Multi-document Analysis

Semantic Analysis of Large Text Corpora
Beyond Topic Modeling

Dissertation

zur Erlangung des Doktorgrades der Naturwissenschaften

vorgelegt beim Fachbereich (12) Informatik und Mathematik
der Johann Wolfgang Goethe-Universität
in Frankfurt am Main

von
Tolga Uslu
aus Frankfurt am Main

Frankfurt, 2019
(D 30)

vom Fachbereich (12) Informatik und Mathematik der

Johann Wolfgang Goethe - Universität als Dissertation angenommen.

Dekan: Prof. Dr.-Ing. Lars Hedrich

Gutachter: Prof. Dr. Alexander Mehler und Prof. Dr. Visvanathan Ramesh

Datum der Disputation:

Acknowledgement

After many years of intensive work, it is finished – my dissertation. At this point I would like to express my special thanks to the following persons, who always supported me during the completion of this work. First of all, I would like to thank Prof. Dr. Alexander Mehler, my doctoral supervisor, for the mentoring of this work. His friendly help, his manifold ideas and his critical thinking have always guided me in the right direction. He advised me not only in the field of computer science, but also enriched me on a personal level with his worldly wisdom. My sincere gratitude also goes to Prof. Dr. Visvanathan Ramesh for taking his precious time and his helpful support as second examiner. I would also like to thank my colleagues and former fellow students who have accompanied me in recent years with their valuable advice and discussions on new and interesting topics. My family takes a prominent position in every respect, since without their loving care this work would not have become the work it is today.

Contents

1 Introduction

1.1 Motivation

More and more data is freely available online these days. Social networks, blogs, forums and online encyclopedias, to name just a few, contribute to this. Many of these data are available in unstructured form as text. And although a lot of this data is impure and contains spelling mistakes, there are still resources available to work with. One of this resources is Wikipedia[1]. Wikipedia is an online encyclopedia that offers free content, created by the joint work of its users (Mehler, Rüdiger Gleim, Lücking, et al. 2018; Stegbauer 2009). Millions of users are working on the further development and improvement of this encyclopedia and this is what makes Wikipedia so interesting in terms of natural language processing (NLP). Over 50 million articles have already been written[2] and more than 10,000 new articles are added daily[3]. Never before have we had such a privilege in NLP to collect so much digital content which is also verified by millions of users in terms of correctness and spelling mistakes. This enables the main goal of this dissertation, namely to analyze documents and automatically comprehend its content.

But in order to analyze documents according to their semantic content, there are several tasks that have to be accomplished beforehand. Starting with the pre-processing step, where unstructured texts are converted into machine-readable formats. In pre-processing, the text is initially broken down into its basic components such as words and sentences. Subsequently, further pre-processing steps on morphological, syntactic and semantic level can be treated.

[1] wikipedia.org

[2] statista.com/statistik/daten/studie/195081/umfrage/anzahl-der-artikel-auf-wikipedia-weltweit

[3] statista.com/statistik/daten/studie/195096/umfrage/anzahl-neuer-artikel-pro-tag-bei-wikipedia

This includes, for example, the analysis of the part of speech, lemma and grammar information of words. All this information can be used to address and solve other NLP tasks. For example, Sun et al. (2014) have shown the influence of pre-processing on the thematic analysis. Peng and Dredze (2016) have shown that they can improve *Named Entity Recognition* using pre-processing. Text classification in the medical field also benefits from pre-processing (P.-H. Chen et al. 2018). Another type of pre-processing at semantic level is word sense disambiguation (WSD). To understand the semantic content of a document, it is helpful to understand the meaning behind each word beforehand (Guo and Diab 2011; H. Song and Yao 2009; Plaza, Stevenson, and Díaz 2010). WSD addresses this problem and utilizes the context of a word to determine its meaning.

After analyzing the semantics at word level, the analysis of semantics at document level can be performed. This requires a semantic model to represent the text. In this dissertation we have dealt with two different models, which are discussed in more detail later in this chapter. All the studies and discoveries included in this dissertation became possible due to the ever-growing number of texts available online. But despite all the benefits gained from this data, there are also challenges that need to be mastered. Especially one of the biggest advantages brings one of the most difficult challenges – big data. Having a lot of data is a blessing in machine learning, as methods and algorithms can be trained much better for a particular task. But not every algorithm is designed to be applied to large amounts of data efficiently. In addition, many shared NLP tasks have the disadvantage that the evaluations contain very little hand-annotated data for training and testing. Although these tasks are good for comparison with the competition, they do not provide a solid basis for achieving good results in real life scenarios, especially when facing big data. Solving a problem such as WSD for a small data set containing about 100 ambiguous words will not work if applied to a larger data set where thousands of words are ambiguous. On the one hand, because most words of the large data set remain unknown and on the other hand, as the algorithms are mostly optimized for the small data sets and cannot keep up with a larger number of documents in terms of runtime. Therefore, the development of the models in this dissertation also focused on the efficient handling of big data. A resource as large as Wikipedia provided a good foundation. By using user annotated

information from a Wikipedia article and other data sources associated with Wikipedia, it is possible to create large corpora containing semantic information. Thus, we were able to generate data sets for different NLP tasks. This allowed the automatic analysis of a text according to its semantics, which has been addressed in this dissertation.

1.2 Focus & Goals

In this chapter, we introduce the focus and goals of this dissertation and discuss the main idea and the content of the included articles. In addition, we will describe the relationship between the individual articles and how the success of each article depends on each other. However, before we discuss the content-related topics, we will first explain the principles, which have been highly focused in this dissertation.

1.2.1 Principles

Besides the main topic of this dissertation, namely the automatic analysis of documents on a semantic level, the following principles are a special focus of this thesis.

Efficiency

As already mentioned, many NLP tasks usually contain smaller datasets. This usually encourages the participants to develop complex systems which are then optimized for this data set and achieve good results. However, these systems are then usually limited to these small datasets and are no longer applicable for big data due to the time complexity, unless days and weeks are available to wait for the results. In machine learning, however, the principle of trial and error is often used to solve certain tasks. A first approach is implemented and applied for a specific task. Subsequently, the errors of the approach are analyzed. Based on these errors, the approach is adjusted accordingly to avoid them. However, waiting days or weeks for each iteration, would require months of experiments to optimize for a specific task. Therefore, an important principle of this dissertation is the efficiency of the developed systems and the manageability of big data.

Language independence

Not only the size of the datasets, but also the number of languages involved in these shared tasks are limited. We decided to take advantage of the sumptuous amount of data available on the Internet and develop the tools language independent. Wikipedia and Wikidata were of great help in realizing this principle. We were able to utilize the multilingual content written by millions of users, which are also interconnected across the languages. Thus we were able to create language-independent corpora for different tasks and train our models language-independently for all supported Wikipedia languages.

Applicability

The next principle concerns the applicability of the models. We did not want to create models or tools that only exist on paper or can only be used by the people creating it. Thus it is another important principle to make the developed approaches available online and applicable. This has been achieved by creating websites for the projects, allowing the approaches to be tested.

By adhering to these principles, it was possible to make the developed systems accessible to all. Due to the *efficiency*, the size of the data sets is no longer an obstacle, even on outdated computers. Due to the *language independence*, even low-resource languages can be addressed and the principle of *applicability* facilitates these tools to be tested and integrated. Now that the principles of the dissertation have been explained, the content-related topics can be discussed.

1.2.2 Document analysis

One of the main goals of this dissertation is to analyze the semantic content of multiple documents – preferably a large amount of documents. At the same time, the implementation of the principles set out in the section before is of great importance. Every day, huge amounts of unstructured text appear, affecting all sorts of business areas. According to Oussous et al. (2018) the number of available data doubles every two years. This includes emails, so-

cial media posts, user feedback, ratings and much more. When it comes to analyzing and organizing all this data, automatic text analysis provides a big advantage. This allows large amounts of data to be explored and frequently used patterns to be identified quickly, easily and in a scalable manner. The analysis of the semantics of a text involves various methods and emphases (Brinker et al. 1988; Rastier 1974). *Text isotopy* for example concerns correlations between the sentences of a text from a semantic point of view (Greimas 1971). The emphasis here, however, is on analyzing the topics of multiple documents or corpora rather than the thematic progression of sentences, which is influenced by coherence and cohesion. There is many research that has already dealt with the semantic analysis of documents. Mimno (2012) for example used topic models on an archive of over 100 years of accumulated journal publications. As a result, this large collection could be organized automatically and scholars had the opportunity to discover even the older studies. Griffiths and Steyvers (2004) and Lamba and Madhusudhan (2019) analyzed trending scientific topics using articles of different journals. There has also been some research analyzing the distribution of topics over time from different newspapers (Newman and Block 2006; T.-I. Yang, Torget, and Mihalcea 2011). Vincent (2007) worked on document understanding by analyzing Google books. This allowed to cluster these books and make them searchable. Clearly, there are numerous possible applications for automatic semantic analysis of documents. Many of these methods use unsupervised topic detection methods, which have some disadvantages. Later in this chapter, we will discuss pros and cons of existing topic models and show the different types of semantic models we addressed in this dissertation. However, before we analyze documents at the semantic level, we must first perform a few steps beforehand, starting with the pre-processing.

1.2.3 Pre-processing

In order to automatically analyze documents, pre-processing is of utmost importance. By reading a text, humans can very quickly grasp the content, meaning and context of a text. We must remember that words, sentences and texts are nothing but a collection of characters to a computer. In this context we often speak of unstructured data. The purpose of pre-processing

is to structure this data in order to automatically process the text for a range of purposes. Therefore, at the beginning of this dissertation we address the automatic pre-processing of texts.

The use of large amounts of data for machine learning requires careful pre-processing, which is fundamental to the accuracy of the result. Thus, the more accurate the information that can be extracted from a text, the more precise the content of the text can be determined. Starting with the recognition of words and sentences, up to the morphological analysis of words, all this information can help in the further processing. In addition, the number of NLP tools and the number of freely accessible resources is growing rapidly, making it difficult to keep track.

For this purpose we have developed `TextImager`, a UIMA[4]-based system for NLP. UIMA is a framework for managing streams of data between components (Ferrucci and Lally 2004). It offers standardized interfaces for the creation of these components, which can be executed individually or in a group in a pipeline structure. This allows TextImager to offer a wide range of NLP tools for text pre-processing and analysis. Furthermore, a big advantage of TextImager is that it combines different NLP methods for different tasks and makes them interoperable. This also enables the principle of language independence, since we can reconcile NLP procedures from all over the world. TextImager currently supports more than 10 languages, including over 10 methods each for high-resource languages such as English, German, Spanish and French. Another advantage is that TextImager is not dependent on a central repository. It can be distributed across multiple servers, allowing developers to set up their own TextImager server and distribute their own NLP tools within the TextImager ecosystem. Furthermore, the multi-server system enables the orchestration of NLP-tools on even larger amounts of data. By distributing the services to multiple servers, TextImager complies with the principle of efficiency and handling of big data. In order to fulfill the principle of applicability, TextImager also visualizes the extracted information in a clearly arranged manner via a web interface in order to give the user a quick overview of all extracted data. Furthermore, these visualizations are interactive and enable to control the navigation in the input texts. Conversely, it is also possible to interact with the text to create context-sensitive visualizations. This facil-

[4]uima.apache.org

itates navigation between text and visualization and the analysis of specific text phrases.

There are already frameworks in the field of NLP. `DKPro` (Eckart de Castilho and Gurevych 2014), `GATE` (Cunningham et al. 2011) and `WebLicht` (M. Hinrichs, Zastrow, and E. W. Hinrichs 2010) just to name a few. Their aim is to provide different NLP methods in a pipeline system. `AllenNLP` (Gardner et al. 2018), `CoreNLP` (C. D. Manning et al. 2014) and `SpaCy` (Honnibal and Montani 2017) are frameworks providing NLP-tools. Their goal is to solve certain problems in the field of natural language and to provide models for different languages. In addition to text analysis, `Voyant Tools` (Sinclair and Rockwell 2012) and `RapidMiner` (M. Hofmann and Klinkenberg 2013) offer visualizations similar to TextImager. However, all the above mentioned tools have some shortcomings and do not combine the benefits of each. Some of these frameworks are not cluster-based or service-based, making it difficult to distribute large amounts of data. Some do not offer a variety of NLP tools for different linguistic processing levels. Some do not provide a user-friendly interface for using these NLP tools. The latter is particularly important, as even non-IT-savvy people can work with these NLP methods, enabling even interdisciplinary research questions to be addressed. TextImager incorporates these functions and advantages and provides the ability to extract many kinds of information.

In order to have a better understanding of the collected data, we have combined the NLP procedures of TextImager with a tool for statistical data analysis – R (R Development Core Team 2008). R is an open source software and a programming language designed to analyze statistical data and generate graphics. The main package repository of the R-project called CRAN[5] consists of about 15,000 packages containing functions for almost every statistical problem. Some of these packages also include NLP functions, making them better interoperable with TextImager. This allows, for example, statistical analyses of linguistic feature distributions, as well as stylometric analysis of documents based on pre-processed features like syntactic information. The combination of these two systems offers a wide range of applications and enhances the capabilities of TextImager by the advantages of R. Therefore TextImager forms the basis for all methods and frameworks developed in this

[5] `cran.r-project.org`

dissertation, since it handles the pre-processing and subsequently incorporates the developed tools and makes them available in a distributed manner throughout its ecosystem.

1.2.4 Classification

In order to understand and analyze documents, classifications are of fundamental importance, since any NLP task can be represented as a kind of classification problem (Biemann et al. 2014). In text classification, tags or categories are assigned to the text according to its content. In NLP it has a variety of applications – at word level for instance part of speech tagging or lemmatization, at document level, tasks such as sentiment, spam or topic detection. Automatic text classifications are used to structure textual data quickly and cost-effectively, to improve and automate processes. Since this dissertation focuses on document analysis, we have primarily dealt with classification at document level. For classifications at word level, such as POS and lemma tagging, which are often used for pre-processing, we have applied established methods available in TextImager. Pre-processing is essential for automatic text classification, as it enables text structuring and the extraction of new text features. This provides the classifier with more information as input, which might improve the classification quality. Furthermore, in recent years one of the biggest trends in NLP has been the use of word embeddings, i.e. word vectors whose relative similarities correlate with their semantic similarity. Mikolov, K. Chen, et al. (2013) has published one of the most popular method for creating word embeddings – `word2vec`. This allows the mapping of words into a semantic space and thus helps the computer to understand the semantics behind the sequences of characters. Many classification tasks have been improved by using this semantic space. `fastText` (Joulin et al. 2016), for example, is a classification framework that supports word embeddings and classifies texts. fastText is based on an artificial neural network with a single hidden layer. The focus of fastText's architecture is on the speed of the classification. Since this fits perfectly with our principle of efficiency and applicability to big data, some of the classification algorithms we developed are based on this architecture.

However, the frequent use of TextImager and the analysis and comparison of its

visualizations raised the question, whether it is possible to classify documents by using only the visual characteristics. This is where `text2voronoi` comes in. In contrast to conventional classification approaches, we have developed a special document visualization technique and use its visual features for automatic classification. First of all, we extracted linguistic features of the text to be visualized. Here we extracted morphological and morphosyntactic information and visualized them using a Voronoi diagram. For the positioning of the Voronoi cells, we have calculated embeddings for each morphological unit. The resulting visual features served as input for the classifier. text2voronoi was applied to classify diseases using data from physician-patient interviews. We have shown that our image-driven classifier can successfully distinguish between epileptic and dissociative disorders. Another advantage of text2voronoi is the interactive visualization, which allows subsequent analysis of the classification and the underlying visualization. We enable the analysis of the input data for the classifier. Otherwise a series of vectors and matrices would need to be observed. This is a first step towards solving the problem that often underlies machine learning – black box.

Classifications can also be used to determine the semantic content of documents. In the next section, we will explain how we used automatic text classification to analyze the topic distribution of documents.

1.2.5 Semantic analysis

Word sense disambiguation

Before we dive into the semantic analysis of documents, an intermediate step is required. In order to understand the content of a document, it is helpful to understand the meaning of each word. The most basic approach is to create a one-level coded vector where each occurring word represents a dimension, whereby 1 indicates the existences and 0 the non-existence of the word. Here the computer receives only the information about the occurrence of words. This input could be enhanced with more information if, for example, the term frequency–inverse document frequency (TFIDF) values were calculated. Word embeddings also help to enrich the information of the input. However, these approaches have the disadvantage that they do not resolve ambiguous words

and map all senses to a single word or vector.

Ambiguous words are particularly important for semantic analysis, because otherwise the computer does not know the meaning of the word and is therefore less likely to grasp the content of the document. Word sense disambiguation (WSD) addresses the elimination of ambiguities of linguistic expressions by utilizing its context. There are several attempts to solve WSD, which we will discuss in more detail in Chapter 9, but these methods often have the disadvantage that they have been trained and optimized on only small amounts of data. An application to larger amounts of data, like the ones we examined, was not feasible with existing methods.

To this end, we have developed `fastSense`, which is designed to solve the problem of WSD for big data. Here we have made use of the universal approximation theorem:

Universal approximation theorem (Csáji 2001): Let $\varphi(.)$ be an arbitrary activation function. Let $X \subseteq R^m$ and X is compact. The space of continuous functions on X is denoted by $C(X)$. Then $\forall f \in C(X), \forall \varepsilon > 0 :$ $\exists n \in N, a_{ij}, b_i, w_i \in R, i \in \{1...n\}, j \in \{1...m\} :$

$$(A_n f)(x_1, ..., x_m) = \sum_{i=1}^{n} w_i \varphi(\sum_{j=1}^{m} a_{ij} x_j + b_i) \tag{1.1}$$

as an approximation of the function $f(.)$; that is

$$\| f - A_n f \| < \epsilon \tag{1.2}$$

The universal approximation theorem states, that a feedforward network with a linear output layer and at least one hidden layer with certain activation functions (sigmoid and ReLUs included) is able to approximate any continuous function defined on a compact subset of R^n. With this in mind, we have used a highly efficient artificial neural network that uses only a single hidden layer and trained it on the disambiguation scheme of the entire Wikipedia. The architecture of fastSense was also motivated by fastText (Joulin et al. 2016), which is known for its time efficiency. However, since fastText was developed for text classification, we had to make some modifications to make the

model applicable to WSD. Thus we were able to apply fastSense to a disambiguation corpus with over 50,000,000 training and test units. To create such a large corpus, we used Wikipedia's disambiguation pages and link structure to match words with their corresponding senses. This results in a corpus of 221,965 ambiguous words with 825,179 meanings. Notwithstanding the high time efficiency, we hardly had to suffer any quality losses. We achieved an F-score of over 80% in our big data corpus and furthermore we were able to keep up with our competitors in Senseval and SemEval tasks. Since our corpus is based on Wikipedia, fastSense could be applied to all supported Wikipedia languages.

