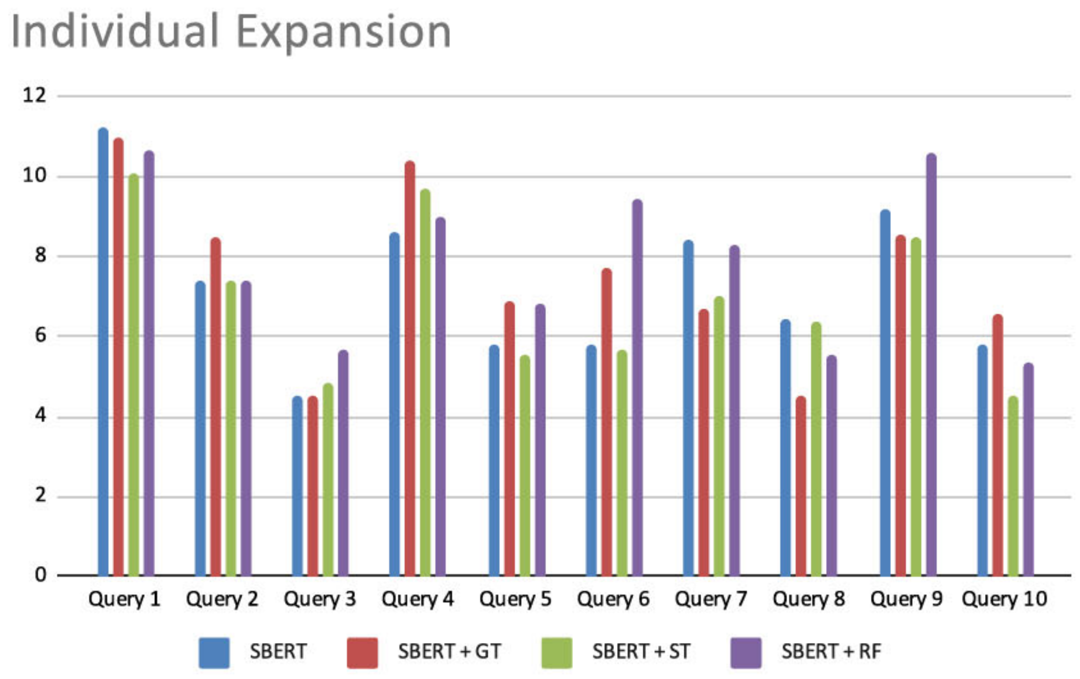
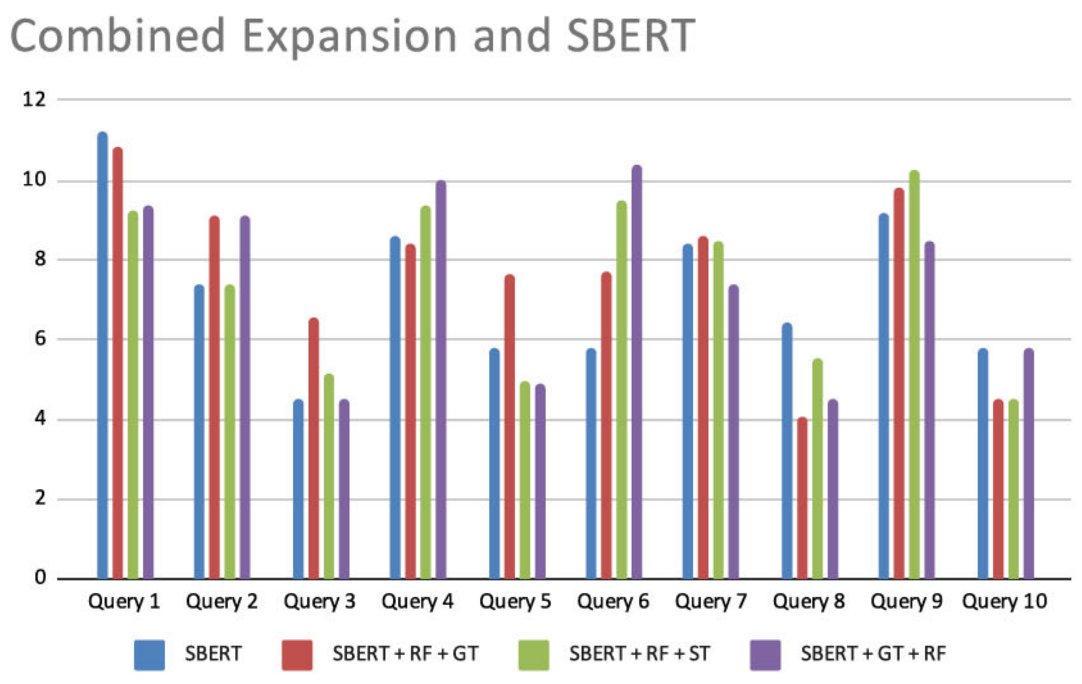
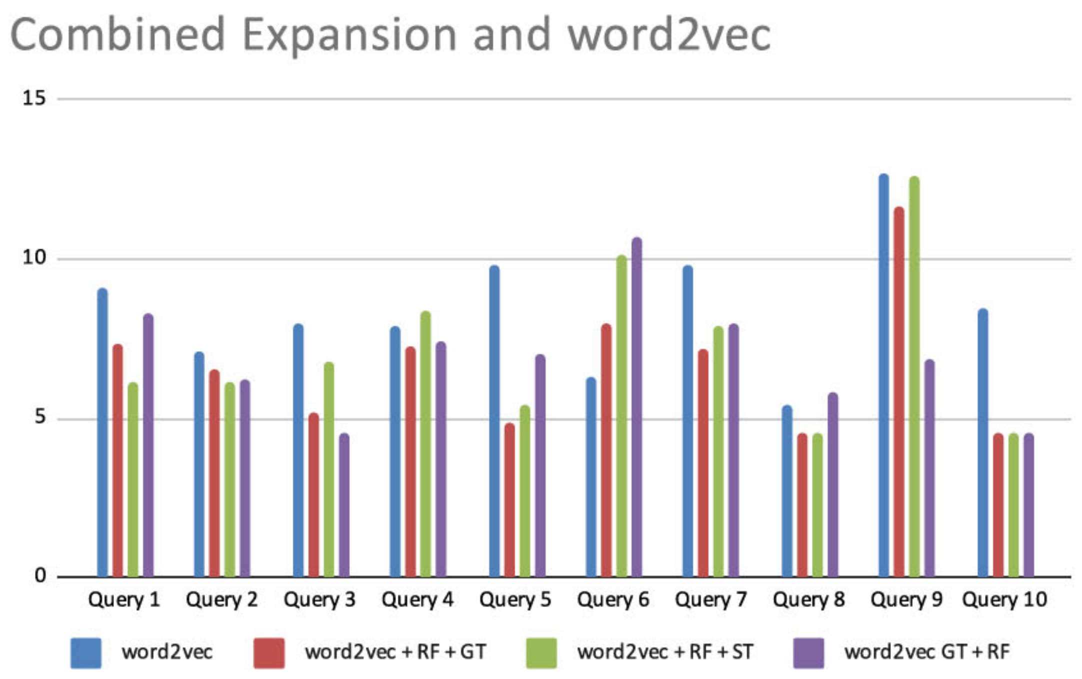