Topic models

Now that we are able to dissolve words into meanings, we can focus on semantic analysis at a higher level, namely of documents. As mentioned before, the emphasis of the semantic analysis in this dissertation is on the thematic analysis of the documents. Document topic detection is about abstracting the content and finding out the core information of a document that dominates all semantic information in the text (Kallmeyer et al. 1974). According to Brinker et al. (1988) text topic is the core of the text content, which for example causes text coherence or creates a context by arousing expectations in the reader.

One of the most popular methods for detecting topics, are for example Latent Dirichlet Allocation (LDA, David M Blei, A. Y. Ng, and Jordan (2003)), Latent Semantic Analysis (LSA, Deerwester et al. (1990)) or the probabilistic version of LSA (PLSA, T. Hofmann (1999)). These are unsupervised topic models, as they are designed to identify patterns and structures without taking any form of prior knowledge into account during the analysis. LSA is one of the fundamental techniques in topic modelling, as it is a statistical method designed to vectorize the meaning of words and texts and allows automatic measurement of the similarity of the content of words and texts. However, the disadvantages include the limited representation and analysis of syntactic structures, which are only considered by the inflections of the words. Moreover, the embeddings are difficult to interpret as the topics are unknown. pLSA tries to solve this problem by using a probabilistic method rather than the *singular value de-*

composition as in the case of LSA. The main idea is to assign a distribution of latent topics to each document and a collection of words to each topic. However, pLSA is not a generative model for assigning topics to documents, which makes it difficult to categorize unknown documents. In addition, the amount of model parameters increases linearly with the size of the corpus, which can lead to serious problems with overfitting (David M Blei, A. Y. Ng, and Jordan 2003). LDA tackles these problems and tries to solve them with a few amendments. LDA is a Bayesian version of pLSA and is therefore suitable for a better generalization. Thus LDA also works well on unseen documents.

Regardless of their great applications, a central problem of this approach is that the identified topics are not directly labeled. Since these are unsupervised topic models, the identified topics are not always comprehensible and distinguishable, making it difficult to interpret the resulting topic distributions (J. Chang, Gerrish, et al. 2009). There have been studies that have dealt with supervised topic detection, for example sLDA (Mcauliffe and David M Blei 2008). The disadvantage of these models is often that they become slow and inefficient as the data set and topic categories grow, requiring more and more computers to work in parallel (Y. Li, W.-Z. Song, and B. Yang 2018). However, this affects the usability of the models.

Besides the methodology of topic detection, the underlying topic model plays an important role. Hereby we differentiate between closed topic models (CTM) and open topic models (OTM) (Mehler and Waltinger 2009). The CTMs are characterized by a predefined classification scheme. This classification scheme is often defined by a small number of experts and barely changes over time. The advantage of this model, however, is that it produces the same results when applied repeatedly and thus becomes comparable. In OTMs, categories are not listed in advance. OTMs focus on the topics of an constantly expanding thematic universe, which is composed and expanded by the collaboration of multitudes of people.

A good example for CTMs is the *Dewey Decimal Classification* (DDC, Dewey (1876)). DDC is an internationally accepted classification system that organizes literature in libraries according to its content. The classification system is hierarchical, starting with 10 main categories and containing up to 10 subcategories for each level, resolving into 99 classes at the second level (DDC 040

is unset) and 915 classes at the third level.

For example, the main category 900 stands for *History and geography.* It is divided into 10 subcategories, for example 950 is representing *History of Asia.* Once again, this class is divided into 10 subcategories. On this level, for example, 952 stands for *History of Japan.*

There are several studies addressing the automatic classification of DDC (Golub, Hagelbäck, and Ardö 2018; Brück, Eger, and Mehler 2016; Lösch et al. 2011; Mehler and Waltinger 2009). But they used classification methods that could not be applied to larger amounts of data, furthermore the amount of available training and test data was small. This is often the case because DDC classes usually occur in connection with books and these are not easily accessible. Thus, in these cases, the title is used as input, even though it does not provide quite as much information.

However, we have found a way to avoid this barrier. By using Wikipedia, Wikidata and the *Gemeinsame Normdatei* (GND), it is possible to create a large language independent DDC-corpus. By connecting the three platforms, we are able to automatically provide Wikipedia articles with DDC information obtained from the GND. By using the language links available in Wikidata, this corpus is also adaptable to all supported Wikipedia languages.

By implementing an efficient neural network based classifier, we have created a language-independent topic model based on DDC called `text2ddc`, which is capable of processing large amounts of data. Classifiers are usually developed to classify or categorize a document into a specific target class. text2ddc, however, utilizes the softmax function to transform the output layer to the value range between 0 and 1. The softmax function is defined by (Goodfellow, Bengio, and Courville 2016):

$$softmax(z)_i = \frac{exp(z_i)}{\sum_j exp(z_j)} \tag{1.3}$$

Furthermore, the softmax function has the advantage that the sum of the output neurons equals 1. Thus text2ddc calculates a probability for each DDC class, resulting in a topic distribution of the documents, similar to LDA. The advantage of this topic model is that we have labeled and internationally ac-

knowledged topics, which are easier to interpret than automatically generated topics. The pre-processing of the documents with TextImager has improved the classification quality, as we have been able to use additional morphological information of the words. The best results were achieved by the additional application of fastSense and using word embeddings based on the disambiguated words.

text2ddc was trained in more than 40 languages supported by Wikipedia and achieves results of over 88% F-score for the second level DDC in German. We also trained text2ddc on the third level of the DDC and here again we achieve great results of about 81% in the example of German. The resulting topic distribution derived from text2ddc can then be used to improve other NLP methods, such as classification or clustering, by better defining the semantic context.

However, there is the disadvantage of CTMs within the DDC that the thematic universe is limited and not constantly evolving. This has a great advantage in terms of comparability, but we also wanted to develop a topic model whose thematic universe will keep up with the times. Therefore we worked on the development of an OTM and examined the Wikipedia category system. Since the Wikipedia category system is created through social tagging and is adapted and expanded daily, it has the perfect prerequisites for an OTM (Mehler and Waltinger 2009; Mehler 2011).

Several studies have already dealt with the Wikipedia category system. Voss (2006) explores this system and found out, that the analysis of structural and statistical properties shows that the Wikipedia category system is a thesaurus that combines collaborative tagging and hierarchical classification in a special way. There are also studies that have dealt with the classification of texts using the Wikipedia category system (J. Raiman and O. Raiman 2018; Medeiros et al. 2018). However, in these cases the category system is reduced to only a few categories. In addition, by specifying these, the criteria of an OTM are not fulfilled. Schönhofen (2009) also used Wikipedia's category system to improve further classifications. For this he searched for Wikipedia titles in a text, then associated them with the Wikipedia articles and finally weighted the associated categories. Türker et al. (2019) calculates word embeddings by additionally using the categories of Wikipedia articles to get more information

and shows that he achieves better results using them. These examples show how useful Wikipedia and its category system can be in the field of natural language processing.

However many advantages the Wikipedia category system has, it also has its challenges. One of the biggest challenges in dealing with the Wikipedia category system is its size and lack of clarity. The advantage that comes with social tagging also brings disadvantages, such as the impurity of the category system. Thus, it is common to find categories that contain only a few or a single article and are so specific that many would not perceive them as a category. Therefore, we first had to define various criteria in order to be able to approach the category system. For example, we have filtered categories that do not contain enough articles or are too low within the category tree. Thus, using this criteria, we have created an initial topic model, with about 3,806 categories. Furthermore, we divided the category system of Wikipedia into 3 parts:

- Space (Universe, World, Europe, etc.)
- Time (Millennium, century, etc.)
- Theme (Mathematics, sports, politics, etc.)

This gives us 3 different models that analyze the text in terms of *"what location is it about?"*, *"what time period is it about?"* and *"what topic is it about?"*. This also has the advantage of combining these models. For example, if we have recognized *Asia* as a location, the *17th century* as a period of time, and *history* as a theme, the text probably concerns Asian history in the context of the 17th century. As a result, our open topic model called `text2wiki` becomes multimodal and thus even more powerful, as it provides semantic analysis beyond topic modeling.

Despite having so many categories, we achieve a classification quality of over 80%. Our previously developed CTM text2ddc has made a significant contribution to this. By using our *small* topic model with only about 100 DDC classes, we were able to help text2wiki to better understand the context of the documents and thus improve the classification of the thousands of Wikipedia categories. In addition, the pre-processing using TextImager and the disambiguation by fastSense has improved the classification.

In order not to be limited to the 3,806 categories, we have trained further models, including much more categories. This gives us an even more specific thematic model as we move even deeper along the category tree, resulting in models containing 7,500, 10,000, 15,000 and 20,000 categories. Despite the growing number of categories, the scores remain consistently high. In this way, the user can decide how detailed the topic universe should be. The more detailed, the larger the topic vector and the more challenging the computational challenge.

To make the resulting categories more legible, we converted them back to the tree format of Wikipedia. This category tree contains only the classified topics and moves up along the usual Wikipedia category tree. For this we have used a tree visualization technique that displays this structure in a clear manner. In Figure 1.1 we see the category tree of text2wiki on a text about the Pythagorean theorem.

In order to evaluate the use of text2wiki, we applied it to other classification tasks including text2ddc. We have incorporated the identified topics from text2wiki as additional input features for the text2ddc classifier. The classification quality of text2ddc has been improved by up to 3% using this additional information. Consequently, many of the related studies that are based on the categories of Wikipedia articles may also benefit from the use of text2wiki. An additional benefit is the categorization of out-domain documents according to the Wikipedia scheme, allowing these related studies to be applied on non-Wikipedia corpora.

We have also measured the change in Wikipedia categories over time. Thus we were able to confirm that the Wikipedia category system is constantly evolving and that the topic universe changes over time, which is an important criterion for OTMs.

In this dissertation we have dealt with various aspects of the analysis of multiple documents. This includes pre-processing, data visualization, text classification and semantic analysis at word and document level. We are able to convert unstructured text into a machine-readable form, visualize and analyze the data and retrieve semantic information. An important factor for the success of this dissertation was the ability of the different models to interact with each other. The developed methods were used or incorporated for the devel-

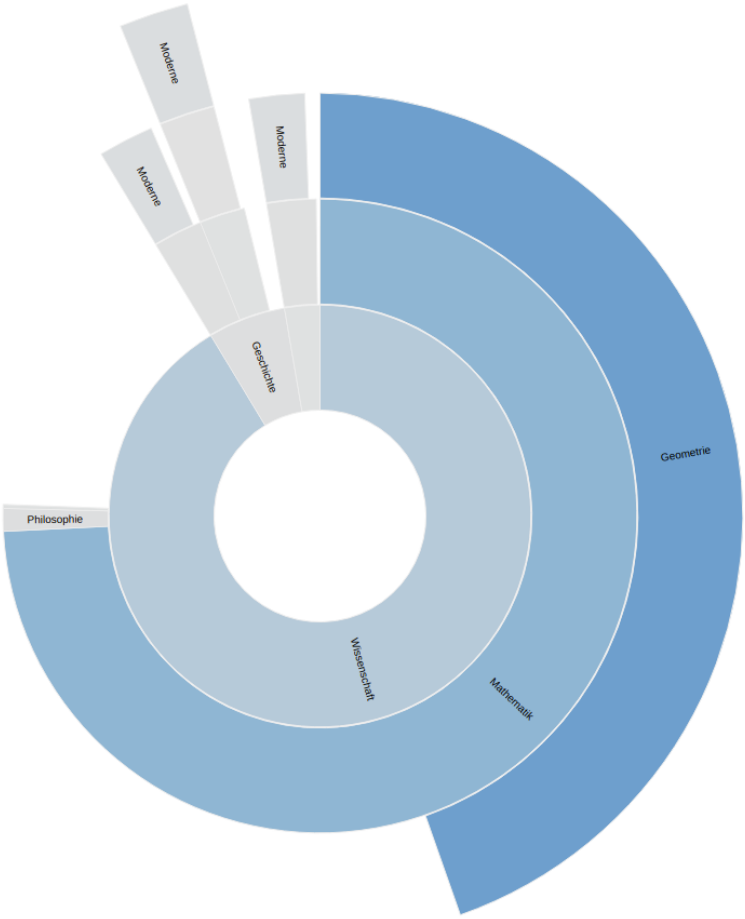


Figure 1.1: Output of the text2wiki visualization for an article about the Pythagorean theorem.

opment of further methods. Thus, we were able to achieve improvements in performance by utilizing extracted information provided by previously created tools and methods of this dissertation. We will explain the interaction model of this dissertation in more detail in the next chapter.

1.3 Interaction model

This dissertation presents many tools and systems developed to solve different aspects in the field of NLP. A big advantage and also a reason why these tools

became so efficient is their dependency on each other. Each tool offers solutions for specific tasks, whereby these solutions can often be used in other areas as well. Figure 1.2 describes the interaction model of the tools and systems presented in this dissertation. The various components are connected to each other in a variety of ways. In this case, a directed edge indicates the usage of the source node in the target node. For example, by using TextImager we were able to successfully develop fastSense and thus efficiently disambiguate words. The disambiguation information by fastSense helped us to improve our classifier-induced topic model – text2ddc. The combination of fastSense and text2ddc has contributed to the optimization of our open topic model – text2wiki. All these methods and models are finally fed back into TextImager to be used in its multi-server architecture. The interaction and the performance improvement through the use of our systems was an important indication that the developed systems are fully interoperable and can contribute to further improvement.

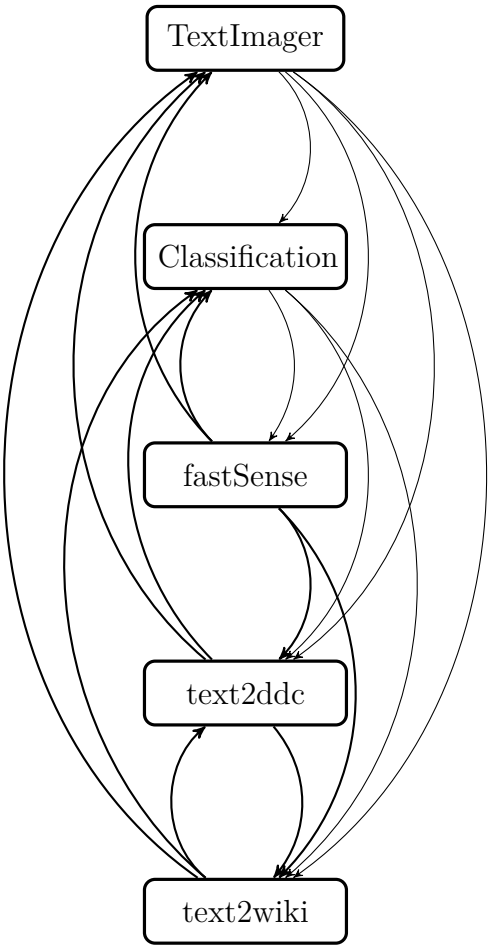


Figure 1.2: The interaction model of the tools and systems presented in this dissertation.

1.4 Dissertation overview

The dissertation is structured as follows:

Chapters 2 to 12 show the articles that have already been published in peer-reviewed conferences. Chapters 2 to 5 include articles dealing with the preprocessing of texts (TextImager) and the subsequent analysis using interactive visualizations. In Chapters 6 to 8, the acquired information and visualizations are used to perform classification on documents. Chapters 9 to 12 include articles about the semantic analysis of documents, whereas Chapter 9 deals with the semantic analysis at word level (WSD), while Chapters 10 and 11 present a closed topic model based on DDC (text2ddc) and Chapter 12 introduces an open topic model by exploring the Wikipedia category system (text2wiki).

Chapter 13 summarizes all these articles, resumes all the results and gives an outlook on future work.

2 TextImager: a Distributed UIMA-based System for NLP

The content of this chapter (Hemati, Uslu, and Mehler 2016) was published in Proceedings of the COLING 2016 System Demonstrations.

2.1 Abstract

More and more disciplines require NLP tools for performing automatic text analyses on various levels of linguistic resolution. However, the usage of established NLP frameworks is often hampered for several reasons: in most cases, they require basic to sophisticated programming skills, interfere with interoperability due to using non-standard I/O-formats and often lack tools for visualizing computational results. This makes it difficult especially for humanities scholars to use such frameworks. In order to cope with these challenges, we present TextImager, a UIMA-based framework that offers a range of NLP and visualization tools by means of a user-friendly GUI. Using TextImager requires no programming skills.

2.2 Introduction

Computational humanities and related disciplines require a wide range of NLP tools to perform automatic text analyses on various levels of textual resolution. This includes, for example, humanities scholars dealing with repositories of historical documents, forensic linguists analyzing unstructured texts of online social media to create digital fingerprints of suspects or even doctors using

clinical NLP to support differential diagnosis based on physician-patient talks. However, established NLP frameworks still require basic to sophisticated programming skills for performing such analyses. This hampers their usage for users who are not sufficiently trained neither in computational linguistics nor in computer science. Further, these frameworks often lack interoperability due to using non-standard I/O-formats. We present TextImager to cope with these challenges. The longer-term goal of TextImager is to provide a platform into which any open source/access NLP tool can be integrated. To this end, TextImager provides a web-based GUI whose usage does not require any programming skills while making accessible a range of tools for visualizing results of text analyses. In order to ensure standardization and interoperability, TextImager is based on the *Unstructured Information Management Applications* (UIMA) framework. Currently, the scope of TextImager ranges from tokenizing, lemmatizing, POS-tagging, text similarity measurements to sentiment analysis, text classification, topic modeling and many more.

2.3 Related Work

Frameworks of computational texts analysis have already been introduced and are now common in industrial use. This includes, for example, UIMA (Ferrucci and Lally 2004), DKPro (Castilho and Gurevych 2014), OpenNLP[1] and Gate (Cunningham et al. 2011). Note that these frameworks do not provide visualization interfaces and require versatile programming skills for set up. Thus, they cannot be recommended for being used by computationally less trained users. We provide the TextImager to cope with this problem while integrating most of the components of these frameworks. On the other hand, Voyant Tools (Bird, Edward, and Klein 2009; Ruecker, Radzikowska, and Sinclair 2011), WebNLP (Burghardt et al. 2014) and conTEXT (Khalili, Auer, and Ngomo 2014) are web-based NLP tools including visualization components. In order to combine the best of both worlds, TextImager additionally subsumes the functionalities of these tools. It also shares functionalities with WebLicht (E. W. Hinrichs, M. Hinrichs, and Zastrow 2010). However, unlike WebLicht, TextImager is based on UIMA and, thus, complies to an industrial

[1] https://opennlp.apache.org/

standard of modeling text processing chains.

2.4 System Architecture of TextImager

TextImager consists of two parts, front end and back end. The front end is a web application that makes all functionalities and NLP processes available in a user-friendly way. It allows users for analyzing and visualizing unstructured text and corpora. The back end is a highly modular, expandable, scalable and flexible architecture with parallel processing capabilities.

2.4.1 Back end

Figure 2.1 shows the architecture of TextImager. Every NLP component of TextImager implements a UIMA interface. Every UIMA compatible NLP-component can easily be integrated into TextImager. Even modules not compatible with UIMA can be integrated with just a slight effort. Amongst others, we have integrated DKPro (see Section 2.3), which offers a variety of UIMA-components. We also integrated the *UIMA Asynchronous Scaleout* (UIMA-AS)[2] add-on.

TextImager allows users for dynamically choosing NLP components in a pipeline. To this end, we extended UIMA-AS by initiating components without XML descriptors by means of uimaFIT[3]. We extended this framework by allowing for dynamic instantiations of pipelines. These extensions make our framework highly flexible, adaptive and extensible during runtime.

All TextImager components are configured as UIMA-AS services, which may run standalone or in a pipeline. All services are located on servers to allow for communication among them. Note that we are not limited to run these components on a single server; rather, they can be distributed among different servers (see Figure 2.1). We developed a mechanism that automatically selects and acquires components and their resources: it arranges components into pipelines and grants the ability to parallelize them. Thus, components

[2] https://uima.apache.org/doc-uimaas-what.html
[3] https://uima.apache.org/uimafit.html

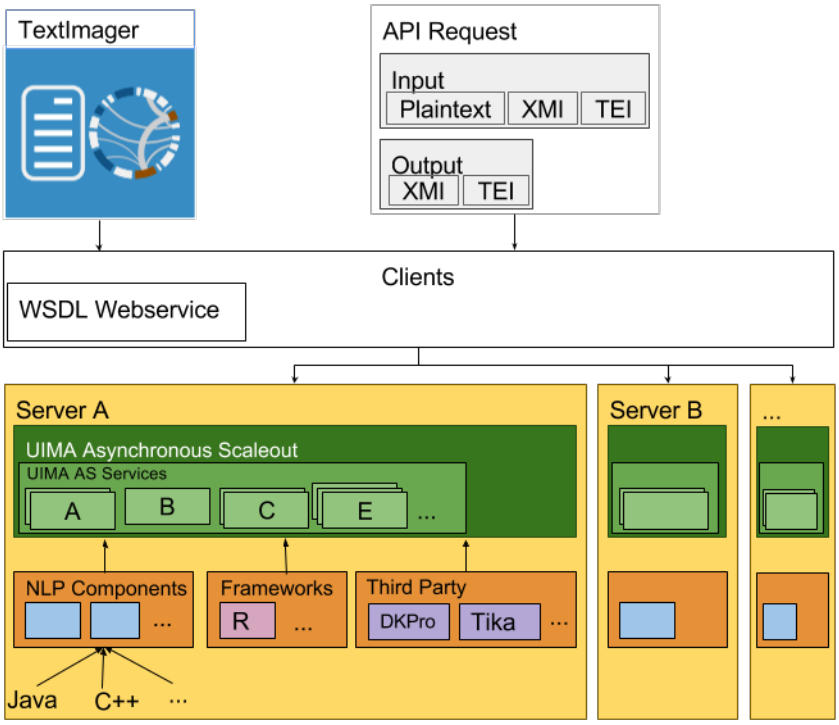
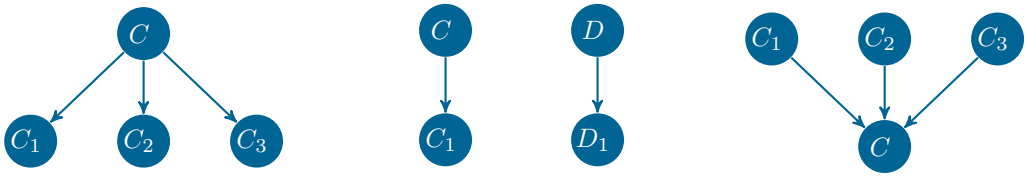


Figure 2.1: TextImager's back end.

that do not depend on each other can run in parallel. For this we developed an advanced UIMA flow controller. Take the examples displayed in Fig.2.2: suppose that vertices in these examples denote NLP components; suppose further that the corresponding arcs denote interdependencies between these components. In Fig. 2.2a, the components C_1, C_2 and C_3 do not depend on each other. Thus, they can run in parallel. In Fig. 2.2b, components C_1 and D_1 do not depend on each other, but on C and D, respectively. Thus, C and D can run in parallel as can do the components C_1 and D_1. In Fig. 2.2c, C depends on C_1, C_2 and C_3. Thus, running C has to wait on the termination of C_1, C_2 and C_3. Within TextImager, dependency hierarchies of components as exemplified by these three examples are generated from information provided by each of the components supposed that their input and output types have been defined appropriately (cf. the class specifications of type `org.apache.uima.fit.descriptor.TypeCapability`). In this way, TextImager allows for realizing a wide range of processing chains.

One advantage of our framework is that it does not rely on a central repository. Rather, TextImager can be distributed across multiple servers. This allows developers for setting up their own TextImager server and to distribute their



(a) (b) (c)

Figure 2.2: Component dependency types.

own NLP tools within the TextImager ecosystem.

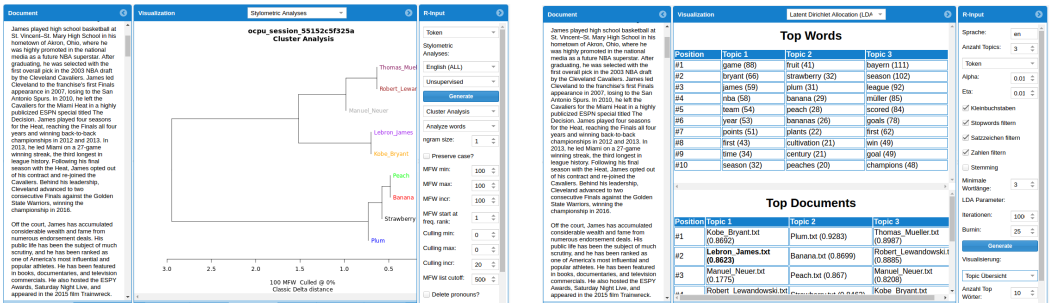
TextImager can be used within a web application that offers a graphical user interface. Alternatively, TextImager can be used via a WSDL webservice client.

2.4.2 Front end

The front end gives access to all NLP tools integrated into TextImager without requiring any programming skills. This is done by means of a GUI that even provides three-dimensional text visualizations (see Figure 2.4b). All visualizations are interactive in the sense of allowing for focusing and contextualizing results of text analysis (e.g., the macro *reference distribution of sentence similarity across multiple documents* exemplified in Figure 2.4d). The GUI contains a text and a visualization panel. One of TextImager's guiding principles is to enable *bidirectional interactivity*. That is, any interaction with the visualization panel is synchronized by automatically adjusting the content of the text panel and vice versa. The front end is based on *Ext JS*, a JavaScript framework for building interactive cross platform web applications. The visualizations are done by means of *D3.js*[4] and *vis.js*[5] to enable browser-based visualizations while handling large amounts of data.

Figure 2.4 exemplifies TextImager's GUI. With a focus on close reading, TextImager supports the interpretation of single texts by determining, for example, their central topics or by depicting their unfolding from constituent to constituent (see Figure 2.4g, 2.4a, 2.4h). Regarding distant reading (Jänicke et al. 2015), TextImager provides more abstract overviews of the content of

[4] `https://www.d3js.org`
[5] `http://visjs.org`



(a) Stylometric (b) LDA

Figure 2.3: R packages in TextImager

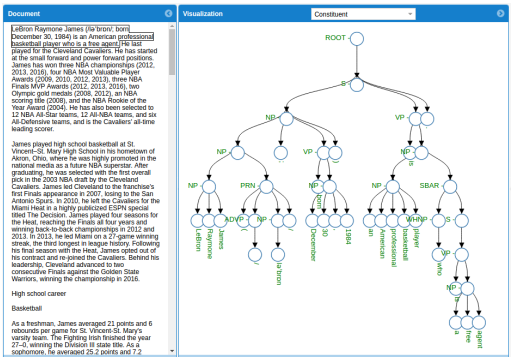
text corpora. Here, visualizations provide summary information as exemplified in Figure 2.4b, 2.4c, 2.4d, 2.4f.

Last but not least, TextImager provides a generic interface to R[6]. The aim is to give access to any NLP-related package in R *once more without requiring programming skills*. This is especially needed for scholars in digital humanities who are not trained in using script languages for modeling statistical procedures, but expect a versatile tool encapsulating this computational complexity. Thus, TextImager users can process input texts using R packages like LDA (see Figure 2.3b), network analysis or stylometrics (see Figure 2.3a) without the need to manipulate or to invoke any R script directly. All these R packages are given a single entrance point in the form of TextImager. See Mehler, Uslu, and Hemati (2016) for a recent research study based on TextImager.

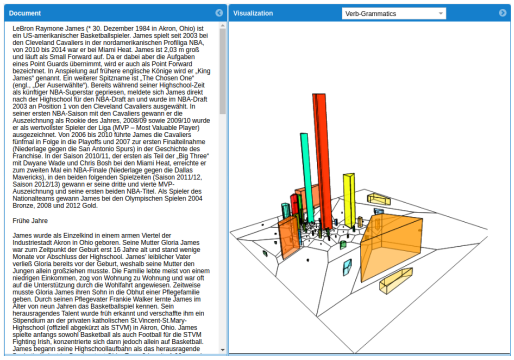
2.5 Future Work

In already ongoing work, we extend the functionality of TextImager. This includes covering all features of tools like conTEXT. In contrast to many current frameworks, we will make TextImager's source code open-source as soon as the framework reaches a stable and documented version. We are going to specify a comprehensive model for component specification. The model will contain specifications of general components and their dependency hierarchy. This model will help defining where new NLP components are settled within the NLP landscape.

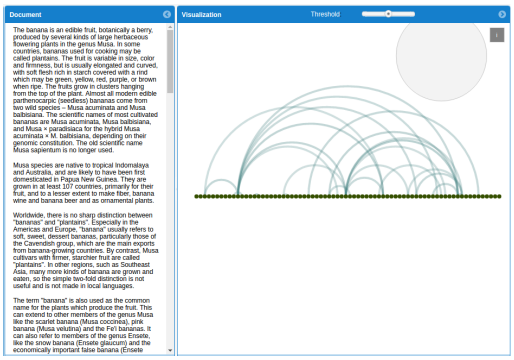
[6] `https://www.r-project.org`



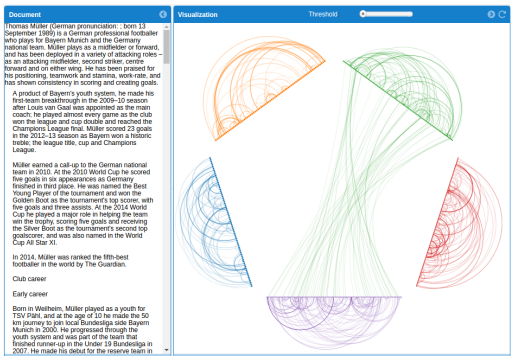
(a) Constituent parse tree



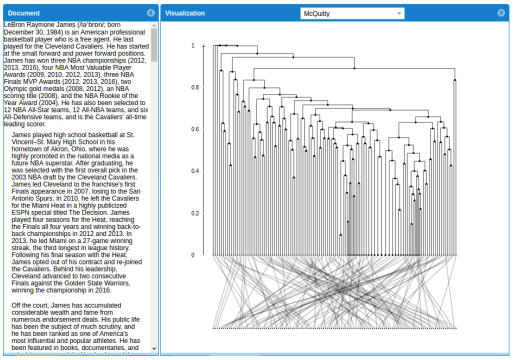
(b) Text2Voronoi



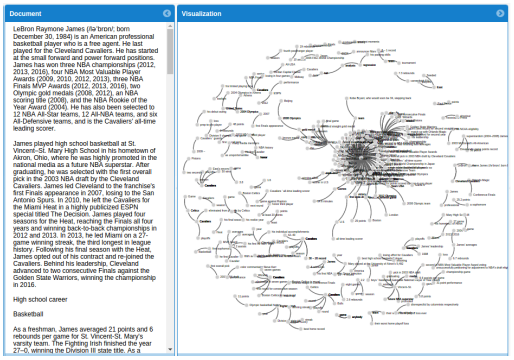
(c) Innertextual similarity



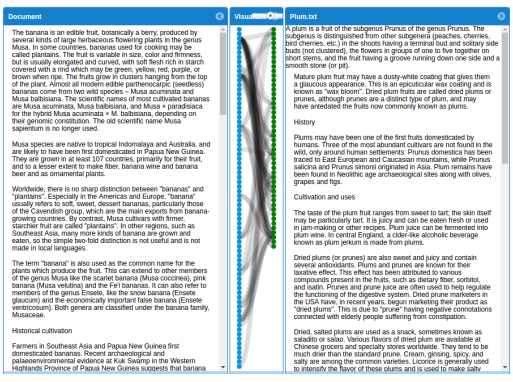
(d) Intertextual similarity



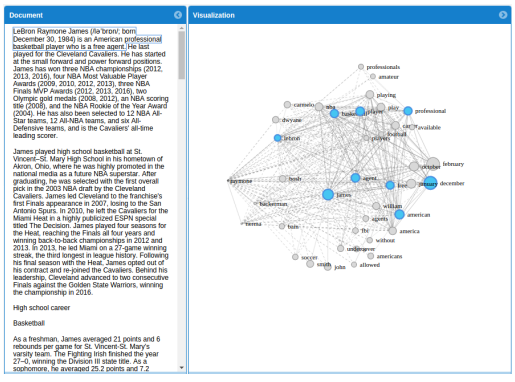
(e) Dendrogramcluster similarity



(f) Relation graph



(g) Bipartite similarity



(h) Semantic relation graph

Figure 2.4: Visualization Examples

3 TextImager as a Generic Interface to R

The content of this chapter (Uslu, Hemati, et al. 2017) was published in Software Demonstrations of the 15th Conference of the European Chapter of the Association for Computational Linguistics (EACL 2017).

3.1 Abstract

R is a very powerful framework for statistical modeling. Thus, it is of high importance to integrate R with state-of-the-art tools in NLP. In this paper, we present the functionality and architecture of such an integration by means of TextImager. We use the *OpenCPU API* to integrate R based on our own *R* server. This allows for communicating with *R*-packages and combining them with TextImager's NLP-components.

3.2 Introduction

Hemati, Uslu, and Mehler (2016) introduce TextImager, focusing on its architecture and the functions of its backend. In this paper, we present the functionality and architecture of *R* interfaced by means of TextImager. For this purpose, we created a separate panel in TextImager for *R*-applications. In this panel, we combine state-of-the-art NLP tools embedded into TextImager with the powerful statistics of *R* (R Development Core Team 2008). We use the *OpenCPU API* (Ooms 2014) to integrate *R* into TextImager by means of our own *R*-server. This allows for easily communicating with the built-in *R*-packages and combining the advantages of both worlds. In the case of

topic detection, for example, the complete text is used as an input string to *R*. Thanks to *TextImager*'s pre-processor, more information is provided about syntactic words, parts of speech, lemmas, grammatical categories etc., which can improve topic detection. Further, the output of R methods is displayed by means of TextImager's visualizations, all of which are linked to the corresponding input text(s). This allows for unprecedented *interaction* between text and the statistical results computed by *R*. For this paper we sampled several Wikipedia articles to present all features of *R* integrated into TextImager. This includes articles about four politicians (*Angela Merkel*, *Barack Obama*, *Recep Tayyip Erdoğan*, *Donald Trump*) and five sportsman, that is, three basketball players (*Michael Jordan*, *Kobe Bryant* and *Lebron James*) and two soccer players (*Thomas Müller* and *Bastian Schweinsteiger*).

3.3 Related Work

R is used and developed by a large community covering a wide range of statistical packages. The *CRAN*[1] package repository is the main repository of the *R* project. It currently consists of about 10,000 packages including packages for NLP.[2] The *R* framework requires basic to versatile skills in programming and scripting. It provides limited visualization and interaction functionalities. Attempts have been undertaken to provide web interfaces for *R* as, for example, *Shiny*[3] and *rApache*[4]. Though they provide a variety of functions and visualizations[5], these tools are not optimized for statistical NLP: their NLP-related functionalities are rather limited. In order to fill this gap, we introduce TextImager's *R* package, that is, a web based tool for NLP utilizing *R*.

3.4 Architecture

TextImager is a *UIMA*-based (Ferrucci and Lally 2004) framework that offers a wide range of NLP and visualization tools by means of a user-friendly *GUI*

[1] https://cran.r-project.org/
[2] https://cran.r-project.org/web/views/NaturalLanguageProcessing.html
[3] http://shiny.rstudio.com/
[4] http://rapache.net/
[5] http://shiny.rstudio.com/gallery/

without requiring programming skills. It consists of two parts: front end and back end. The back end is a modular, expandable, scalable and flexible architecture with parallel and distributed processing capabilities (Hemati, Uslu, and Mehler 2016). The front end is a web application that makes NLP processes available in a user-friendly way with responsive and interactive visualizations (Hemati, Uslu, and Mehler 2016). TextImager already integrated many third party tools. One of them is *R*. This section describes the technical integration and utilization of *R* into TextImager.