Figure 7.5: Discounted Cumulative Gain for each Query and Expansion Method



(a) Discounted Cumulative Gain of Combined Expansion Methods and SBERT



(b) Discounted Cumulative Gain of Combined Expansion Methods and word2vec

Figure 7.6: Discounted Cumulative Gain for each Query and Combined Expansion Methods

The results of each query for the combined expansion methods are visualized in Figure 7.6a and Figure 7.6b. Similarly to the individual expansion, we can observe that query 1 is worsened by all expansion combinations. Query 6 also stands out again, as all its expansion combinations result in an improvement.

The average DCG that is visualized in Figure 7.7 is calculated only over the 4 labor queries (queries 3-6). The average DCG of the labor queries and the SBERT embedding model is 6,175. The individual ST-based expansion scores an average DCG of 6,439 and the combined expansion of SBERT + RF + ST scores a DCG of 7,241. Both methods improve the original labor queries when the SBERT model is used for calculating the embeddings. Interestingly, the same can not be observed in case of the word2vec model. Here, the word2vec + RF + ST combination scores lower than the original word2vec queries on average.

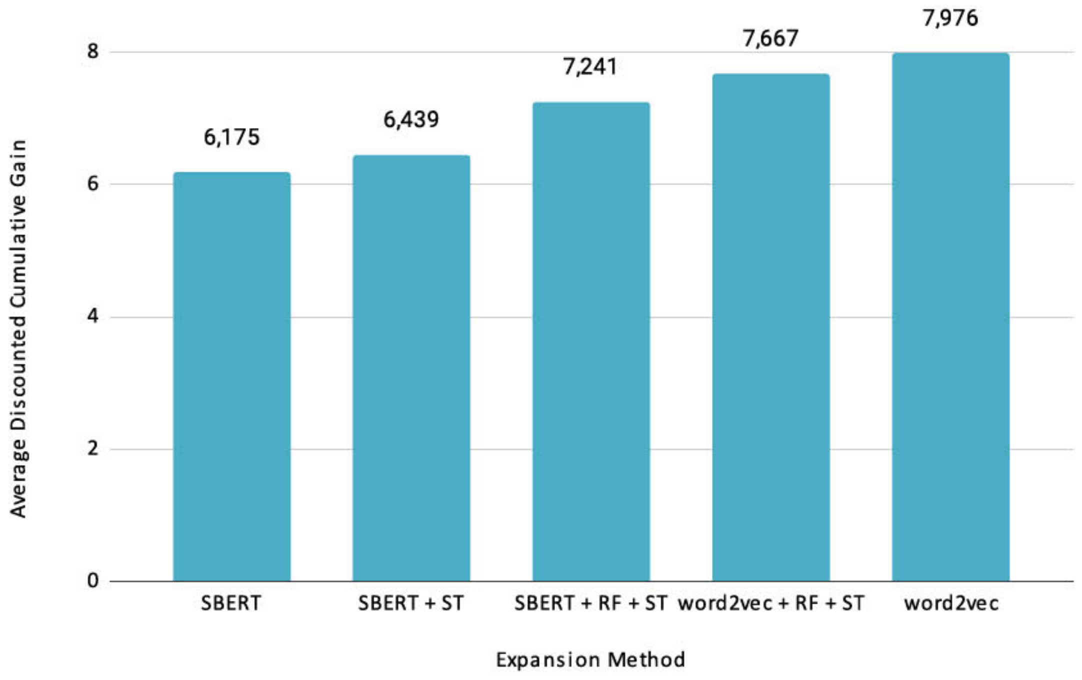


Figure 7.7: Average Discounted Cumulative Gain of the 4 Labor Queries

A single factor analysis of variance (Anova) was conducted to determine whether the differences between the embedding and expansion methods are statistically significant [Saw09]. A significance level of 0.05 was chosen. Our results show a F-statistic, which is the ratio of the variation between methods to the variation within a method, of 1.81 (rounded). Furthermore, the p-value, which corresponds to the F-statistic, is 0.0544 (rounded), which is higher than our chosen significance level and therefore indicates that our results show no evidence that there are statistically significant differences between the means of our methods. Note however, that this p-value is very close to our significance level, thus it could show a tendency towards the significance of our results.

Overall, it can be observed that there has not been a drastic change in terms of search effectiveness due to the expansion methods. In the case of the relevance feedback based method we set the parameter to 3 terms and in case of the thesaurus based expansion we set the parameter to 1 term. Nonetheless, we can observe that each expansion method adds different terms to the original query.
On the contrary, the choice of the embedding model seemed to have a greater impact on the results. In our evaluation, doc2vec scored the lowest out of all methods. We suspect this to be the case since all of our queries were keyword-based, with an average count of 3.1 words per query. The performance of the SBERT model could certainly be enhanced, if it was fine-tuned on our law data, which requires a labeled dataset of sentence pairs and their similarity, or if a model was chosen that was pre-trained for the German law domain. The SBERT model profited from every type of expansion, except for the individual specific thesaurus based expansion. Note that for now RF is also the individual method which consumes the most computational time in our system. Word2vec, which was trained entirely on our law data, was the embedding model that scored the highest. However, it did not profit from any combined expansion methods. As per queries, there were differences in overall retrieval performance. The best scoring queries were query 9 with an overall average DCG of 9.45 and query 1 with an average DCG of 8.994. In contrast, one of the lowest scoring queries was query 10 with an average DCG of 5.474.

7.3 Computational Time

Comparing the computational time of the individual techniques, the first notable difference is that the RF-based query expansion takes much longer than the other techniques with an average time of 28.514 seconds. We believe that the fact of sending individual query terms to Elasticsearch to extract the necessary parameters to calculate their wpq value, to be the reason for this. For future work, Elasticsearch should either be setup locally instead of using its cloud service to decrease the query time, or the necessary parameters to calculate the wpq values should be extracted through a different method. The ST-based expansion (0.029 seconds) on average takes up less time than the GT-based expansion (0.963 seconds), since the specific thesaurus is much smaller than the general thesaurus, thus less words are found in the thesaurus and fewer terms come into question as possible candidates for query expansion. Furthermore, the ST is maintained locally whereas the GT is accessed remotely via an API. Lemmatization takes up less time with an average of 0.095 seconds. As for the combined techniques, RF takes up most of the computing time. GT + RF took an average time of 30.546 seconds and ST + RF an average time of 29.352. The average time of the RF + GT

Table 7.2: Computational Time of each Technique in Seconds, rounded, "L" refers to "Lemmatization"

Query	L	GT	ST	RF	GT + RF	ST + RF	RF + GT	RF + ST
1	0.082	0.97	0.013	42.347	31.409	42.872	43.066	37.144
2	0.07	0.829	0.013	26.443	42.897	26.139	26.627	25.584
3	0.144	0.777	0.09	26.592	28.558	27.672	34.498	25.602
4	0.069	1.605	0.026	25.248	28.312	24.105	28.043	24.956
5	0.148	0.903	0.019	27.949	30.681	30.541	28.717	25.945
6	0.068	1.11	0.011	25.67	27.441	26.332	27.67	26.273
7	0.131	0.783	0.012	23.729	24.464	23.98	25.107	23.481
8	0.061	0.899	0.076	29.764	32.047	28.713	30.756	28.575
9	0.059	1.027	0.016	30.885	32.514	30.289	33.478	30.499
10	0.120	0.731	0.013	26.515	27.133	42.872	27.969	26.515
AVG	0.095	0.963	0.029	28.514	30.546	29.352	30.593	27.457

combination with 30.593 seconds is nearly equal to the sum of the individual techniques, when measurement inaccuracies are ignored. Note that through relevance feedback our query size is usually increased by 3. The average time of the RF + ST combination is with 27.457 seconds lower than the average time of the relevance feedback alone, which is probably due to measurement inaccuracies.

7.4 Findings and Interview

In this section we want to point out some findings which were discovered while conducting the evaluation and the interview with the law expert. We believed that our data contains all applied German laws in their latest form. This turned out not to be the case, as some laws were out of date because they were either already abolished or recently updated and changed (e.g., Documents for Query 4). This information was mentioned in the interview by the law expert after the evaluation was conducted. In that case, the law expert scored the corresponding retrieved document as irrelevant. This problem occurred for every method, thus it is expected that the results were negatively influenced. A further extension of this prototype should consider a data source that is regularly updated. Furthermore, in some cases the same document for the same query was assigned a different score. This was due to the expert not being presented a list of unique results, but the expert in some cases had to rate the same document for the same query.

As we have mentioned earlier, the queries and the corresponding information need were chosen and formulated in cooperation with the law expert. However, after the evaluation it became apparent that the information need was not captured broad enough for one of the queries. Namely: according to the law expert, the query ”Pflichten Mieter ausziehen” (”responsibilities tenant move out”) would have scored significantly higher, if the information need was also defined from the landlords point of view. For a future evaluation of our system, a higher focus should be set on defining adequate information needs for each query.

Furthermore, every type of document was treated equally by our system. This was done due to our prototype being intended to serve a broad group of users, but including users without any expertise in the field of German law. In our case, the expert rated entire laws lower than articles, if the article is part of the law and satisfies the information need of the query completely. For an evaluation conducted on a larger scale, it should be decided whether or not both articles and laws should be treated equally. After the evaluation was conducted it has come to our attention that for some headings the wrong content was shown due to a error in the program, which has since been fixed. The content that was shown belonged to an article of the same law and the law-expert did not mention this problem during the interview. Regarding the results of our evaluation, we can not observe that a document with the type heading was scored lower by the expert due to this problem.

An interview with the law expert was conducted before and after the evaluation. During the interview after the evaluation, the law expert was shown each final query after the AQE of individual methods. Some of the findings we want to highlight are:

- There is a demand for search systems in the German legal domain.
- ”Gesetze im Internet” was known to the law expert, but not used in practice for searching over laws. Law experts typically use other platforms (for a fee) to browse German laws (e.g., Beck-online). These platforms also provide other information on the laws beyond their content (e.g., Comments).
- The retrieval of entire laws can be useful, since it provides a better overview over relevant documents than the retrieval of single articles.

Table 7.3: Queries and Modification of each individual Technique

No.	Query	General Thesaurus	Specific Thesaurus	Relevance Feedback
1	Kraftfahrzeuge Zulassung erstmalig	Fahrzeug Genehmigung erstmals	-No Expansion-	Voraussetzung Halt zulassen
2	Luft chemische Verschmutzung	Pressluft Chemiker Verunreinigung	-No Expansion-	-No Expansion-
3	Kündigung widersprechen Fristen	kündigen verweigern Monat	Entlassung Ausschlussfrist	Beendigung gelten unwirksam
4	Gehalt Mindestlohn Höhe Anspruch	Vergütung Erhöhung Forderung	Arbeitsentgelt Mindestarbeitsbedingung Gratifikation Vorschuss	Arbeitnehmer Höhe Ansprüche
5	Arbeitsunfähigkeit Zeiträume anzeigen	Erwerbsunfähigkeit Monat Anzeige	Rente Abtretung	gelten vorlegen Höhe
6	arbeitslos HARTZ4 Geld	Arbeitslosengeld Vermögen	-No Expansion-	Arbeitslosigkeit Arbeitslosengeld Agentur
7	Pflichten Mieter ausziehen	Verpflichtung Vermieter ausziehen	-No Expansion-	Vermieter gegenüber unwirksam
8	Gewährleistung Garantie Produkte	Sicherstellung Versicherung Erzeugnis	-No Expansion-	Absicherung Massnahmen insbesondere
9	Kinder Sorgerecht elterliche Pflichten	Person Verpflichtung	-No Expansion-	Eltern Personensorge Elternteil
10	Versammlung Durchführung	Sitzung Ausführung	Personalratswahl	Behörde abs Massnahmen

8 Conclusion

Semantic search has shown the potential to enhance traditional IR. In this work we have presented an approach for the semantic search of German legal norms. Our goal was to create a prototype which allows non-expert users to effectively search law documents. To achieve this, we have first analyzed existing standards and ontologies for describing the legal domain, in addition to available thesauri and German legal information systems. State-of-the-art platforms and systems, which search over documents in the German legal domain, are available. However, they are often commercially offered or do not provide all law documents. Other platforms enable the retrieval of all legal documents, but only provide full text search. Furthermore, to our knowledge, there is no standard for semantically annotating the granular content (not the high level structure) of legal documents to allow semantic search. Semantic search in the German legal domain could benefit from a specific ontology.