3.4.1 R / OpenCPU

R is a software environment for statistical computing. It compiles and runs on a variety of UNIX and Windows platforms. One of our goals is to provide an easy to use interface for R with a focus on NLP. To this end, we use *OpenCPU* to integrate *R* into TextImager. *OpenCPU* provides an *HTTP API*, which allocates the functionalities of R packages (Ooms 2014). The *OpenCPU* software can be used directly in *R*; alternatively, it can be installed on a server. We decided for the latter variant. In addition, we used the JavaScript library *opencpu.js* which simplifies the API usage in JavaScript and allows for calling *R*-functions directly from TextImager. To minimize the communication effort between client and server and to encapsulate *R* scripts from TextImager's functionality, we created a so called TextImager-*R*-package that takes TextImager data, performs all *R*-based calculations and returns the results. This package serves for converting any TextImager data to meet the input requirements of any *R*-package. In this way, we can easily add new packages to TextImager without changing the HTTP request code. Because some data and models have a long build time we used *OpenCPU*'s session feature to keep this data on the server and access it in future sessions so that we do not have to recreate it. This allows the user for quickly executing several methods even in parallel without recalculating them each time.

3.4.2 Data Structure

The data structure of TextImager differs from *R*'s data structure. Therefore, we developed a generic mapping interface that translates the data structure

from TextImager to an *R*-readable format. Depending on the *R*-package, we send the required data via *OpenCPU*. This allows for combining each NLP tool with any *R*-package.

3.4.3 OpenCPU Output Integration

Visualizing the results of a calculation or sensitivity analysis is an important task. That is why we provide interactive visualizations to make the information available to the user more comprehensible. This allows the user to interact with the text and the output of R, for example, by highlighting any focal sentence in a document by interacting with a graph generated by means of R.

3.4.4 R packages

This section gives an overview of *R*-packages embedded into the pipeline of TextImager.

tm The *tm*-package (Feinerer and Hornik 2015) is a text mining R-package containing pre-processing methods for data importing, corpus handling, stopword filtering and more.

lda The *lda*-package (J. Chang 2015) provides implementations of *Latent Dirichlet Allocation* algorithms. It is used to automatically identify topics in documents, to get top documents for these topics and to predict document labels. In addition to the tabular output we developed an interactive visualization, which assigns the decisive words to the appropriate topic (see Figure 3.1). This visualization makes it easy to differentiate between the two topics and see which words classify these topics. In our example, we have recognized words such as *players*, *season* and *game* as one topic (sportsman) and *party*, *state* and *politically* as a different topic (politics). The parameters of every package can be set on runtime. The utilization and combination of TextImager and *R* makes it possible, to calculate topics not only based on wordforms, but also takes other features into account like lemma, pos-tags, morphological features and more.

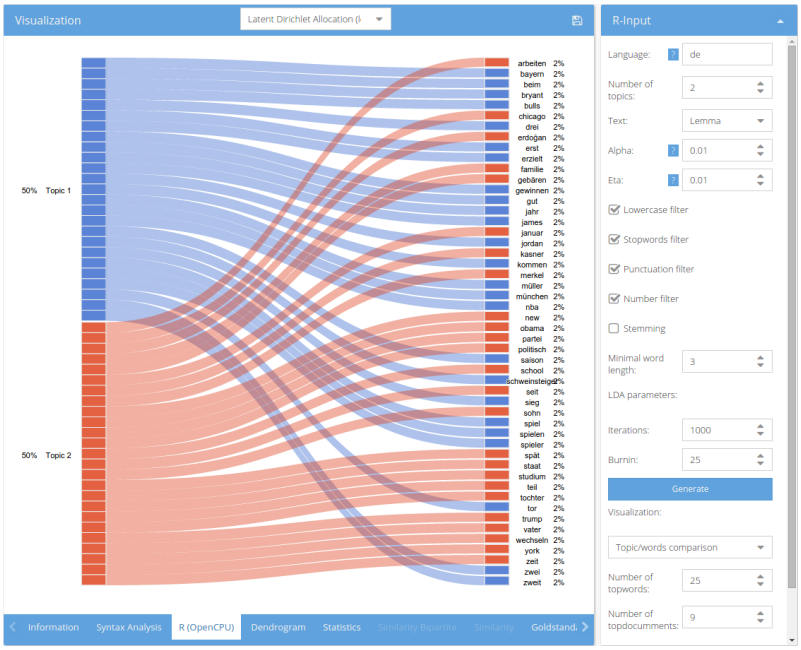


Figure 3.1: Words assigned to their detected topics

stylo The *stylo*-package (Eder, Rybicki, and Kestemont 2016) provides functionality for stylometric analyses. All parameters of the package can be set through the graphical user interface of TextImager. The package provides multiple unsupervised analyses, mostly based on a most-frequent-word list and contrastive text analysis. In Figure 3.2 we have calculated a cluster analysis based on our example corpus. We can see that the politicians, basketball players and soccer players have been clustered into their own groups.

LSAfun The *LSAfun* (Günther, Dudschig, and Kaup 2015) and *lsa* (Wild 2015) packages provide functionality for *Latent Semantic Analysis*. In TextImager it is used to generate summaries of documents and similarities of sentences.

igraph The *igraph*-package (Csardi and Nepusz 2006) provides multiple network analysis tools. We created a document network by linking each word with its five most similar words based on word embeddings. The *igraph*-package also allows to layout the graph and calculate different centrality measures. In the end, we can export the network in many formats (*GraphML*, *GEXF*, *JSON*, etc.) and edit it with graph editors. We also build an interactive network visualization (see figure 3.3) using

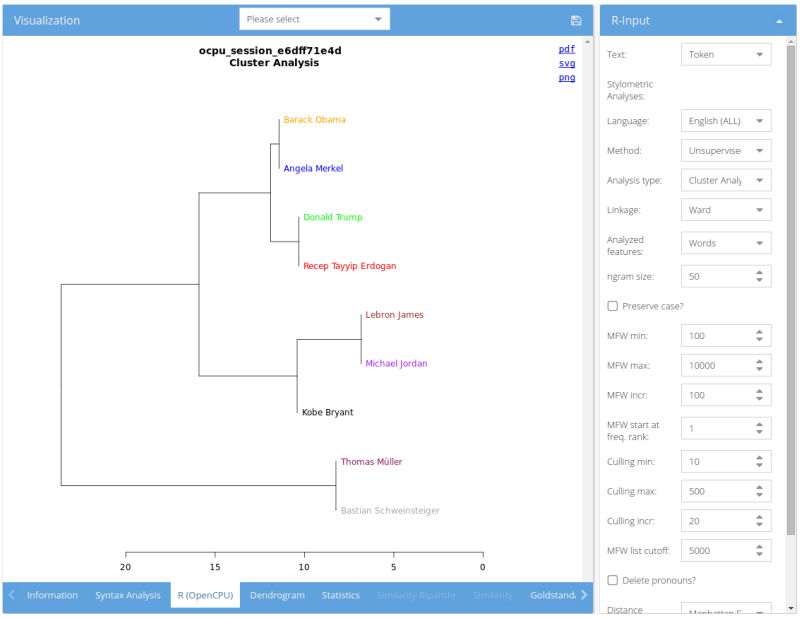


Figure 3.2: Cluster analysis of the documents

sigma.js[6] to make it interoperable with TextImager.

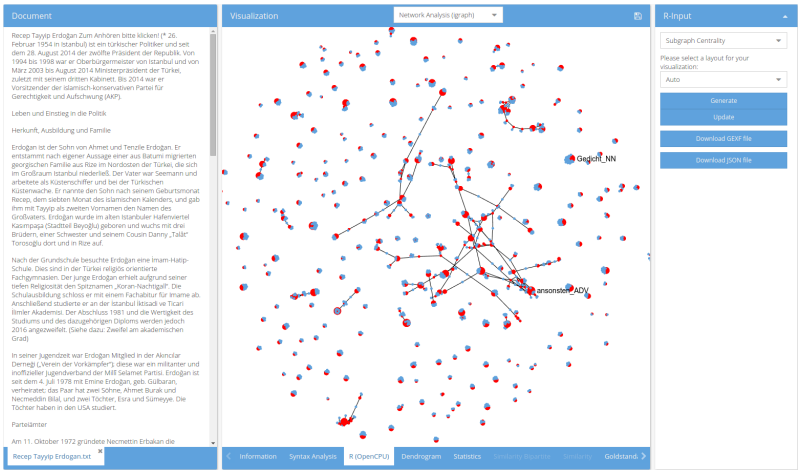
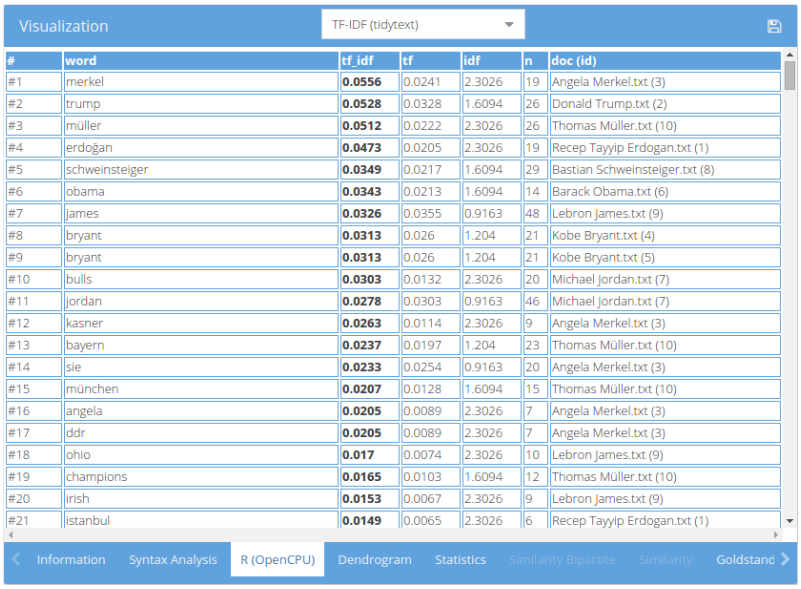


Figure 3.3: Network graph based on word embeddings

tidytext The *tidytext*-package (Silge and Robinson 2016) provides functionality to create datasets following the tidy data principles (Wickham 2014). We used it with our *tm* based corpus to calculate *TF-IDF* information of documents. In Figure 3.4 we see the output tabular with informations like *tf*, *idf* and *tf-idf*

stringdist The *stringdist*-package (van der Loo 2014) implements distance cal-

[6] sigmajs.org



#	word	tf_idf	tf	idf	n	doc (id)
#1	merkel	**0.0556**	0.0241	2.3026	19	Angela Merkel.txt (3)
#2	trump	**0.0528**	0.0328	1.6094	26	Donald Trump.txt (2)
#3	müller	**0.0512**	0.0222	2.3026	26	Thomas Müller.txt (10)
#4	erdoğan	**0.0473**	0.0205	2.3026	19	Recep Tayyip Erdogan.txt (1)
#5	schweinsteiger	**0.0349**	0.0217	1.6094	29	Bastian Schweinsteiger.txt (8)
#6	obama	**0.0343**	0.0213	1.6094	14	Barack Obama.txt (6)
#7	james	**0.0326**	0.0355	0.9163	48	Lebron James.txt (9)
#8	bryant	**0.0313**	0.026	1.204	21	Kobe Bryant.txt (4)
#9	bryant	**0.0313**	0.026	1.204	21	Kobe Bryant.txt (5)
#10	bulls	**0.0303**	0.0132	2.3026	20	Michael Jordan.txt (7)
#11	jordan	**0.0278**	0.0303	0.9163	46	Michael Jordan.txt (7)
#12	kasner	**0.0263**	0.0114	2.3026	9	Angela Merkel.txt (3)
#13	bayern	**0.0237**	0.0197	1.204	23	Thomas Müller.txt (10)
#14	sie	**0.0233**	0.0254	0.9163	20	Angela Merkel.txt (3)
#15	münchen	**0.0207**	0.0128	1.6094	15	Thomas Müller.txt (10)
#16	angela	**0.0205**	0.0089	2.3026	7	Angela Merkel.txt (3)
#17	ddr	**0.0205**	0.0089	2.3026	7	Angela Merkel.txt (3)
#18	ohio	**0.017**	0.0074	2.3026	10	Lebron James.txt (9)
#19	champions	**0.0165**	0.0103	1.6094	12	Thomas Müller.txt (10)
#20	irish	**0.0153**	0.0067	2.3026	9	Lebron James.txt (9)
#21	istanbul	**0.0149**	0.0065	2.3026	6	Recep Tayyip Erdogan.txt (1)

Figure 3.4: Statistical information of the documents

culation methods, like `Cosine`, `Jaccard`, `OSA` and other. We implemented a function for calculating sentence similarities and provide an interactive visual representation. Each node represents an sentence of the selected document and the links between them represent the similarity of those sentences. The thicker the links, the more similar they are. By interacting with the visualization, corresponding sections of the document panel are getting highlighted, to see the similar sentences (see Figure 3.5). The bidirectional interaction functionality enables easy comparability.

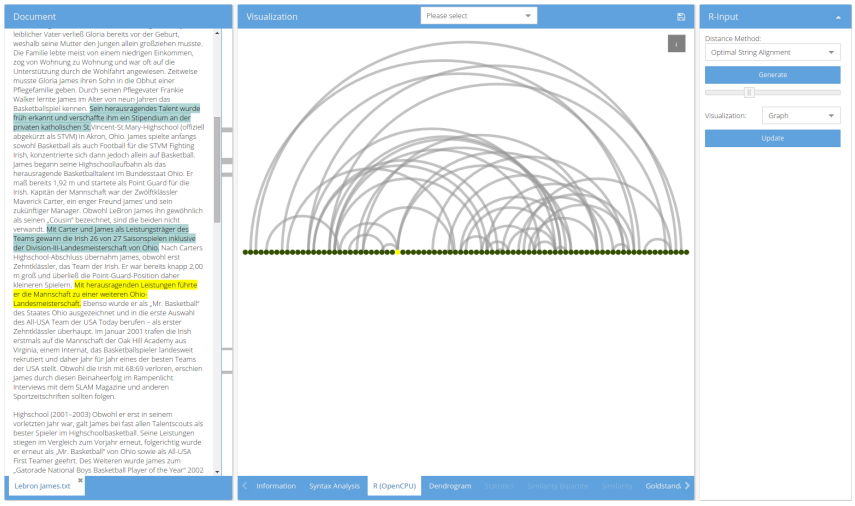


Figure 3.5: Depiction of sentence similarities.

stats We used functions from the *R*-package *stats* (R Development Core Team 2008) to calculate a hierarchical cluster analysis based on the sentence similarities. This allows us to cluster similar sentences and visualize them with an interactive dendrogram. In Figure 3.6 we selected one of these clusters and the document panel immediately adapts and highlights all the sentences in this cluster.

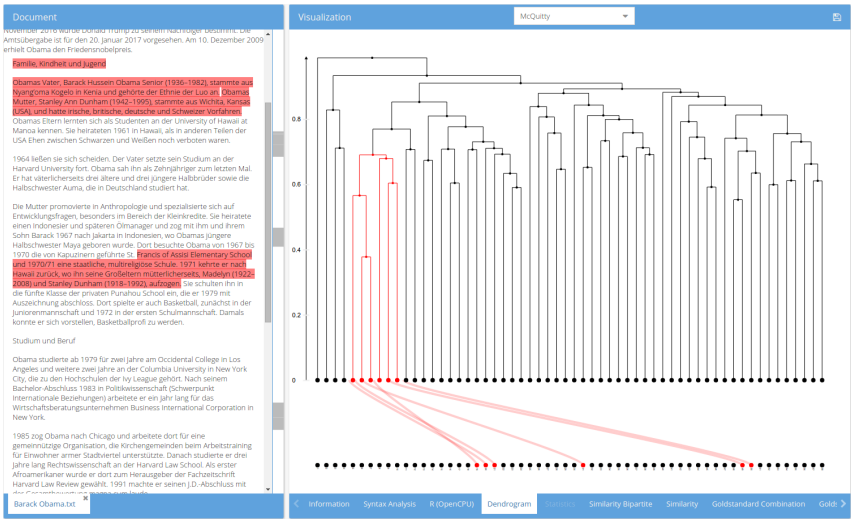


Figure 3.6: Similarity-clustered sentences.

An interesting side effect of integrating these tools into TextImager's pipeline is that their output can be concerted in a way to arrive at higher-level text annotations and analyses. In this way, we provide to an integration of two heavily expanding areas, that is, NLP and statistical modeling.

3.5 Future work

In already ongoing work, we focus on big data as exemplified by the Wikipedia. We also extend the number of built-in *R*-packages in TextImager.

4 PolyViz: a Visualization System for a Special Kind of Multipartite Graphs

The content of this chapter (Uslu and Mehler 2018) was published in Proceedings of the IEEE VIS 2018.

4.1 Abstract

In this paper we present PolyViz, a new visualization system that can efficiently display a special kind of k-partite graphs with the benefit that the k groups themselves can also have links. PolyViz not only allows for the generation of the visualization, but also for the adaptation and the analysis of the underlying data. This was achieved by providing various interaction possibilities. We illustrate the visualization in the context of two conducted experiments. One of these experiments includes the analysis of the topic distribution of the German Wikipedia and the linkage of these topics. The other experiment is about the visual representation of sentence similarities including their analysis. PolyViz is not limited to these applications but can be used for visualizing any multipartite data.

4.2 Introduction

Many visualizations serve to refine papers and to display the calculated data in a readable way. However, it often takes a lot of time and adaptation to

create these visualizations. In this paper, we present PolyViz, a visualization tool that can display a special kind of multipartite graphs and adapt them according to various requirements. To this end, one has to transfer the data into the requested JSON input structure and can create, adapt, analyze and download the resulting visualization. In a standard multipartite graph, connections between nodes of the same group are not allowed. In our approach, we allow for such links in order to increase the number of use cases. Links in the same group are displayed by spanning an arc diagram for each group. The paper demonstrates two use cases in which PolyViz has already been used.

In Section 4.3 we will describe the visualization and how it was implemented. In Section 4.4 we describe the system, the visualization can be created, adapted and analyzed with. In Section 4.5 the example applications of the visualization are described and in Section 4.6 we give a short summary and an insight to future work.

4.3 Visualization

In this chapter we will discuss the implementation of the visualization. It is based on *D3* (Bostock, Ogievetsky, and Heer 2011), a Javascript library for creating dynamic and interactive visualizations in the web browser. In our case we wanted to visualize a special case of a k-partite graph. A k-partite graph is partitioned into k groups each of a certain number of nodes. A standard k-partite graph does not allow for edges linking nodes of the same group. However, in order to increase the number of use cases addressed by our visualization method, we allow for such an extension.

First of all, each group is uniquely mapped onto a path. This is achieved by drawing a k-corner in a circle and using the intersections of the corner with the circle to determine the beginning and end of the corresponding path. Figure 4.1 illustrates the construction of the paths for different number of groups. In the next step, the target nodes are mapped onto these paths by taking into account an even distribution of the distances among nodes of the same path. The maximum size of a node equals the smallest distance between two nodes. All other nodes are scaled accordingly. The same procedure applies to the edges. This has the advantage that the size of the nodes and edges does

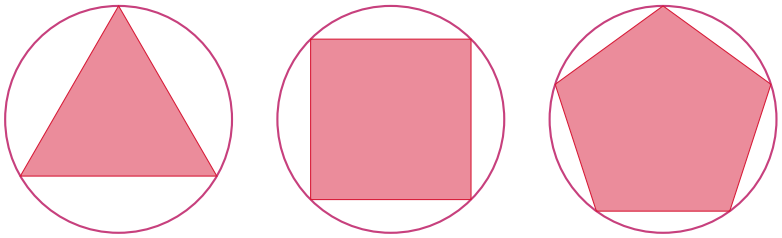


Figure 4.1: Generation of the paths for k = 3, 4 and 5.

not become too large to avoid overlaps. For edges between nodes of different groups, Bézier curves are drawn using the center of the graph as the focus point. For edges between nodes of the same group, a semi-circle is drawn outwards, resulting in an arc diagram for each group – see Figure 4.3 for a good example.

For determining the colors, we utilize the color palette of D3. Each group

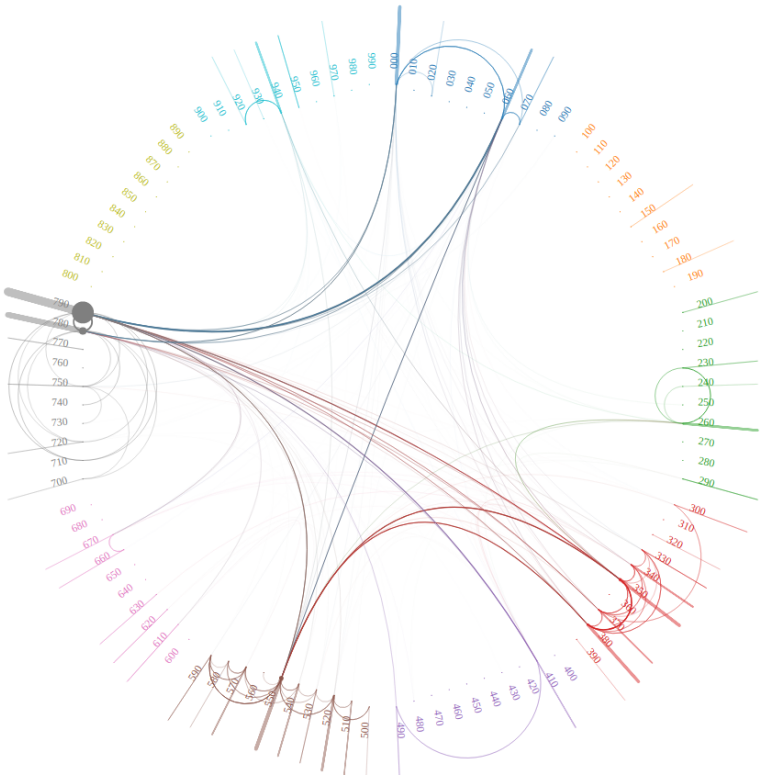


Figure 4.2: Example of a 10-partite graph visualizing the DDC-related topic distribution of the German Wikipedia: nodes denote classes of the second level Dewey Decimal Classification (DDC). Edges denote links between articles subsumed under the corresponding topic nodes.

has its own color and all nodes are colored according to the color of the corresponding group. Group-internal edges are assigned the color of the group; edges between different groups are represented by a color gradient between the colors of the groups involved. This makes it easier to find out where outgoing edges end and where incoming edges come from.

4.4 The System

Our system consists of a web application that allows for uploading a file and for creating the visualization interactively. The format of this file is JSON, a widely used data format in support of browser communication. The input data first declares the existing nodes with all their information such as:

- group
- index
- name
- size

Then the edges are defined by means of the following information:

- sourceGroup & sourceIndex
- targetGroup & targetIndex
- name
- size

This information is sufficient to display the targeted visualization. After creation, it is possible to optimize the visualization according to specific requirements by using various modes of interaction:

- enlarging/reducing the size of nodes and links;
- enlarging/reducing the size of node and link labels;
- filtering links by referring to specific values;
- filtering groups;

- selecting nodes and displaying only adjacent edges.

4.5 Examples

This chapter illustrates two applications of PolyViz.

4.5.1 Topic distribution and referencing

In Uslu, Mehler, Niekler, et al. (2018) we analyzed the topic distribution and linkage of the German Wikipedia. For this purpose we developed a state-of-the-art DDC topic classifier and used it to categorize all articles of the German Wikipedia. The link structure of Wikipedia was used to analyze which topics refer to each other (inter-topic links) or are thematically closed (intra-topic links). We visualized this information by means of our visualization technique. Since topics in the DDC are hierarchically organized (10 topics on the 1st level, 99 on the 2nd and 915 on the 3rd), we referred to the 10 main topics to define the groups (paths) while the topics of the second level are taken to define the nodes on the paths. This results in a 10-partite graph as shown in Figure 4.2. This visualization states that a few topics dominate the German Wikipedia.

4.5.2 Sentence similarity

A second example is the usage of PolyViz within TextImager (Hemati, Uslu, and Mehler 2016). TextImager is a tool that performs various *natural language processing* (NLP) procedures and visualizes their results. One of these procedures concerns the computation of (semantic or structural) similarities between sentences. Figure 4.3 depicts the sentence similarities within a document as well as cross-document sentence similarities. Each document is given its own color and each sentence in a document is represented as a node. The thicker and stronger the links between the nodes, the more similar these sentences are. By this example, one can see which texts contain similar content – without reading the documents. One also sees that sentences of the same

text (arc diagram) tend to be more similar among each other than sentences of different texts. This usage scenario is in support of distant reading.

Figure 4.3: Example of a 9-partite graph showing similarities of documents on the sentence level.

4.6 Conclusion

We present PolyViz, a new and interactive visualization system that allows for depicting a special kind of multipartite graphs. While in conventional multipartite graphs it is not allowed to have edges among members of the same group, we allow this case in order to increase the number of use cases. We additionally introduced the web application that can be used to create instances of our visualization technique and to adapt it. Further, we exemplified PolyViz by means of two use cases. Since the range of applications is wide, we provide this visualization technique open source. Currently, the visualization works for k-partite graphs with k greater than 2. In future work we want to offer the

same advantages also for bipartite and for 1-partite graphs, where the latter is equivalent to a single arc diagram.

5 SemioGraph: Introducing Multi-Codal Graphs by Example of Word Embeddings

The content of this chapter has been submitted to the Digital Humanities 2020 conference and is awaiting review.

5.1 Abstract

In this article, we introduce SemioGraphs (alias *multicodal graphs*), that is, graphs whose vertices and edges are simultaneously mapped onto different systems of types or labels. To this end, we present a technique for visualizing SemioGraphs in an interactive manner. SemioGraphs aim at coding as much information as possible within the same graph visualization. This is required for displaying word networks for which one has to visualize different information units such as POS, node weight, node salience, node centrality etc. To showcase SemioGraphs, we refer to word embedding networks. Word embeddings have become indispensable in the field of NLP, as they allow for significant improvement in many machine learning tasks. The paper additionally describes the SemioGraph website that we built to facilitate the analysis of pre-trained word embedding models based on SemioGraph.

5.2 Introduction

In this article, we introduce *multicodal graphs* henceforth called SemioGraphs, that is, graphs whose vertices and edges are simultaneously mapped onto different systems (or codes) of types or labels. To this end, we present a technique for visualizing SemioGraphs in which information units of different provenance can be interactively browsed within the same visualization. As an application scenario for exemplifying this technique, we utilize word embedding networks. Word embeddings have become indispensable in the field of natural language processing, as they allow for significant improvement in many machine learning tasks. In our application scenario we experiment with a range of different embeddings that have been computed for a set of different training corpora. The aim is to demonstrate the expressiveness of our technique for visualizing SemioGraphs that even allow for interactively comparing different lexical neighborhoods of the same seed word. The paper describes the functional spectrum of our visualization technique for SemioGraphs. This is done by means of the SemioGraph website on which users have free access to our implementation of SemioGraphs in the context of visualizing word embeddings. In this sense, the paper contributes to the further development of word embeddings and their utilization in digital humanities by providing a visualization technique that allows for comparing word embeddings across different corpora and being based on different techniques in an interactive manner.

5.3 Related Work

There have been many attempts to visualize word embeddings in order to analyze their quality. Most visualizations of word embeddings are an approximate representation of these vectors. This means that the vectors are reduced from several hundred dimensions to only two dimensions using dimension reduction methods (Smilkov et al. 2016; Maaten and Hinton 2008). In the case of a reduction to a two-dimensional space, the resulting dimensions are then displayed as the x- and y-axis. However, in this way a lot of information is lost and the visualizations becomes less accurate.

There are also approaches, who create a word embedding network using the

k-nearest neighbors or neighbors with a cosine similarity higher than a certain threshold (Gyllensten and Sahlgren 2015; Beelen 2015; Goldhahn, Eckart, and Quasthoff 2012). In SemioGraph we have implemented both possibilities and let the user decide which type to use or even to combine. In addition, we offer interaction functions that enable further analyses.

For the training of word embeddings we use the following approaches: Mikolov, K. Chen, et al. (2013), Pennington, Socher, and C. Manning (2014), Ling et al. (2015), Komninos and Manandhar (2016), and Levy and Goldberg (2014). We have trained them with different corpora such as Wikipedia, newspaper articles (Sueddeutsche Zeitung 2018) and literature from Project Gutenberg (2018). This makes it possible to analyze the differences and similarities of the methods depending on different corpora.

5.4 SemioGraph: Features and Parameters

For the initial creation of SemioGraphs we use *force-graph* (Asturiano 2018), a framework to represent a graph data structure in a 2-dimensional canvas using a force-directed iterative layout algorithm. In order to informationally enrich this structure, we expand it in several ways.

In a SemioGraph a node can represent several numerical parameters. This is achieved, by encoding information into its height, width and transparency. Thus, unlike traditional graph visualizations, we do not use circles to represent nodes, but ellipses, which can vary in height and width (see Figure 5.1). The color of a node encodes its membership to certain groups or classes. In addition, we provide a special mode that transforms the nodes into pie charts and can thus encode multiple classifications and their membership values. This even allows us to display class membership distributions at node level. Each node can have a specific label, which is displayed next to the node. The line color and thickness of a node can also be configured to encode certain information.

Concerning the edges of a SemioGraph, we also encode and display various information units. Here we refer to the thickness and transparency of the edges to display numerical information. Furthermore, the color of the edges can be used to represent different edge classes. In addition, the edges can

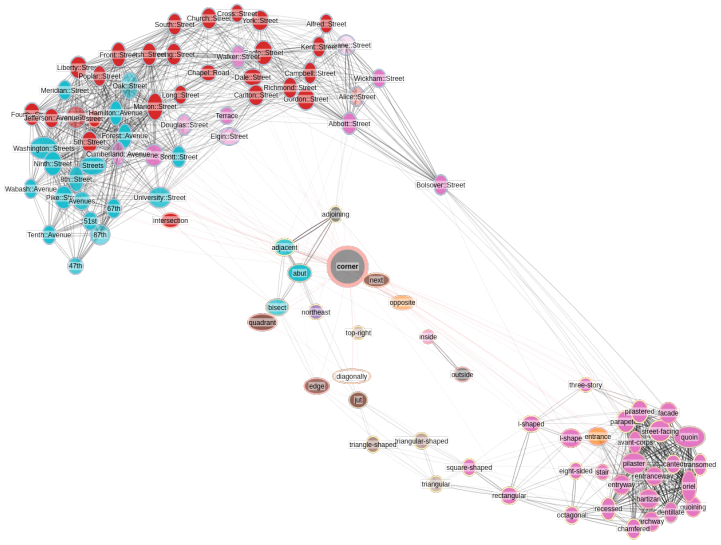


Figure 5.1: Example of an SemioGraph.

obtain arrows in a directional graph.

Various interaction options are available to further refine the graph analysis. One can hover, for example, over a node or an edge to perform a function. Usually, a tool-tip is created to provide further information. With various buttons, checkboxes and sliders, it is possible to filter a SemioGraph according to user criteria even after its creation. We have also implemented various node click functions, which further improve the analysis (see Section 4).

With this functional spectrum it is possible to visualize, analyze and interact with highly complex (directed and undirected) graphs. The next section describes how these functionalities were used in application.

5.5 Use Case

We visualize word embedding networks as a use case scenario of SemioGraph and provide the SemioGraph website to make our approach reusable. Using this website, users can explore all features of SemioGraph in order to analyze and compare different procedures for computing word embeddings and the impact of the underlying corpora. We generated the word embedding networks using k nearest neighbors per seed word. In order not to be limited to a certain number of nearest neighbours, we have made this parameter adjustable. The

user can also set a threshold for the minimum cosine similarity of words linked by an edge. In this way one is not limited to one of the two possibilities and can even combine them.

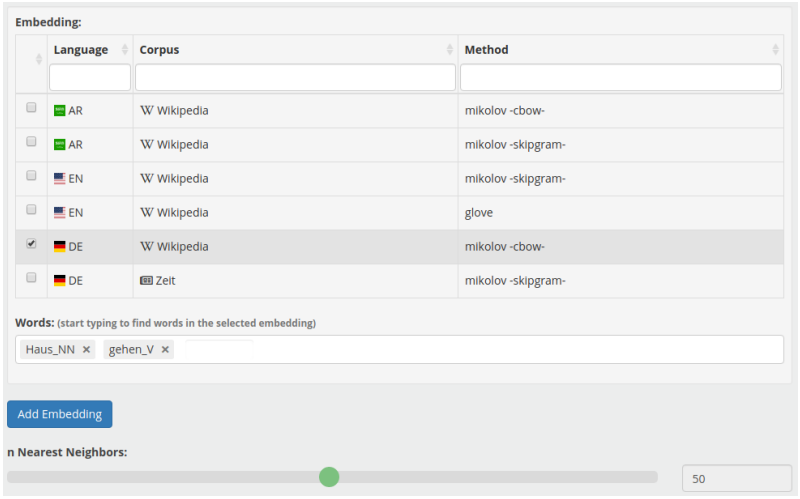


Figure 5.2: Input panel of the SemioGraph website.

First, the pre-trained embedding model to be analyzed must be selected in the input panel of the SemioGraph website (see Figure 5.2). These can be filtered according to their language, corpus and method. Using a slider one can select how many of the k nearest neighbours should be considered for creating the network. Subsequently, one can select the seed word to be analyzed by means of a auto-complete text field. It is also possible to select multiple seed words and embedding models. When selecting multiple models, a separate SemioGraph window is created for each model. In this way, the generated graphs can be compared to each other. To exemplify the visualizations in this paper we have chosen the seed word `corner` and display its word embeddings based on *word2vec* in comparison to *GloVe*, both trained on the English Wikipedia. The resulting visualization is shown in Figure 5.3.

We have attempted to use all the features of Section 3 to display as much information as possible. Starting with the nodes, we have coded the similarity to the seed word using the width of the node. The height of a node then corresponds to the number of its links (node degree). This allows a user to quickly identify which nodes are similar to the seed word, but are also strongly connected. We used text2ddc (Uslu, Mehler, Niekler, et al. 2018) to detect the topic distribution of the words used in the underlying corpus. The nodes are

colored according to the best predicted DDC (Dewey Decimal Classification) class. The transparency of the nodes expresses the probability by which the corresponding class is assigned to the word (i.e. the membership value of the word to the class). The part of speech of a word is represented by the border color of the node. The border width emphasizes the seed word, making it is easy to find.

Concerning the edges, we consider directed links. Since the graph is complex enough, we coded the similarity between two nodes using the thickness and transparency of the edges. By using two visualization effects for the same value, this value is even more recognizable. The edge color indicates whether the edge is connected to the seed word (red) or not (black).

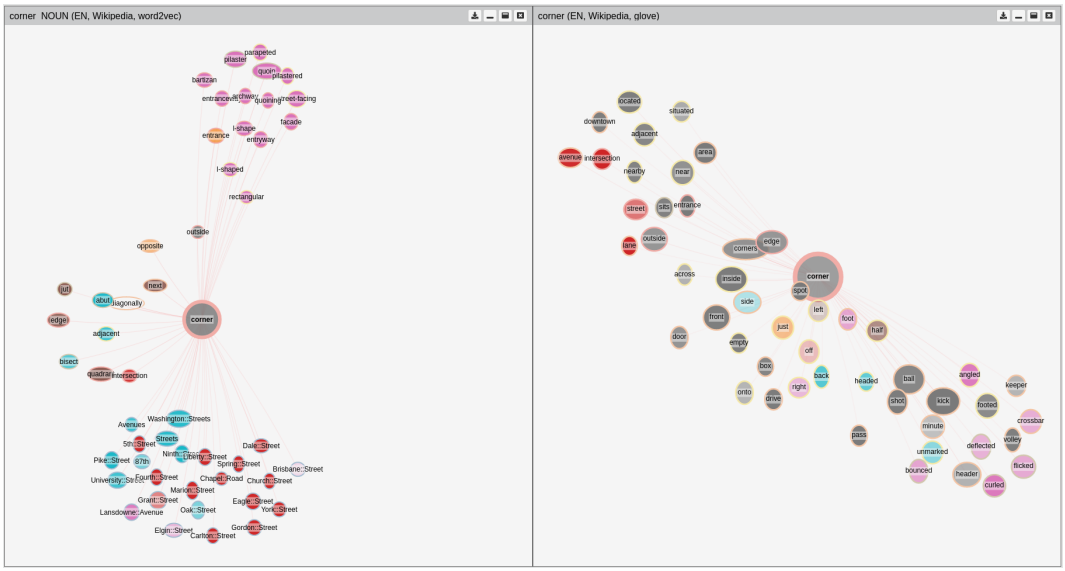


Figure 5.3: Comparison of two different embedding methods in the example of `corner`.