Furthermore, we have explored and compared various embedding models for document and query modeling, and two query expansion techniques, namely a pseudo relevance feedback and a query expansion aided by a domain specific or general thesaurus, to find out to which extend they can be beneficial for our task. The embedding models were also used for word disambiguation and marking relevant documents for pseudo relevance feedback. We have used doc2vec, word2vec, and SBERT to calculate document and query embeddings for similarity calculation and thus to rank the documents by relevance. The word2vec algorithm has performed the best, followed by Okapi BM25 based ranking and the SBERT model. Doc2vec has shown poor performance for our task. However, when combined with our query expansion methods, SBERT has shown better improvements than word2vec. Furthermore, the SBERT model did not have the level of preparation as the word2vec model, as it was not trained on legal documents. Due to these reasons we believe that the SBERT model could outperform the word2vec model in future development.

Generally, the evaluation of our query expansion methods has shown smaller influence on the result. Overall, it can be said that the pseudo relevance feedback based query expansion, which is the most time consuming technique, has shown the best results, followed by the

general thesaurus based expansion.
The analysis of the individual queries has shown that while for some queries our expansion results in improvements, for others it seems to have the opposite effect. We believe that the effectiveness of the expansion is heavily reliant on each terms relatedness to the query's concept.

8.1 Outlook

One key part of a search system is its user interface, which was not developed for our system within this work. Additionally, users could have the option to chose the relevant terms for their search in the relevance feedback based query expansion and the thesaurus based expansion. Thus far, we have disregarded the reweighing of query words and expansion terms. Further development could enhance our system, since it has shown to be an effective technique in the past.

An extended evaluation of our system could be conducted. It could include more queries, by which our system is evaluated, and more experts, who score the results. An amount of around 50 queries and ten retrieved documents per query has shown to be suitable by TREC ("Text REtrieval Conference") [Bac+11]. Furthermore, the queries should be chosen without detailed knowledge about how they are handled by our system and how their results will be. Afterwards, the information need of each query should be determined in detail, and the retrieved documents should be presented to the experts in a unique set to avoid the problem of having different scores for the same document.
Following the extension of our prototype into a complete search system criteria such as user satisfaction gain greater importance. An evaluation could therefore also include the performance of our system when different parameters for the expansion are chosen. This includes the testing of different thresholds for determining suitable expansion terms as well as the total number of terms that are added. Another way to evaluate this feature is to give a group of expert and non-experts users time to freely use our system. By doing so, knowledge could be gained as to what expansion size yields to good results. Users could also be questioned about the experiences they had interacting with our system.
So far we have disregarded how our system performs in comparison to other German legal search systems. It could be beneficial to conduct such a comparison to gain knowledge about the effectiveness of our system.

Further research and development of the SBERT model could benefit search effectiveness. We have used a pre-trained multilingual SBERT model for calculating embeddings. A multilingual model is often less suited than a model that was solely trained on data in one language. Thus a model should be chosen that was only trained on German data, preferably legal data. One common step when using a pre-trained SBERT model is to fine-tune it on a specific task using a labeled dataset. However, since word2vec has shown to perform very well for creating law document embeddings, it could be used to automatically generate the parameters necessary for fine-tuning.

A different choice for the domain specific and general thesaurus should be made. Wolters Kluwer German labor law thesaurus is relatively small and not maintained anymore. OpenThesaurus has to be accessed through its API for every inquiry. A large scale evaluation could also include GermaNet as a general knowledge base for the German language. When extending this prototype, another aspect is to increase the efficiency of retrieving relevant results to avoid longer interaction times.

References

[AX19] F Almeida and G Xexéo. Word Embeddings: A Survey. 2019. DOI: 10.48550/ARXIV.1901.09069.

[Alt15] L Altamarino Sainz. Applying lexical knowledge to improve search quality for a German legal information database. Master's thesis, Technical University of Munich, Department of Informatics, Apr. 2015.

[Bac+11] A Baccini et al. How many performance measures to evaluate information retrieval systems? In: *Knowledge and Information Systems* 30(3):(Apr. 2011), 693–713. DOI: 10.1007/s10115-011-0391-7.

[Bar22] A Barbaresi. Simplemma. 2022. DOI: 10.5281/ZENODO.4673264.

[Bas] Basic Register of Thesauri, Ontologies & Classifications (BARTOC). *Wolters Kluwer Germany labor law Thesaurus*. https://bartoc.org/en/node/678 (visited on 10/15/2022).

[BBH16] H Bast, B Buchhold, and E Haussmann. Semantic Search on Text and Knowledge Bases. In: *Foundations and Trends® in Information Retrieval* 10:(Jan. 2016), 119–271. DOI: 10.1561/1500000032.

[BMS07] J Bhogal, A Macfarlane, and P Smith. A review of ontology based query expansion. In: *Information Processing &amp Management* 43(4):(July 2007), 866–886. DOI: 10.1016/j.ipm.2006.09.003.

[BMI12] A Bialecki, R Muir, and G Ingersoll. Apache Lucene 4. In: *OSIR@SIGIR*. 2012.

[Bif+05] A Bifet et al. An analysis of factors used in search engine ranking. In: *AIRWeb*. 2005, pp. 48–57.

[Bla+13] R Blanco et al. Repeatable and reliable semantic search evaluation. In: *Journal of Web Semantics* 21:(Aug. 2013), 14–29. DOI: 10.1016/j.websem.2013.05.005.

[BHW02] A Boer, R Hoekstra, and R Winkels. MetaLex: Legislation in XML. In:(Dec. 2002).

[Boj+17] P Bojanowski et al. Enriching Word Vectors with Subword Information. In: *Transactions of the Association for Computational Linguistics* 5:(Dec. 2017), 135–146. DOI: 10.1162/tacl_a_00051.

[BTW01] H Boley, S Tabet, and G Wagner. Design Rationale for RuleML: A Markup Language for Semantic Web Rules. In: *SWWS*. 2001.

[BüH98] J BüHLER. Jurivoc: Le nouveau thesaurus juridique suisse. In: *Terminologies nouvelles* (18):(1998), 36–38.

[Bun] Bundesministerium der Justiz. *Gesetze im Internet*. https://www.gesetze-im-internet.de/ (visited on 10/15/2022).

[Cao+08] H Cao et al. Context-aware query suggestion by mining click-through and session data. In: *Proceeding of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining - KDD 08*. ACM Press, 2008. DOI: 10.1145/1401890.1401995.

[Cel+06] I Celino et al. Squiggle: a Semantic Search Engine for Indexing and Retrieval of Multimedia Content. In: *SEMPS*. 2006.

[Cha+13] R Chauhan et al. Domain ontology based semantic search for efficient information retrieval through automatic query expansion. In: *2013 International Conference on Intelligent Systems and Signal Processing (ISSP)*. 2013, pp. 397–402. DOI: 10.1109/ISSP.2013.6526942.

[CAS16] H Christian, MP Agus, and D Suhartono. Single Document Automatic Text Summarization using Term Frequency-Inverse Document Frequency (TF-IDF). In: *ComTech: Computer, Mathematics and Engineering Applications* 7(4):(Dec. 2016), 285. DOI: 10.21512/comtech.v7i4.3746.

[Cona] S Conelly. *Practical BM25 - Part 1: How Shards Affect Relevance Scoring in Elasticsearch*. https://www.elastic.co/de/blog/practical-bm25-part-1-how-shards-affect-relevance-scoring-in-elasticsearch (visited on 10/15/2022).

[Conb] S Conelly. *Practical BM25 - Part 2: The BM25 Algorithm and its Variables*. https://www.elastic.co/de/blog/practical-bm25-part-2-the-bm25-algorithm-and-its-variables (visited on 10/15/2022).

[Dat10] J Datta. Ranking in Information Retrieval. In:(Apr. 2010).

[Dev+18] J Devlin et al. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. 2018. DOI: 10.48550/ARXIV.1810.04805.

[Die+14] M Diepenbroek et al. Towards an integrated biodiversity and ecological research data management and archiving platform: the German federation for the curation of biological data (GFBio). In: *Informatik 2014*. Ed. by E Plödereder et al. Bonn: Gesellschaft für Informatik eV., 2014, pp. 1711–1721.

[DGA17] J Dimyadi, G Governatori, and R Amor. Evaluating LegalDocML and LegalRuleML as a Standard for Sharing Normative Information in the AEC/FM Domain. In: *Lean and Computing in Construction Congress - Volume 1: Proceedings of the Joint Conference on Computing in Construction*. Heriot-Watt University, July 2017. DOI: 10.24928/jc3-2017/0012.

[Din+05] L Dini et al. Cross-lingual legal information retrieval using a WordNet architecture. In: *Proceedings of the 10th international conference on Artificial intelligence and law - ICAIL '05*. ACM Press, 2005. DOI: 10.1145/1165485.1165510.

[DP20] VR Doncel and EM Ponsoda. LYNX: Towards a Legal Knowledge Graph for Multilingual Europe. In: *Law in Context. A Socio-legal Journal* 37(1):(Dec. 2020), 175–178. DOI: 10.26826/law-in-context.v37i1.129.

[DHC08] H Dong, FK Hussain, and E Chang. A survey in semantic search technologies. In: *2008 2nd IEEE International Conference on Digital Ecosystems and Technologies*. IEEE, Feb. 2008. DOI: 10.1109/dest.2008.4635202.

[Eft96] EN Efthimiadis. Query Expansion. In: *Annual review of information science and technology (ARIST)* 31:(1996), 121–87.

[ES18] P Eigenmann and R Sivasothilingam. Semantische Suche von juristischen Dokumenten mittels Word Embedding und Netzwerkanalyse. In:(2018).

[Ela] Elasticsearch B.V. *Elasticsearch*. https://www.elastic.co/ (visited on 10/15/2022).

[Elb+15] KM Elbedweihy et al. An overview of semantic search evaluation initiatives. In: *Journal of Web Semantics* 30:(Jan. 2015), 82–105. DOI: 10.1016/j.websem.2014.10.001.

[Fel00] CD Fellbaum. WordNet : an electronic lexical database. In: *Language* 76:(2000), 706.

[Fra+19] T Francart et al. The European legislation identifier. In: *Knowledge of the Law in the Big Data Age* 317:(2019), 137–148.

[FJA05] G Fu, CB Jones, and AI Abdelmoty. Ontology-Based Spatial Query Expansion in Information Retrieval. In: *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, 2005, pp. 1466–1482. DOI: 10.1007/11575801_33.

[GGV17] F Gandon, G Governatori, and S Villata. Normative requirements as linked data. In: *JURIX 2017-The 30th international conference on Legal Knowledge and Information Systems*. 2017, pp. 1–10.

[Gao+04] J Gao et al. Dependence language model for information retrieval. In: *Proceedings of the 27th annual international conference on Research and development in information retrieval - SIGIR '04*. ACM Press, 2004. DOI: 10.1145/1008992.1009024.

[GJH07] JR Griffiths, F Johnson, and RJ Hartley. User satisfaction as a measure of system performance. In: *Journal of Librarianship and Information Science* 39(3):(Sept. 2007), 142–152. DOI: 10.1177/0961000607080417.

[Gru92] TR Gruber. *Ontolingua: A Mechanism to Support Portable Ontologies*. Tech. rep. 1992.

[GMM03] R Guha, R McCool, and E Miller. Semantic search. In: *Proceedings of the twelfth international conference on World Wide Web - WWW '03*. ACM Press, 2003. DOI: 10.1145/775152.775250.

[Hab+98] B Habert et al. Towards tokenization evaluation. In: *LREC*. 1998.

[HF97] B Hamp and H Feldweg. GermaNet - a Lexical-Semantic Net for German. In: *Workshop On Automatic Information Extraction And Building Of Lexical Semantic Resources For NLP Applications*. 1997.

[Har88] D Harman. Towards interactive query expansion. In: *Proceedings of the 11th annual international ACM SIGIR conference on Research and development in information retrieval - SIGIR '88*. ACM Press, 1988. DOI: 10.1145/62437.62469.

[He+09] Q He et al. Web Query Recommendation via Sequential Query Prediction. In: *2009 IEEE 25th International Conference on Data Engineering*. IEEE, Mar. 2009. DOI: 10.1109/icde.2009.71.

[HH10] V Henrich and EW Hinrichs. GernEdiT - The GermaNet Editing Tool. In: *LREC*. 2010.

[HJH07] M Hildebrand, van Jacco Ossenbruggen, and HL Hardman. An analysis of search-based user interaction on the semantic web. In: 2007.

[Hug] Hugging Face. *sentence-trasnformers/distiluse-base-multilingual-cased-v1*. https://huggingface.co/sentence-transformers/distiluse-base-multilingual-cased-v1 (visited on 10/15/2022).

[IS09] H Imran and A Sharan. Thesaurus and query expansion. In: *International journal of computer science & information Technology (IJCSIT)* 1(2):(2009), 89–97.

[Jär+08] K Järvelin et al. Discounted Cumulated Gain Based Evaluation of Multiple-Query IR Sessions. In: *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, 2008, pp. 4–15. DOI: 10.1007/978-3-540-78646-7_4.

[JBB14] V Jindal, S Bawa, and S Batra. A review of ranking approaches for semantic search on Web. In: *Information Processing &amp Management* 50(2):(Mar. 2014), 416–425. DOI: 10.1016/j.ipm.2013.10.004.

[Jiv11] AG Jivani. A Comparative Study of Stemming Algorithms Ms . In: 2011.

[KR16] J K. and J R. Stop-Word Removal Algorithm and its Implementation for Sanskrit Language. In: *International Journal of Computer Applications* 150(2):(Sept. 2016), 15–17. DOI: 10.5120/ijca2016911462.