This is all the static information we can display with the visualization features in use. However, using the interaction features of SemioGraph, the network can be examined in more detail. By hovering over a node or an edge, a tooltip appears which provides more information. We implemented various click functions to enhance the graph analysis. A regular click on a node highlights all adjacent edges and nodes. If multiple windows are displayed, the selected word and its adjacent nodes and edges are also highlighted. When clicking on a node in combination with the *Ctrl*-key, the nearest k neighbors of the selected node are included in the visualization. This allows the comparison

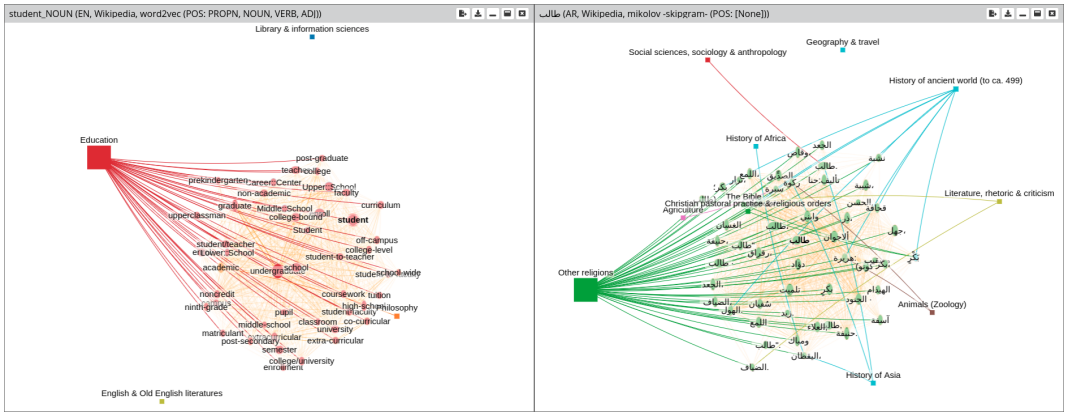


Figure 5.4: *DDC classes as nodes* - comparison of two languages (EN and AR) models in the example of `student`.

of two words in the same semantic space. Clicking on a node in combination with the *Alt*-key replaces the visualization with the k nearest neighbours of the selected node.

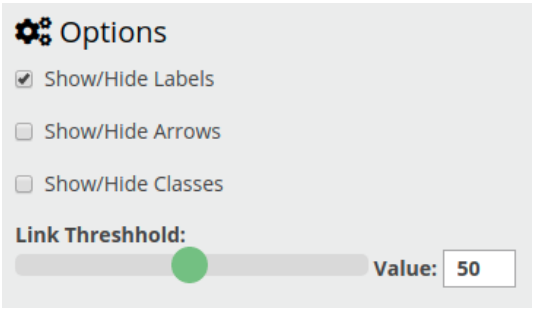


Figure 5.5: Options panel of the SemioGraph website.

In our *corner* example in Figure 5.3 we can see clear differences between *word2vec* and *GloVe*. While word2vec (left SemioGraph) creates the two main clusters with the DDC-class *Building and Construction* (e.g. entrance, street-facing, facade, l-shape, etc.) and *street names* (e.g. Washington Streets, 5th Street, Avenues, 87th, etc.), GloVe (right SemioGraph) contains DDC-classes like *Sport* (e.g. ball, kick, shot, header, etc.) and *Architecture and Area planning* (e.g. downtown, nearby, area, located, etc.) – though both are based on the same corpus.

In Figure 5.4 we demonstrate another example in which we compare the word embedding SemioGraphs derived for the seed word *student* and its translation in Arabic.

In this example we have selected the option *DDC classes as nodes*. By enabling this option, each DDC class gets its own square node. The size of these nodes depends on the number of times this class is assigned to the nearest neighbors. In the example of *student* one can see that the English seed word and its Arabic translation have different topical meanings in these two languages (EN and AR). In the English Wikipedia model, *Education* is the most dominant DDC topic, while in the Arabic Wikipedia model it is *Religion*. This is the case because the *student* in Arabic (talibé) is a boy studying the Quran. This semantic difference becomes directly visible by means of the thematic coloring provided by SemioGraph.

By using the option panel (see Figure 5.5), the visualization can be further optimized to meet the user requirements. Edges below a certain threshold value can be filtered out using a slider. Using the checkboxes, it is possible to enable and disable node labels and edge arrows. We also have the option to display the DDC class distribution as a pie chart for each node (see Figure 5.6).

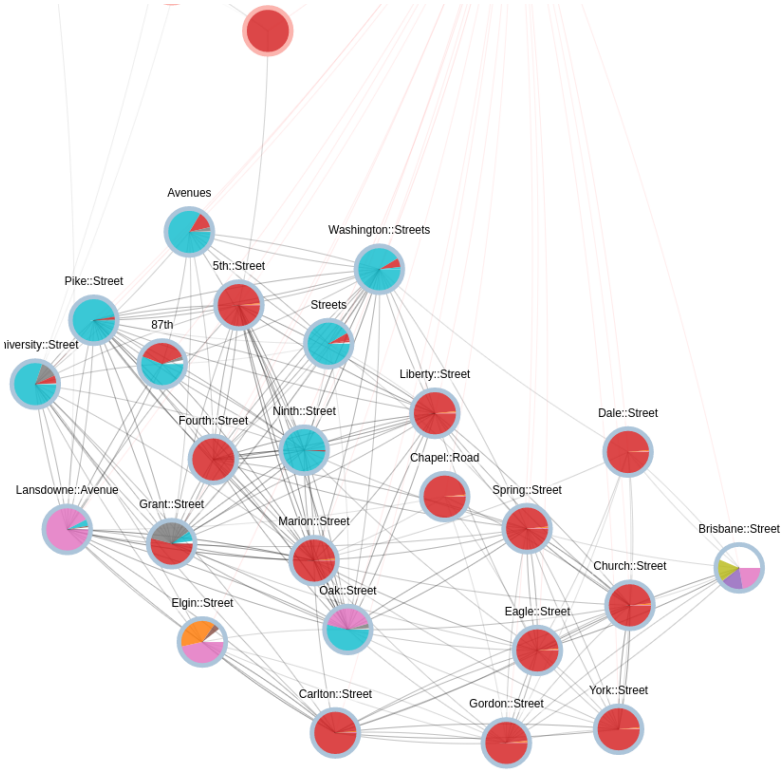


Figure 5.6: Pie charts representing DDC-classes of each node.

5.6 Conclusion

We introduced a novel visualization technique, called SemioGraph, for depicting multicodal graphs. To demonstrate all features of SemioGraph, we prepared a use case application in which we visualized word embedding networks. We showed that different models can lead to completely different graphs and thus may help in finding word embeddings for specific NLP-tasks. In future work we want to provide even more features and interaction possibilities for SemioGraph. SemioGraph will be made available open source, so that everyone has the opportunity to present his own graph data using our technique.

6 text2voronoi: An Image-driven Approach to Differential Diagnosis

The content of this chapter (Mehler, Uslu, and Hemati 2016) was published in Proceedings of the 5th Workshop on Vision and Language (VL'16) hosted by the 54th Annual Meeting of the Association for Computational Linguistics (ACL).

6.1 Abstract

Differential diagnosis aims at distinguishing between diseases causing similar symptoms. This is exemplified by epilepsies and dissociative disorders. Recently, it has been shown that linguistic features of physician-patient talks allow for differentiating between these two diseases. Since this method relies on trained linguists, it is not suitable for daily use. In this paper, we introduce a novel approach, called text2voronoi, for utilizing the paradigm of text visualization to reconstruct differential diagnosis as a task of text categorization. In line with current research on linguistic differential diagnosis, we explore linguistic characteristics of physician-patient talks to span our feature space. However, unlike standard approaches to categorization, we do not use linguistic feature spaces directly, but explore visual features derived from the talks' pictorial representations. That is, we provide an approach to *image-driven differential diagnosis*. By example of 24 talks of epileptics and dissociatively disordered patients, we show that our approach outperforms its counterpart based on the bag-of-words model.

6.2 Introduction

Physicians already use medical imaging for diagnosis: bone fractures, for example, are visualized by radiographs, pregnancies are examined by means of ultrasound scans, while neurological disorders are studied with the help of MRI scans. Our goal is to assist physicians in diagnosing mental disorders by analogy to such image-driven methods. To this end, we introduce a method for scanning physician-patient talks to get pictorial representations as input of classifiers which perform the differential diagnosis. This approach is in line with recent efforts in clinical NLP to utilize computational methods for automatically analyzing medical histories (Friedman, Rindflesch, and Corn 2013). It profits from recent findings showing that linguistic features provide reliable bases for differentiating between epilepsies and dissociative disorders (Gülich 2010; Reuber et al. 2009; Opp, Job, and Knerich 2015). Since the latter approach relies on trained linguists for performing the feature analysis it does not allow for daily use. The present paper aims at filling this gap. It introduces a new method for visualizing linguistic data by means of images as input to classifiers which learn from their pictorial features to arrive at the desired diagnoses. The main hypothesis of our paper runs as follows: *Linguistic features of physician-patient talks can be visualized in a way that a certain range of diagnoses can be derived from analyzing pictorial features of these visualizations.* We introduce so called *Voronoi diagrams of Text*s (VoTe) to provide such expressive visualizations. VoTes are generated by our text2voronoi algorithm as described in Section 6.4. Unlike the classical bag-of-words model, this approach explores *bags* of visual features derived from the talks' image representations in terms of VoTes. To this end, we utilize TextImager which automatically extracts a wide range of linguistic information from input texts to derive representational images thereof. In Section 6.5, we describe an experiment, which shows that our image-driven classifier can indeed differentiate between epilepsies and dissociative disorders: its F-score outperforms its classical counterpart based on the bag-of-words model. Note that we do not claim that VoTes allow for differentiating between whatever mental diseases. Rather, we start with epilepsies and dissociative disorders as two initial examples and will extend our approach by including related diseases in future work (cf. Section 6.6).

6.3 Related Work

Recent studies have shown that a linguistic examination of physician-patient talks based on *Conversation Analysis* (CA) (Drew, Chatwin, and Collins 2001) allows for distinguishing between epileptic and non-epileptic seizures (Reuber et al. 2009; Plug, Sharrack, and Reuber 2009; Plug, Sharrack, and Reuber 2010; Gülich 2010; Opp, Job, and Knerich 2015). Reuber et al. (2009) describe a CA-inspired experiment where two linguists blinded to medical data attempted to predict the diagnosis on the basis of qualitative linguistic assessments. Using these assessments, the linguists predicted 17 of 20 (85%) diagnoses correctly. Opp et al. (2015) found that patients with epileptic seizures try to describe their attacks as accurate as possible, whereas patients suffering from dissociative disorders avoid detailed descriptions of their seizures. As a matter of fact, such differences are mirrored by linguistic choices. However, these and related methods (Gülich 2010) rely on the expertise of trained linguists and are, thus, not practical in terms of daily use.

Other approaches use machine learning to predict diagnoses from therapy transcripts by means of extracted linguistic features (Howes, Matt Purver, et al. 2012). Howes et al. (2013), for example, use topics that have been derived by means of LDA. Support vector machines operating on linguistic features have also been used to predict diagnoses (Howes, Matthew Purver, McCabe, et al. 2012; DeVault, Georgila, et al. 2013; DeVault, Artstein, et al. 2014). Unlike these approaches to text categorization, which rely on the bag-of-words model or some of its descendants, we use pictorial representations of linguistic features as input for our classifier. This is done by extending the UIMA-based TextImager by means of visual scans of physician-patient talks as explained in Section 6.4. Alternatives to TextImager are given by the UIMA-based frameworks cTAKES (Savova et al. 2010) and EpiDEA (L. Cui et al. 2012). Unlike TextImager, both tools do not provide a visualization engine and, thus, do not fit our task of text classification based on pictorial text representations.

Note that the pictorial representations of texts as introduced here rely on so called Voronoi diagrams (Berg et al. 2000). Voronoi diagrams have already been used to represent semantic structures of lexical units (Jäger 2006). We further develop this approach in the sense of deriving Voronoi diagrams as representations of natural language texts in general.

6.4 The text2voronoi Model of Texts

Our goal is to generate images from physician-patient talks whose visual features can be used by classifiers to perform the desired differential diagnosis. To this end, we provide the *text2voronoi algorithm* which computes this visualization in four steps (see Figure 6.1):

1. extraction of linguistic features,
2. embedding the features in vector space,
3. Voronoi tesselation of this space and
4. extraction of visual features from the tesselation.

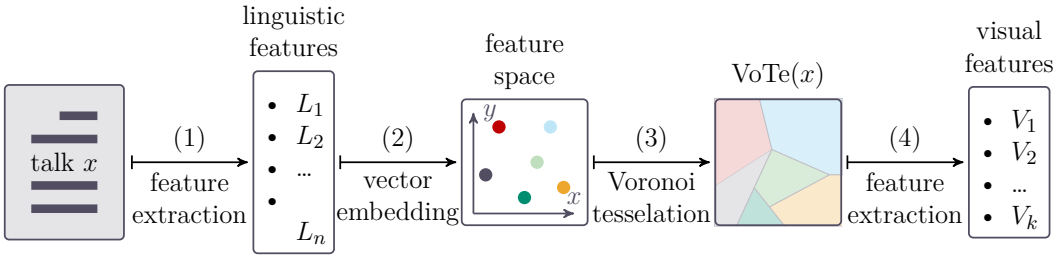


Figure 6.1: Workflow of the text2voronoi algorithm generating a *Voronoi diagram of* the *Text* (VoTe) x.

In what follows, we describe each of these steps.

Label	POS
C1	Noun
C2	Verb
C3	Preposition
C4	Adjective
C5	Adverb
C6	Temporal expression

Table 6.1: Parts of speech and expressions explored by text2voronoi.

6.4.1 Linguistic Feature Extraction

Each input text is pre-processed by TextImager which utilizes several NLP tools to tag a range of linguistic features per lexical token. This includes

Label	Category	Example
G1	Case	{nominative, accusative,..}
G2	Mood	{indicative, imperative,..}
G3	Number	{singular, plural}
G4	Person	{first, second,..}
G5	Tense	{past, present,..}
G6	Gender	{feminine, masculine,..}
G7	Degree	{positive, comparative,..}

Table 6.2: Categories explored by text2voronoi.

POS tags (e.g., pronouns, prepositions), grammatical categories (e.g., case, gender, number, tense) and temporal expressions (e.g., dates, temporal adverbs) – see Table 6.1 and 6.2 for all POS and their features considered in Step 1 combining to 180 features. The reason for selecting these features is that according to Gülich (2010) and Opp, Job, and Knerich (2015), patients suffering from epilepsies tend to give detailed descriptions of their seizures, while dissociatively disordered patients tend to avoid such descriptions. Thus, while the former group of patients likely uses personal pronouns in connection with prepositions (for localizing their seizures) and polarity cues (for evaluating them), the latter group will rather avoid the co-selection of such features. For tagging POS and grammatical features, we use a retrained instance (Eger, Rüdiger Gleim, and Mehler 2016) of *MarMoT* (Müller, Schmid, and Schütze 2013), while *HeidelTime* (Strötgen and Gertz 2010) is used for tagging temporal expressions.

6.4.2 Embedding the Features in Vector Space

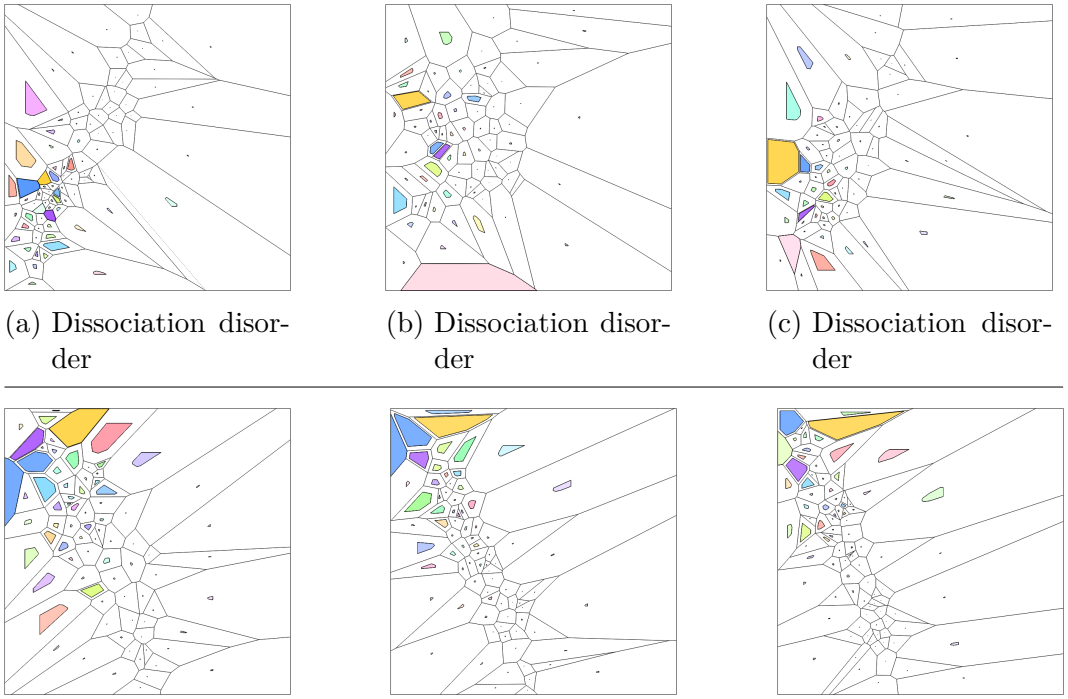
Since our features are tagged per token, we can transcode each sentence of the corresponding input text as a sequence of these features to make them as input to word2vec (Mikolov, Sutskever, et al. 2013) by projecting on exactly two dimensions. The reason behind this approach is to compute feature associations in a manner that is characteristic of the input text. Thus, we do not use a (huge) reference corpus (e.g., Wikipedia) for computing "reference" associations but explore text-*specific* patterns in our two-dimensional feature space.