[Kan+07] G Kanaan et al. A Comparison between Interactive and Automatic Query Expansion Applied on Arabic Language. In: *2007 Innovations in Information Technologies (IIT)*. IEEE, Nov. 2007. DOI: 10.1109/iit.2007.4430498.

[Kan+08] G Kanaan et al. Interactive and Automatic Query Expansion: A Comparative Study with an Application on Arabic. In: *American Journal of Applied Sciences* 5(11):(Nov. 2008), 1433–1436. DOI: 10.3844/ajassp.2008.1433.1436.

[Kas99] V Kashyap. Design and Creation of Ontologies for Environmental Information Retrieval. In: *Proc. of the 12th Workshop on*

Knowledge Acquisition, Modeling and Management. 1999.

[KG16] H Kaur and V Gupta. Indexing process insight and evaluation. In: *2016 International Conference on Inventive Computation Technologies (ICICT)*. Vol. 3. 2016, pp. 1–5. DOI: 10.1109/INVENTIVE.2016.7830087.

[KP+21] M Kharis, U Pairin, et al. How to Lemmatize German Words with NLP-Spacy Lemmatizer? In: *International Seminar on Language, Education, and Culture (ISoLEC 2021)*. Atlantis Press. 2021, pp. 189–193.

[KY00] A Kilgarriff and C Yallop. What's in a Thesaurus? In: *LREC*. 2000.

[Kor+11] R Korra et al. Performance evaluation of Multilingual Information Retrieval (MLIR) system over Information Retrieval (IR) system. In: *2011 International Conference on Recent Trends in Information Technology (ICRTIT)*. 2011, pp. 722–727. DOI: 10.1109/ICRTIT.2011.5972453.

[Kra97] RW van Kralingen. A Conceptual Frame-based Ontology for the Law. In: 1997.

[KSK16] S Kuzi, A Shtok, and O Kurland. Query Expansion Using Word Embeddings. In: *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*. ACM, Oct. 2016. DOI: 10.1145/2983323.2983876.

[Lan+17] J Landthaler et al. Extending Thesauri Using Word Embeddings and the Intersection Method. In: June 2017.

[Lan20] J Landthaler. Improving semantic search in the German legal domain with word Embeddings. PhD thesis. Technische Universität München, 2020.

[Lan] LanguageTooler GmbH. *OpenThesaurus*. https://www.openthesaurus.de/ (visited on 10/15/2022).

[LM14] QV Le and T Mikolov. Distributed Representations of Sentences and Documents. 2014. DOI: 10.48550/ARXIV.1405.4053.

[LCV19] V Leone, LD Caro, and S Villata. Taking stock of legal ontologies: a feature-based comparative analysis. In: *Artificial Intelligence and Law* 28(2):(June 2019), 207–235. DOI: 10.1007/s10506-019-09252-1.

[Liu+12] H Liu et al. BioLemmatizer: a lemmatization tool for morphological processing of biomedical text. In: *Journal of Biomedical Semantics* 3(1):(Apr. 2012). DOI: 10.1186/2041-1480-3-3.

[Liu+19] Y Liu et al. Multi-info Fusion Based Video Recommendation System. In: *Journal of Physics: Conference Series* 1229:(May 2019), 012010. DOI: 10.1088/1742-6596/1229/1/012010.

[Löf+17] F Löffler et al. Honey Bee Versus Apis Mellifera: A Semantic Search for Biological Data. In: *Lecture Notes in Computer Science*. Springer International Publishing, 2017, pp. 98–103. DOI: 10.1007/978-3-319-70407-4_19.

[Lop04] E Loper. NLTK: Building a Pedagogical Toolkit in Python. In: 2004.

[Lov68] JB Lovins. Development of a stemming algorithm. In: *Mech. Transl. Comput. Linguistics* 11:(1968), 22–31.

[Luk+02] RW Luk et al. A survey in indexing and searching XML documents. In: *Journal of the American Society for Information Science and Technology* 53(6):(2002), 415–437. DOI: 10.1002/asi.10056.

[Mae+03] A Maedche et al. An infrastructure for searching, reusing and evolving distributed ontologies. In: *Proceedings of the twelfth international conference on World Wide Web - WWW '03*. ACM Press, 2003. DOI: 10.1145/775152.775215.

[Man07] C Mangold. A survey and classification of semantic search approaches. In: *International Journal of Metadata, Semantics and Ontologies* 2(1):(2007), 23. DOI: 10.1504/ijmso.2007.015073.

[Mea+06] VD Mea et al. Measuring Retrieval Effectiveness with Average Distance Measure (ADM). In: 2006.

[Mik+13] T Mikolov et al. Efficient Estimation of Word Representations in Vector Space. 2013. DOI: 10.48550/ARXIV.1301.3781.

[Mil95] GA Miller. WordNet. In: *Communications of the ACM* 38(11):(Nov. 1995), 39–41. DOI: 10.1145/219717.219748.

[MH07] A Mnih and G Hinton. Three new graphical models for statistical language modelling. In: *Proceedings of the 24th international conference on Machine learning - ICML '07*. ACM Press, 2007. DOI: 10.1145/1273496.1273577.

[Nab05] D Naber. OpenThesaurus: ein offenes deutsches Wortnetz. In: *Sprachtechnologie, mobile Kommunikation und linguistische Ressourcen: Beiträge zur GLDV-Tagung, Bonn, Germany*:(2005), 422–433.

[NV03] R Navigli and P Velardi. An analysis of ontology-based query expansion strategies. In: *Proceedings of the 14th European Conference on Machine Learning, Workshop on Adaptive Text Extraction and Mining, Cavtat-Dubrovnik, Croatia*. 2003, pp. 42–49.

[NC18] VM Ngo and TH Cao. Discovering Latent Concepts and Exploiting Ontological Features for Semantic Text Search. 2018. DOI: 10.48550/ARXIV.1807.05578.

[NHO05] K Nilsson, H Hjelm, and H Oxhammar. SUiS–cross-language ontology-driven information retrieval in a restricted domain. In: *NODALIDA*. 2005.

[Now11] A Nowak. Semantic Search: Design and Implementation of a Vertical Search Service. In:(2011). DOI: 10.13140/RG.2.2.28220.95369.

[Ooi+15] J Ooi et al. A survey of query expansion, query suggestion and query refinement techniques. In: *2015 4th International Conference on Software Engineering and Computer Systems (ICSECS)*. IEEE, Aug. 2015. DOI: 10.1109/icsecs.2015.7333094.

[Pai90] CD Paice. Another stemmer. In: *ACM SIGIR Forum* 24(3):(Nov. 1990), 56–61. DOI: 10.1145/101306.101310.

[PS15] AR Pal and D Saha. Word Sense Disambiguation: A Survey. In: *International Journal of Control Theory and Computer Modeling* 5(3):(July 2015), 1–16. DOI: 10.5121/ijctcm.2015.5301.

[Pal+11] M Palmirani et al. LegalRuleML: XML-Based Rules and Norms. In: *Rule-Based Modeling and Computing on the Semantic Web*. Springer Berlin Heidelberg, 2011, pp. 298–312. DOI: 10.1007/978-3-642-24908-2_30.

[PSM14] J Pennington, R Socher, and C Manning. Glove: Global Vectors for Word Representation. In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, 2014. DOI: 10.3115/v1/d14-1162.

[Pet+18] ME Peters et al. Deep contextualized word representations. 2018. DOI: 10.48550/ARXIV.1802.05365.

[PLM04] J Plisson, N Lavrac, and D Mladenic. A rule based approach to word lemmatization. In: *Proceedings of IS04*. 2004.

[Por80] M Porter. An algorithm for suffix stripping. In: *Program* 14(3):(Mar. 1980), 130–137. DOI: 10.1108/eb046814.

[Por01] MF Porter. Snowball: A language for stemming algorithms. 2001.

[PU16] T Proisl and P Uhrig. SoMaJo: State-of-the-art tokenization for German web and social media texts. In: *Proceedings of the 10th Web as Corpus Workshop*. 2016, pp. 57–62.

[Pub] Publications Office of the European Union. *EuroVoc*. https://publications.europa.eu/en/web/eu-vocabularies/th-dataset/-/resource/dataset/eurovoc (visited on 10/15/2022).

[QF93] Y Qiu and HP Frei. Concept based query expansion. In: *Proceedings of the 16th annual international ACM SIGIR conference on Research and development in information retrieval - SIGIR '93*. ACM Press, 1993. DOI: 10.1145/160688.160713.

[RKA12] F Rahutomo, T Kitasuka, and M Aritsugi. Semantic Cosine Similarity. In: Oct. 2012.

[RP14] AS Ramkumar and B Poorna. Ontology Based Semantic Search: An Introduction and a Survey of Current Approaches. In: *2014 International Conference on Intelligent Computing Applications*. IEEE, Mar. 2014. DOI: 10.1109/icica.2014.82.

[Raz+19] MA Raza et al. A Taxonomy and Survey of Semantic Approaches for Query Expansion. In: *IEEE Access* 7:(2019), 17823–17833. DOI: 10.1109/access.2019.2894679.

[RG19] N Reimers and I Gurevych. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. In: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Association for Computational Linguistics, 2019. DOI: 10.18653/v1/d19-1410.

[Rod+18] P Rodríguez et al. Beyond one-hot encoding: Lower dimensional target embedding. In: *Image and Vision Computing* 75:(July 2018), 21–31. DOI: 10.1016/j.imavis.2018.04.004.

[Rut03] I Ruthven. Re-examining the potential effectiveness of interactive query expansion. In: *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval - SIGIR '03*. ACM Press, 2003. DOI: 10.1145/860435.860475.

[SWY75] G Salton, A Wong, and CS Yang. A vector space model for automatic indexing. In: *Communications of the ACM* 18(11):(Nov. 1975), 613–620. DOI: 10.1145/361219.361220.

[San94] M Sanderson. Word Sense Disambiguation and Information Retrieval. In: *SIGIR '94*. Springer London, 1994, pp. 142–151. DOI: 10.1007/978-1-4471-2099-5_15.

[SBK20] R Sastriani, ZKA Baizal, and DS Kusumo. Ontology-Based Semantic Search on Tourism Information Search System. In: *Indonesia Journal on Computing (Indo-JC)* Vol. 5 No. 1:(2020 2020). DOI: 10.34818/INDOJC.2020.5.1.397.

[Saw09] SF Sawyer. Analysis of Variance: The Fundamental Concepts. In: *Journal of Manual &amp Manipulative Therapy* 17(2):(Apr. 2009), 27E–38E. DOI: 10.1179/jmt.2009.17.2.27e.

[SO17] A Schoknecht and A Oberweis. LS3: Latent Semantic Analysis-based Similarity Search for Process Models. In: *Enterprise Modelling and Information Systems Architectures*:(2017), Vol 12 (2017). DOI: 10.18417/EMISA.12.2.

[SG07] E Schweighofer and A Geist. Legal Query Expansion using Ontologies and Relevance Feedback. In: *LOAIT*. 2007.

[Sid+14] G Sidorov et al. Soft Similarity and Soft Cosine Measure: Similarity of Features in Vector Space Model. In: *Computacion y Sistemas* 18(3):(Sept. 2014). DOI: 10.13053/cys-18-3-2043.

[SV04] A Sihvonen and P Vakkari. Subject knowledge improves interactive query expansion assisted by a thesaurus. In: *Journal of Documentation* 60(6):(Dec. 2004), 673–690. DOI: 10.1108/00220410410568151.

[SG01] A Singhal and I Google. Modern Information Retrieval: A Brief Overview. In: *IEEE Data Engineering Bulletin* 24:(Jan. 2001).

[SKO95] AF Smeaton, F Kelledy, and R O'Donnell. TREC-4 Experiments at Dublin City University: Thresholding Posting Lists, Query Expansion with WordNet and POS Tagging of Spanish. In: *TREC*. 1995.

[ST10] D Strasunskas and SL Tomassen. On variety of semantic search systems and their evaluation methods. In: *Proceedings of International Conference on Information Management* *and Evaluation, University of Cape Town, South Africa*. 2010, pp. 25–26.

[SDK] M Synak, M Dabrowski, and SR Kruk. Semantic Web and Ontologies. In: *Semantic Digital Libraries*. Springer Berlin Heidelberg, pp. 41–54. DOI: 10.1007/978-3-540-85434-0_3.

[Tec] Tech4Germany. *Rechtsinformationsportal*. https://alpha.rechtsinformationsportal.de/ (visited on 10/15/2022).

[Ure+10] V Uren et al. Reflections on five years of evaluating semantic search systems. In: *International Journal of Metadata, Semantics and Ontologies* 5(2):(2010), 87. DOI: 10.1504/ijmso.2010.033280.

[VB94] A Valente and J Breuker. A Functional Ontology of Law. In: *Arti intelligence and law* 7:(1994), 341–361.

[Val+96] A Valente et al. Towards principled core ontologies. In: 1996.

[Vas+17] A Vaswani et al. Attention is all you need. In: *Advances in neural information processing systems* 30:(2017).

[Ven+03] Venkatesh et al. User Acceptance of Information Technology: Toward a Unified View. In: *MIS Quarterly* 27(3):(2003), 425. DOI: 10.2307/30036540.

[VJP13] HJ Vibert, P Jouvelot, and B Pin. Legivoc - connecting law in a changing world. In: 2013.

[Voo94] EM Voorhees. Query Expansion using Lexical-Semantic Relations. In: *SIGIR '94*. Springer London, 1994, pp. 61–69. DOI: 10.1007/978-1-4471-2099-5_7.