6.4.3 Voronoi Tesselation of the Feature Space

The vector embeddings span a two-dimensional space for which we compute a Voronoi decomposition (Berg et al. 2000). Each cell of the resulting *Voronoi diagram of* a *Text* (VoTe) corresponds to a single feature. Generally speaking, starting from a set P of distinct points in a plane we get a corresponding Voronoi diagram by coloring all points $q_1, \ldots$ of equal distance to at least two points in P (Berg et al. 2000). The points $q_1, \ldots$ manifest the borders of the Voronoi regions that consist of all points with the same single nearest neighbor in P. To color the VoTe of a text, we additionally explore two kinds of frequency information: while the overall frequency of a feature determines how much of its cell is filled (starting from the center), the transparency of the cell depends on the feature's inverse sentence frequency: the smaller this value, the more transparent the cell. Fig. 6.2 exemplifies the VoTes of 6 texts. Note that for each text each feature is mapped onto the same color in order to allow for comparing different texts. However, the exact position of a feature cell in a text's VoTe, its size, degree of filling, transparency and neighborhood depend on the specifics of that text. That is, they depend on the characteristics of the given physician-patient talk in terms of the co-occurrence statistics of the underlying linguistic features. Thus, our classification hypothesis is: *talks of patients suffering from the same disease induce similar VoTes.* Exploring the visual patterns of VoTes is then a way to perform the targeted classification.

6.4.4 Extracting Visual Features from VoTes

For the sake of the latter classification, we extract a set of visual features for each cell of the VoTes (see Table 6.3). The underlying hypothesis is that two VoTes are the more similar, the more of their equally colored cells share similar visual features. Each cell is characterized (1) by its *gestalt* (area, corner, filling, shape, transparency), (2) *location* (position, shape) and (3) *neighborhood* (centrality). While the first group of features informs about how a single cell looks like, the second group informs about its localization on the map, and the third group about its relations to other cells. The more of these features are shared by two equally colored cells, the more visually similar they are. For



(a) Dissociation disorder (b) Dissociation disorder (c) Dissociation disorder

(d) Epilepsy (e) Epilepsy (f) Epilepsy

Figure 6.2: Visualizations (*VoTes*) of six physician-patient talks as used in our classification experiment.

mapping neighborhood-related features, we compute the closeness centralities of the cells in the graph representation of the Voronoi diagrams. Next, for all Voronoi cells that correspond to the 180 features of Step 1, we compute 11 features (see Table 6.3) so that each VoTe of a text is finally mapped onto a vector of 1980 visual features. Note that if a linguistic feature did not occur in a talk, it was mapped onto a null vector so that VoTes get also comparable for commonly absent features.

Feature	**Description**	**#Features**
Area	The surface area	1
Position	x/y coordinates of center	2
Shape	Min (x, y), max (x, y)	4
Filling	Percentage of fill coverage	1
Transparency	Degree of opacity	1
Corner	Number of corners	1
Centrality	Closeness centrality	1

Table 6.3: Visual features of the cells of a Voronoi tesselation (VoTe) explored by text2voronoi.

6.5 Experiment

This section provides experimental data on testing the text2voronoi model. To this end, we use a German corpus of 24 physician-patient talks of 12 epileptics and 12 dissociatively disordered patients. The talks were transcribed according to GAT2 (Selting et al. 2009) and annotated w.r.t. turns and seizure descriptions (Gülich 2010; Opp, Job, and Knerich 2015). The corpus was further processed according to Section 6.4 so that each talk was mapped onto a vector of 1980 visual features. Finally, the vectors were independently made input to *SVMlight* and *LIBSVM* to compute F-scores based on a leave-one-out cross-validation. Using all features, both kernels (linear and RBF) achieve an F-score of 83.2% – see Table 6.4. Next, we performed an optimal feature selection for SVMs (M. H. Nguyen and De la Torre 2010) using a genetic search on our feature space with the aim of optimizing F-scores based on the same setting of cross-validation.

This optimization resulted in a perfect classification (see Table 6.4) regardless of the kernel and the implementation of SVMs in use. Finally, we computed a bag-of-words model based on the lexical data of all talks in our corpus. Using an RBF kernel (leave-on-out cross-validation) this model achieved an F-score of 69% (see Table 6.5); a search for an optimal feature subset raised this score to 91% (by means of a linear kernel).

Features	**Kernel**	**nu-SVC**	**C-SVC**	**SVM light**
All	Linear	0.832	0.832	0.832
Subset	Linear	1.0	1.0	1.0
All	RBF	0.832	0.832	0.832
Subset	RBF	0.958	1.0	1.0

Table 6.4: F-scores of text2voronoi-based classification.

Features	**Linear kernel**	**RBF kernel**
All	0.60	0.69
Subset	0.91	0.82

Table 6.5: F-scores of the bag-of-words model.

6.5.1 Discussion

Obviously, our findings are independent of the kernels (linear or RBF) and the SVM implementations in use. They show that by example of our corpus data, differential diagnoses come into reach based on visual depictions of the underlying talks. Moreover, we show that an optimal feature selection for SVMs can boost the classifier enormously. This may hint at problems of overfitting (negative interpretation) or at the expressiveness of the visual features in use (positive interpretation). Evidently, our corpus data is too small to decide between these alternatives. Thus, further research is required that starts from larger corpora of physician-patient talks. As a matter of fact, such data is extremely difficult to obtain (Friedman, Rindflesch, and Corn 2013) so that comparative studies have to be considered in related areas of more easily accessible data. However, as indicated by our F-scores and as exemplified by Figure 6.2, our VoTe representations of texts are seemingly informative enough to provide visual depictions of text that may be used by physicians as scans of neurologically disordered patients based on their medical histories. Based on our results, we may speak of a novel approach to text representation according to which symbolically coded information in texts is visually reconstructed in a way that allows for performing text operations (in our case *text classification*) indirectly by processing the resulting visual representations.

6.6 Conclusion

We presented a novel approach to image-driven text classification based on Voronoi tesselations of linguistic features spaces. Our method allows for high score differential diagnoses by exploring features of the pictorial representations of physician-patient talks. Our experiments show that this approach outperforms classifiers based on the bag-of-words models. In order to further test its validity, we plan to experiment with larger corpora and various tasks in text classification (e.g., authorship attribution and genre detection). A major reason to do this is to clarify whether the F-scores reached by our approach so far reflect overfitting or not. To this end, we will also experiment with data of different languages. Moreover, since a great deal of information about the correct diagnosis relates to whether a patient tends to suppress the memory of her

or his seizures, polarity cues are promising candidates for extending our feature space. However, since we deal with *seizure* descriptions, such a distinction is rather challenging. The reason is that turns of patients about seizures have very likely negative connotations. An alternative is to consider simpler quantitative features (turn length, number of turns etc.) to simplify the generation of VoTes. This is needed to enable automatic differential diagnoses instantaneously during physician-patient talks, which – because of error-prone speech recognition systems – require easy to measure features. Obviously, this requirement implies a trade-off: the more easily a feature is measured, the lower its semantic specificity with respect to the target classes to be learnt. Thus, a great deal of progress may be expected by developing speech recognition systems that focus on expressive linguistic features especially of physician-patient talks. Last but not least, we may consider quantitative characteristics that are more closely related to the geometry of Voronoi diagrams (e.g., in terms of their order and size – cf. Berg et al. (2000)). In this way, we want to contribute to the further development of text representation models based on text visualizations.

7 LitViz: Visualizing Literary Data by Means of text2voronoi

The content of this chapter (Uslu, Mehler, and Meyer 2018) was published in Proceedings of the Digital Humanities 2018.

7.1 Abstract

We present LitViz, a webbased tool for visualizing literary data which utilizes the text2voronoi algorithm (Mehler, Uslu, and Hemati 2016) to map natural language texts onto Voronoi diagrams. These diagrams can be used, for example, to visually differentiate between (groups of) authors. text2voronoi utilizes the paradigm of text visualization to reconstruct text classification (e.g., authorship attribution) as a task of image classification. This means that, in contrast to conventional approaches to text classifiction, we do not directly use linguistic features, but explore visual features derived from the texts' visualizations to perform operations on texts. We illustrate LitViz by means of 18 authors, each of whom is represented by 5 literary works.

7.2 Introduction

In this paper we present a new tool, called LitViz, for the visual depiction of literary works. To this end, we utilize the text2voronoi algorithm (Mehler, Uslu, and Hemati 2016) which maps natural language texts to image representations. The idea is to generate images of texts which can be used instead of these texts' symbolic information to characterize them, for example, in terms

of authorship, topic or genre. text2voronoi is in line with the paradigm of text visualization to reconstruct text classification (e.g., authorship attribution) as a task of image classification. In contrast to conventional approaches to text classification, we therefore do not directly use linguistic features, but explore visual features derived from the texts' visualizations in order to identify, for example, their authors. We exemplify LitViz by means of 18 authors, each with 5 literary works.

LitViz allows for interacting with the visualizations of these works in two modes: two- and three-dimensionally (see Figure 7.1 and 7.2).

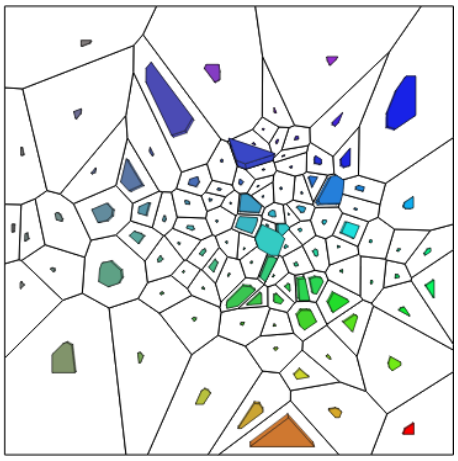


Figure 7.1: Visual depiction of E.T.A. Hoffmann's *Das steinerne Herz*

7.3 Related Work

The idea of visualizing literature was inspired by Martin Wattenberg's *The Shape of Song*[1] (Wattenberg 2001; Wattenberg 2002). Wattenberg explores identical or otherwise repetitive passages of a composition to visually depict them. This is done by means of semicircles, which combine repeated and repetitive positions in such a way that the micro- and macro-structure of a composition becomes visible. Our idea is to transpose this idea to the visualization of literary data.

Kucher and Kerren (2015) give an overview of state-of-the-art techniques of text visualization and present a website that allows for differentiating between

[1] `http://turbulence.org/Works/song/gallery/gallery.html`

these techniques.

Cao and W. Cui (2016) provide a systematic review of many advanced visualization techniques and discuss the fundamental notion of information visualization.

Mehler, Rüdiger Gleim, Brück, et al. (2016) present a web tool called Wikidition which allows for automatically generating large-scale editions of text corpora. This is done by using multiple text mining tools for automatically linking lexical, sentential and textual data. The output is stored and visualized using a MediaWiki. Thus, any Wikidition is extensible by its readers based on the wiki principle.

Rockwell and Sinclair (2016) present a detailed web tool, called Voyant tools, for visualizing texts. Unlike Voyant, our focus is on non-standard techniques of visualizing textual data that go beyond histograms, scatterplots, line charts and related tools.

Generally speaking, text visualization supports distant reading as introduced and exemplified by Moretti (2013), Rule, Cointet, and Bearman (2015) and Michel et al. (2011). These approaches show how visualizations that support distant reading could look like to get an overview of the documents by just looking at the final visualizations. LitViz is a tool following this tradition: it utilizes text2voronoi to extend the set of techniques mapping textual data. In this way, it combines Wattenberg's approach with distant reading techniques from the point of view of text visualization.

7.4 Model

Our goal is to generate images from literary works in a way that text classifiers can be fed by the features of these iconic representations in order to perform classification experiments, for which usually linguistic features are explored. This is the task of the text2voronoi algorithm, which calculates image representations of texts in four steps (Mehler, Uslu, and Hemati 2016): In the first step, the input text is analyzed by means of TextImager (Hemati, Uslu, and Mehler 2016) to extract linguistic features in the usual way, that is, features, spanning a vector space of linguistic data. In the second step, the resulting

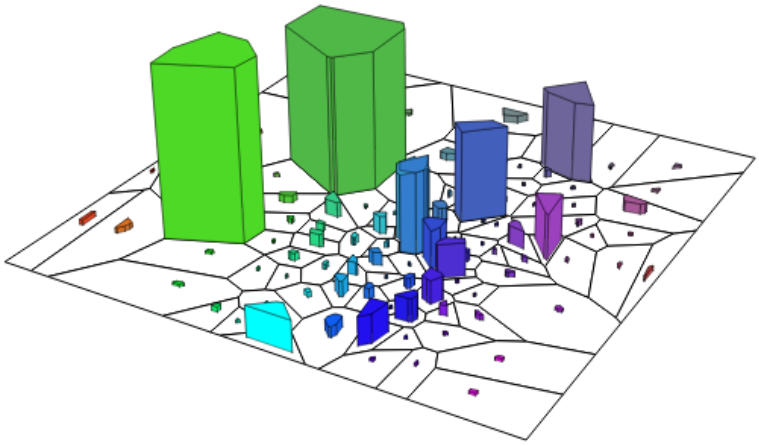


Figure 7.2: 3D visualization of Franz Kafka's *Der Kübelreiter*.

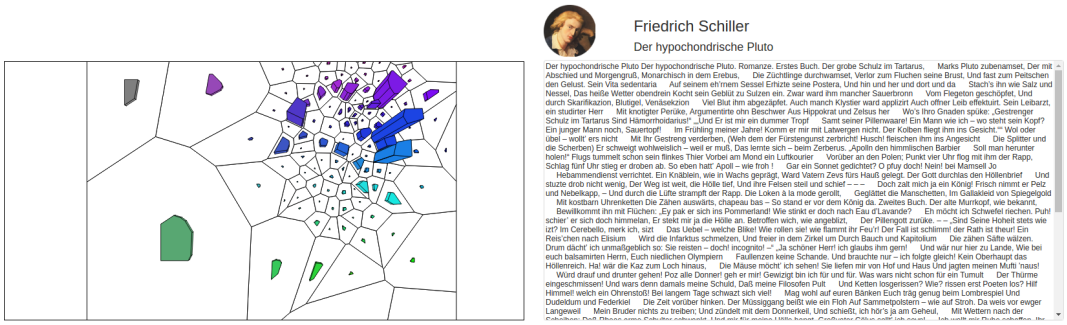


Figure 7.3: Front page of LitViz.

vector space is used to compute embeddings for each of the extracted linguistic features. Embeddings are produced by means of word2vec (Mikolov, K. Chen, et al. 2013). In the third step, a Voronoi tessellation of the embedded features is computed. As a result, each lexical feature is mapped onto a separate Voronoi cell whose neighborhood reflects the feature's syntagmatic and paradigmatic associations with other features of the same space. The topology of the Voronoi cells spans a Voronoi diagram that visually represents the input text. Each of these cells is characterized by its filling level, transparency and height (third dimension) thereby reflecting its co-occurrence statistics within the input text, while the position and size of a cell is determined by the embedding of the corresponding feature – for the mathematical details of this algorithm see Mehler, Uslu, and Hemati (2016). Finally, the text2voronoi algorithm extracts visual features from the Voronoi diagrams to feed classifiers performing classifications of the input texts.

LitViz utilizes the first three steps of this algorithm. Unlike the classical text2voronoi procedure, it does not address the final step of classification. Rather, it gives access to Voronoi diagrams of input texts via a two-dimensional graphical interface, which can be transformed into a three-dimensional one by means of user interaction. These two- and three-dimensional text representations can be used by the user of LitViz to interact with the underlying input texts in order to highlight single Voronoi cells, to change her or his reading perspective or to visually compare Voronoi diagrams of different texts. In this way, LitViz paves the way to a kind of a *comparative distant reading* by making the visual depictions of different texts accessible in an interactive manner.

7.5 The LitViz Tool

We have selected 18 authors of German literature, each represented by 5 literary works. The works are taken from the Project Gutenberg (`https://www.gutenberg.org/`) and visualized by means of the text2voronoi algorithm. These examples are made accessible by the front page of LitViz (see Figure 7.3). When hovering over a Voronoi cell of the Voronoi diagram of a sample work, information about the underlying linguistic feature represented by this cell is displayed. According to Mehler, Uslu, and Hemati (2016), we call these images *VoTes*: Voronoi diagram of a Text. LitViz presents VoTes via a graphical user interface for two- and three-dimensional interactive graphics. In this way, we go beyond Wattenberg's 2D depictions of musical pieces.

The second page (tab) of LitViz gives access to the comparison tool. Here the user first selects the number of VoTes to be compared. Then the user selects a subset of works of the authors to be compared. In the example in Figure 7.5, we compare four VoTes of two authors: two VoTes of two works of Heinrich Heine (top) and two VoTes of Heinrich Mann (bottom). It is easy to see that these VoTes fall into two classes, depending on the underlying authorship. Heinrich Mann's two VoTes are organized around a center that is composed of many small cells, while there is a small subgroup of peripheral cells that are large. In contrast to this, the two VoTes of Heinrich Heine do not display such a center and are more evenly distributed in terms of their size. It is a main task of

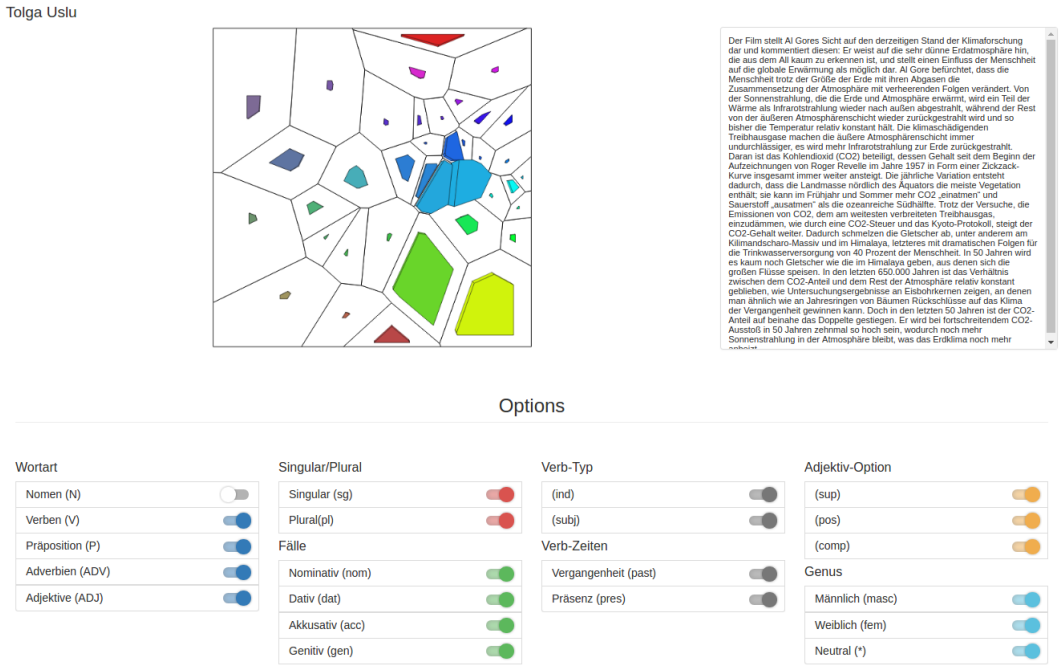


Figure 7.4: Custom VoTe with filter options.