[WBB08] W Wang, PM Barnaghi, and A Bargiela. Search with Meanings:An Overview of Semantic Search Systems. In: 2008.

[WF08] YD Wang and G Forgionne. Testing a decision-theoretic approach to the evaluation of information retrieval systems. In: *Journal of Information Science* 34(6):(May 2008), 861–876. DOI: 10.1177/0165551508091308.

[War19] C Wartena. A Probabilistic Morphology Model for German Lemmatization. In: Proceedings of the 15th Conference on Natural Language Processing (KONVENS 2019). 2019, pp. 40–49. DOI: 10.25968/opus-1527. http://nbn-resolving.de/urn:nbn:de:bsz:960-opus4-15271.

[WK92] JJ Webster and C Kit. Tokenization as the initial phase in NLP. In: *Proceedings of the 14th conference on Computational linguistics* -. Association for Computational Linguistics, 1992. DOI: 10.3115/992424.992434.

[Wol] Wolters Kluwer N.V. *Wolters Kluwer*. https://www.wolterskluwer.com/en (visited on 10/15/2022).

[XWD20] P Xia, S Wu, and BV Durme. Which *BERT? A Survey Organizing Contextualized Encoders. In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, 2020. DOI: 10.18653/v1/2020.emnlp-main.608.

[Xio+19] Z Xiong et al. New Generation Model of Word Vector Representation Based on CBOW or Skip-Gram. In: *Computers, Materials &amp Continua* 60(1):(2019), 259–273. DOI: 10.32604/cmc.2019.05155.

[Yeu19] CM Yeung. Effects of inserting domain vocabulary and fine-tuning BERT for German legal language. MA thesis. University of Twente, 2019.

[Zha+16] Z Zhang et al. Learning for Efficient Supervised Query Expansion via Two-stage Feature Selection. In: *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*. ACM, July 2016. DOI: 10.1145/2911451.2911539.

[ZN12] Z Zhong and HT Ng. Word Sense Disambiguation Improves Information Retrieval. In: *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers - Volume 1*. ACL '12. Jeju Island, Korea: Association for Computational Linguistics, 2012, pp. 273–282.

[Zho07] L Zhou. Ontology learning: state of the art and open issues. In: *Information Technology and Management* 8(3):(Mar. 2007), 241–252. DOI: 10.1007/s10799-007-0019-5.

[Zou+06] F Zou et al. Automatic Construction of Chinese Stop Word List. In: *Proceedings of the 5th WSEAS International Conference on Applied Computer Science*. ACOS'06. Hangzhou, China: World Scientific, Engineering Academy, and Society (WSEAS), 2006, pp. 1009–1014.

Acknowledgements

We thank ref. iur. Lars Tielo (Friedrich-Schiller University, Jena) for helping us with the evaluation of our search system and using his proficiency in the German law domain to interpret our results.

A Further Material

Table A.1: Queries and GermaNet Expansions

No.	Query	GermaNet
1	Kraftfahrzeuge Zulassung erstmalig	Personenkraftwagen Erlaubnis
2	Luft chemische Verschmutzung	Abluft Verunreinigung
3	Kündigung widersprechen Fristen	Beendigung abweichen Zeitpunkt
4	Gehalt Mindestlohn Höhe Anspruch	Arbeitsentgelt Mindestentgelt Wert Erstattungsanspruch
5	Arbeitsunfähigkeit Zeiträume anzeigen	Dienstunfähigkeit Dauer Anzeige
6	arbeitslos HARTZ4 Geld	Vermögen
7	Pflichten Mieter ausziehen	Pächter entledigen
8	Gewährleistung Garantie Produkte	Sicherheit Erzeugnis
9	Kinder Sorgerecht elterliche Pflichten	Verwandte Verpflichtung
10	Versammlung Durchführung	Sitzung Ausführung

Table A.2: Average Discounted Cumulative Gain for each Query and Method, rounded with Averages over Queries and Methods. Bold numbers - highest averages, emphasized numbers - highest scores within a query, "*" - highest scores within a method

Query	**Okapi BM25**	**doc2vec**	**word2vec**	**SBERT**	**SBERT + GT**	**SBERT + ST**	**SBERT + RF**
1	8.091	5.544 *	9.101	*11.240* *	10.988 *	10.082 *	10.622 *
2	8.361	5.174	7.064	7.383	8.458	7.383	7.383
3	6.275	4.544	*7.946*	4.544	4.544	4.845	5.674
4	7.707	5.044	7.905	8.591	*10.367*	9.681	8.983
5	*10.084* *	5.192	9.780	5.805	6.850	5.544	6.818
6	6.561	4.544	6.275	5.761	7.714	5.687	9.406
7	6.894	4.544	*9.793*	8.412	6.669	6.992	8.284
8	*6.696*	4.900	5.405	6.451	4.544	6.396	5.544
9	9.182	4.544	*12.711* *	9.165	8.550	8.454	10.582
10	*8.476*	4.544	8.419	5.805	6.544	4.544	5.317
AVG	7.833	4.857	**8.440**	7.316	7.523	6.961	7.861

Table A.3: Average Discounted Cumulative Gain for each Query and Method, rounded with Averages over Queries and Methods (continuation). Bold numbers - highest averages, emphasized numbers - highest scores within a query, "*" - highest scores within a method

SBERT + RF + GT	SBERT + RF + ST	word2vec + RF + GT	word2vec + RF + ST	SBERT + GT + RF	word2vec GT + RF	AVG
10.873 *	9.276	7.305	6.109	9.378	8.310	8.994
9.089	7.383	6.516	6.134	9.089	6.201	7.355
6.576	5.174	5.189	6.762	4.544	4.544	5.474
8.415	9.353	7.277	8.341	10.037	7.407	8.393
7.674	4.930	4.877	5.430	4.877	7.008	6.528
7.710	9.505	7.984	10.136	10.413 *	*10.707* *	7.877
8.609	8.469	7.196	7.885	7.403	7.999	7.627
4.044	5.44	4.544	4.544	4.544	5.805	5.304
9.811	10.293 *	11.617 *	12.599 *	8.471	6.868	**9.450**
4.544	4.544	4.544	4.544	5.805	4.544	5.552
7.734	7.447	6.705	7.248	7.456	6.939	

Ehrenwortliche Erklärung

Ich erkläre, dass ich die vorliegende Arbeit selbstständig und nur unter Verwendung der angegebenen Quellen und Hilfsmittel angefertigt habe.

Seitens des Verfassers bestehen keine Einwände die vorliegende Bachelorarbeit für die öffentliche Benutzung im Universitätsarchiv zur Verfügung zu stellen.

Jena, den 04.12.2022

Friedrich Tydecks