LitViz to allow for such comparisons. In this way, that is, by interacting with the texts' image representations, the user can study single features and how they are related to other features of the same representational space.

Last but not least, LitViz provides a so-called custom tab. Here, the user can upload and visualize its own texts. It is then possible to set filter options using an option tool (see Figure 7.4) in order to further restrict the visualization.

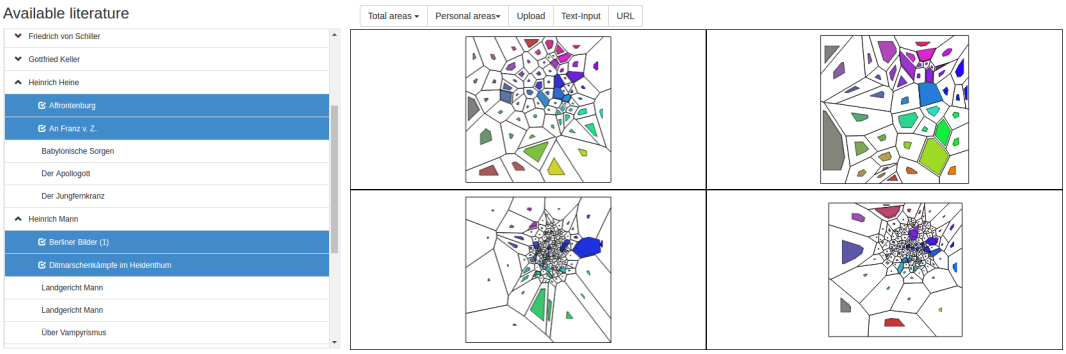


Figure 7.5: Comparison tool: Heinrich Heine (top) in comparison to Heinrich Mann (bottom).

7.6 Conclusion

We introduced a novel web tool, called LitViz, for visually depicting natural language texts based on the text2voronoi algorithm. LitViz enables the comparison of the visualizations of different texts. This allows, for example, for comparing the styles of the underlying authors *visually*. In this way, we extend the existing tool palette of distant reading.

8 Automatic Classification in Memory Clinic Patients and in Depressive Patients

The content of this chapter (Uslu, Miebach, et al. 2018) was published in Proceedings of Resources and ProcessIng of linguistic, para-linguistic and extra-linguistic Data from people with various forms of cognitive/psychiatric impairments (RaPID-2).

8.1 Abstract

In the past decade the preclinical stage of *Alzheimer's Disease* (AD) has become a major research focus. *Subjective cognitive decline* (SCD) is gaining attention as an important risk factor of AD-pathology in early stages of *mild-cognitive impairment* (MCI), preclinical AD and depression. In this context, neuropsychological assessments aim at detecting sorts of subtle cognitive decline. Automatic classification may help increasing the expressiveness of such assessments by selecting high-risk subjects in research settings. In this paper, we explore the use of neuropsychological data and interview based data designed to detect AD-related SCD in different clinical samples to classify patients through the implementation of machine learning algorithms. The aim is to explore the classificatory expressiveness of features derived from this data. To this end, we experiment with a sample of 23 memory-clinic patients, 21 depressive patients and 21 healthy-older controls. We use several classifiers, including SVMs and neural networks, to classify these patients using the above mentioned data. We reach a successful classification based on neuropsychological data as well as on cognitive complaint categories. Our analysis indicates

that a combination of these data should be preferred for classification, as we achieve an F-score above 90% in this case. We show that automatic classification using machine learning is a powerful approach that can be used to improve neuropsychological assessment.

8.2 Introduction

According to the world Alzheimer report, over 46 million people are estimated to have dementia. This number is expected to rise (Prince et al. 2015). Early detection and accurate diagnostic in preclinical stages is therefore of paramount importance. As an indicator of the earliest clinical stage of Alzheimers Disease (AD) subjective cognitive decline (SCD), defined as the individual's concerns related to cognitive functioning, is gaining interest in different settings (Jessen et al. 2014). With the growing interest in early diagnosis and early detection, SCD has been proposed as an established risk factor for AD, increased risk of future cognitive decline (Koppara et al. 2015) and abnormal AD biomarkers (Rebecca E. Amariglio et al. 2012; Chetelat et al. 2010; Wolfsgruber et al. 2015; Rachel F Buckley et al. 2017). However, in older community based samples the prevalence of memory concerns varies from 25-50% (Jonker, Geerlings, and Schmand 2000) which made it difficult to distinguish AD-related cognitive complaints from those related to normal aging. Furthermore, subjective cognitive decline (SCD) is reported in the context of depression (Balash et al. 2013) and has been positively associated with SCD in different samples (R. Buckley et al. 2013; Benito-León et al. 2010). Some researcher therefore argued that SCD is mainly driven by depressive symptomatology than being an indicator of an underlying AD-pathology. Current investigations try to refine the assessment of SCD with the aim to find AD-like complaints and those which may be more representative of a mood disorder or of aging in general (Molinuevo et al. 2016; Rabin et al. 2015). In line with the problematic assessment of SCD, some common-used neuropsychological screening tests such as the *Mini-Mental State Examination* (MMSE) are not sensitive enough for a reliable detection of subtle impairments presented in patients with *mild cognitive impairment* (MCI). Even when some results suggest specific types of neuropsychological deficits associated with *Major depressive Disorders* (MDD), it is still challenging for clinicians to differentiate subjective complaints as a re-

sult of a depressive symptomatology from cognitive complaints in the context of preclinical and prodromal AD (Zihl et al. 2010). In memory-clinic settings, early detection of AD is time consuming and requires multiple cost intensive information (e.g. neuropsychological testing including subjective concerns and objective impairment, detailed medical history and neurological examination) as well as clinicians with a certain level of expertise. Current assessments of subjective cognitive decline are unable to capture all aspects of SCD specific for preclinical AD and could potentially confound results in the SCD field. Recently, studies started to compare specific aspects of cognitive complaints in different samples using qualitative interview based approaches (Rachel F. Buckley et al. 2015; Miebach, Wolfsgruber, Frommann, R. Buckley, et al. 2017; Miebach, Wolfsgruber, Frommann, Fließbach, et al. 2018)

In conclusion, there is large room for improvement regarding the quantitative assessment of SCD and subtle cognitive decline which pose a major task for further research (Jessen et al. 2014). Automatic classification and machine learning might help detecting specific assessment strategies for preclinical AD and the refinement of neuropsychological test batteries.

We generated various neuropsychological and clinical parameters from patient conversations and examinations. To allow automatic classification using this data, we used multiple types of classifiers to make a diagnosis. In some cases we even managed to get an F-score of over 90%.

In any event, it is very time-consuming to generate the underlying medical data. Therefore, it is of utmost importance to generate only those data that is required to produce a good classification. To find out this data, we evaluated different approaches. On the one hand, we used a genetic search over the underlying feature space to find out which subset of features leads to better results. On the other hand, we calculated distance correlation to detect dependencies between pairs of features. We discovered that in some cases, less than 50% of the features of the underlying medical study suffice to generate the best performing classification.

8.3 Related Work

Machine learning techniques are becoming more and more popular in clinical research and are an established technique in MRT studies (Bede 2017). Recent studies start from optimizing neuropsychological assessment for cognitive, behavioral and functional impairment using machine learning (Battista, Salvatore, and Castiglioni 2017). However, studies using automatic classification to distinguish AD from non-AD patients did not focus on earlier preclinical or early MCI stages (Gurevich et al. 2017). Further, modern machine learning techniques have up to now only very rarely been used for the differential diagnosis of cognitive complaints based on the results of interview data. Mehler, Uslu, and Hemati (2016), for example, automatically analyzed physician-patient talks for differentiating patients suffering from epilepsies or dissociative disorders. This was done by means of the text2voronoi algorithm, which is also used in this paper. Regarding the assessment of SCD, Miebach, Wolfsgruber, Frommann, R. Buckley, et al. (2017) were able to confirm several qualitative complaint categories proposed by Rachel F. Buckley et al. (2015) which are specific for memory-clinic and depressive patients. This suggests that the subjective experience of cognitive decline can be captured by means of a set of interview questions and categories and therefore could be useful for clinicians to detect individuals at high-risk for AD. Investigations of MCI patients self-awareness and experience of their diagnosis have revealed that qualitative approaches may well lead to a more in-depth view than quantitative measurements (Lingler et al. 2006; Roberts and Clare 2013). However, a qualitative approach is more time consuming than a quantitative one making the diagnostic process more cost intensive. With the gaining interest in an improved detection rate of AD-pathology with less time and cost intensive screening tools, clinicians have the unique opportunity to take advantage of automated classification techniques. This exploratory example of machine learning combined neuropsychological data for the assessment of cognitive impairment and qualitative extracted interview-based features for cognitive complaints in memory-clinic-patients, depressive patients and in healthy controls.

8.4 Models

In the present study, we experiment with several classification models to be independent of the classifier and to assess the significance of features while being less dependent on these classifiers. As input, the classifiers are fed with neuropsychologically and clinically determined feature values. The neuropsychological part of our study includes a test battery for assessing cognitive performance and depressive symptoms. The clinically determined values are ratings based on qualitative interviews designed to capture aspects of subjective cognitive complaints in the context of preclinical dementia. In contrast to the neuropsychological data set these values are based on expert ratings instead of self-ratings or performance measures. The different group status (memory-clinic-patients, depressive patients, healthy controls) were set as output.

Since we only have a limited amount of data, we carried out a leave-one-out cross-validation for each classifier being tested. This makes sense since each patient is referred to individually for classification. With other data splitting methods, the risk of overfitting is too high.

8.4.1 SVM

As a baseline for the experiments we trained a *support vector machine* (SVM) and used it for classification. This is done by means of the SVM-light (Joachims 1998) implementation using the radial basis function (RBF) kernel. To find optimal parameters for training, we carried out a parameter study on the gamma and the cost parameter. For the cost parameter we examined values between 0.01 and 0.000,001; for the gamma parameter we considered values in the range of 1 and 1,000,000.

8.4.2 Neural Network

To carry out the same experiments using modern classification methods, neural network-based methods were indispensable. To this end, we used the framework Keras (Chollet et al. 2015). More specifically, we trained a feedforward

network to get a classifier of medical data. Here again, we conducted a parameter study to find the best performing parameters in each experiment. The following parameters were evaluated:

- optimizer: [adam, adamax, rmsprop]
- dropouts: [0.25, 0.5, 0.75]
- layersize: [50, 100, 200, 500]
- layersize2: [0, 50, 100, 200]

We achieved the best results with a dropout of 0.25, *adam* (Kingma and Ba 2014) as optimizer and two hidden layers.

8.4.3 Systematic Feature Evaluation for SVM

We examine the impact of feature selection on the F-Measure. While some features may consistently contribute to good classification results, others may reduce performance. That is, we expect that using all available features will most likely not yield the best F-Measure. Since a systematic evaluation of all $2^{138} - 1$ feature combinations is impossible, we apply several approaches to determine local optimal values and to examine the overall robustness of the feature set. If not stated otherwise each evaluation of a given feature set includes a parameter study regarding the optimal *gamma* and *cost* value for the SVM. Here again, our studies are based on SVM-light (Joachims 1998).

We start with performing a genetic search for optimal feature selection. Genetic algorithms have successfully been used for feature selection (L. Li et al. 2005). In our case, a population of n variants, which have been initialized randomly, are evaluated, ranked and flipped (bitwise) over t turns. In each turn, the best ranking variants are kept and mutated to generate additional variants while worst performing instances are dropped. In this way, a hill-climbing algorithm is implemented that approaches local maxima of better performing subsets of features.

Top-down and bottom-up search

In order to examine the overall robustness of the feature set we gradually remove features from the entire set (top-down).

In addition we explore the effect of gradually increasing the number of features starting from an empty set (bottom-up). At each step, the feature that maximizes the performance of the remaining set is added or removed, resulting in $\frac{n^2+n}{2}$ computations. Whenever multiple variants achieve the same top value we chose one of them randomly.

Applying this methodology to feature reduction is an important step, as it not only improves the classification results but also helps reducing the computation time in further analyses.

8.4.4 text2voronoi

Mehler, Uslu, and Hemati (2016) have developed a new classification method which visualizes input texts and then uses the visual representation of these texts to drive the classification. The advantage of this method is that one gets a visual depiction of the underlying text that can be used by analogy to MRI scans. Instead on working on the content words of a text, text2voronoi is mainly working on distributions of grammatical features of words in this text. In this way, it allows for completely abstracting from text content. This is indispensable when dealing with rather short talks of doctor and patients which, though describing the same disease, may select words of a completely unrestricted semantic universe. Using grammatical information, embeddings are produced by means of word2vec (Mikolov, K. Chen, et al. 2013). Then, a Voronoi tessellation is calculated on this data to map texts onto 2- or 3D spaces. Finally, the resulting depictions are used to drive the classification.

8.4.5 fastText

We additionally experimented with fastText (Joulin et al. 2016), an efficient text classifier, to compare it with text2voronoi. fastText is based on a feed-forward neural network with only one hidden layer. Joulin et al. (2016) show

that fastText compares with state-of-the art classifiers while being faster than its competitors.

8.5 Experiments

8.5.1 Sample description

The total sample of this study includes n=65 older subjects (mean age=70.03 years; 52.3% female). All participants were above the age of 55 and had sufficient ability to speak German. All procedures contributing to this work comply with the ethical standards of the relevant national and institutional committees on human experimentation and with the Helsinki Declaration of 1975, as revised in 2008. The study was approved by the local ethical committees of the University of Bonn, and informed written consent was obtained from all subjects.

Memory-clinic patients (MC)(n=23) were referred by their general practitioners to the Clinical Treatment and Reseach Center for Neurodegenerative Disorders (KBFZ), Department for Neurodegenerative Diseases and Geriatric Psychiatry, University Hospital Bonn for a diagnostic work up of cognitive functioning. Diagnosis of AD Dementia or MCI was made according to the core clinical criteria of the NIA-AA (Albert et al. 2011; McKhann et al. 1984). The diagnostic procedure included a cognitive assessment, detailed medical history, and a neurological examination. Of the total sample 15 fulfilled the core clinical criteria of mild cognitive impairment (MCI) according to the NIA-AA criteria (performance under 1.5 SD below age, gender, education adjusted norms) (Albert et al. 2011). The remaining 8 patients only had subjective concerns without objective impairment, and were classified as patients with Subjective Cognitive Decline (SCD).

Major depressive disorder (MDD) patients (n=21) were recruited from the Clinic of Psychiatry and Psychotherapy, University Hospital Bonn. All patients fulfilled a diagnosis of a unipolar, major depressive disorder according to ICD-10 criteria (Organization 1993).

The Healthy control group (HC) (n=21) was recruited from a scope of a nor-

mative study of the German Center for Neurodegenerative Diseases (DZNE) Bonn that evaluated neuropsychological performance of healthy older individuals. They were excluded from the participation when they (1) were concerned about mental abilities or memory (2) had been in psychological, psychiatric or neurological treatment within the last 6 months (3) had any severe or chronic disease (e.g. diabetes or MS) (4) had experienced head injury with a loss of consciousness, (5) had a neurological disease (e.g. AD or Parkinson) (6) or had a relative with a first-degree relative with a documented diagnosis of neurodegenerative disease in their family history.

Clinical Rating of cognitive complaints

The Clinical rating was made based on a semi-structured interview designed to capture all complaint categories proposed by Rachel F. Buckley et al. (2015). The Interview similar to a clinical routine interview, started with an open question asking whether the patient had noticed "*any changes in memory or thinking during the last years"* followed by detailed questions about the complaint itself. The interview procedure followed a semi-structured format and lasted between 8 and 31 min. Each interview had an unstructured beginning, which allowed patients to determine the initial focus of the conversation. If cognitive changes were reported, the participants were asked to give an example of their everyday life. Then the patient was asked whether he/she has noticed further cognitive problems followed by the request to give an everyday example. This process was repeated until the participant did not mention further complaints. He/she was then asked to name the most concerning symptom which was selected for further detailed questioning. If the participant reported another concerning symptom, we repeated the detailed questions about the complaint itself. Therefore, 58% of the sample named two concerning symptoms. All Interviews were digitally recorded and later transcribed verbatim by the interviewer. Data for analyses presented in this manuscript were derived from the ratings of a single clinical psychologist (LM) who also conducted all the interviews. To capture all aspects of cognitive complaints, the clinical rating in this study was based on glossary of cognitive complaints based on a combination of the cognitive complaint categories proposed by Rachel F. Buckley et al. (2015) and the complaint themes proposed by Miebach, Wolfsgruber,

Frommann, Fließbach, et al. (2018). The glossary contains the following categories: *Increasing frequency, Sense of predomination and growing concern, Situational lapses, Relative absence of spatio-temporal contextualisation, burdensome coping strategies, Dismissive attitude, attentional fluctuation/vagueness, Impact on affect, Progression, an over-endorsed complaint, dependency, affective influence on memory, distractible speech, general complaints about increasing memory problems, difficulties in Action monitoring, difficulties in initiating actions, deceleration, slowing of cognitive processing speed, nonspecific overwork, forgetfulness, short-term memory problems, content memory problems, blank mind, loss-of-control experience, derealisation, formal though disorder, prospective memory, planning, learning, cognitive flexibility, increased distractibility, concentration difficulties, word finding difficulties, memory for names, dyscalculia, visual-spatial-disorientation, general decline, no changes in cognitive functions.* The categories were extracted from the interview material using inductive qualitative approaches. The complaint categories based on Rachel F. Buckley et al. (2015) were related to the grounded theory (Strauss and Corbin 1997) whereas the complaint themes extracted by Miebach, Wolfsgruber, Frommann, Fließbach, et al. (2018) were based on the interpretative phenomenological analysis (IPA) (J. Smith, Flowers, and Larkin 2009). Therefore the presented deductive rating of cognitive concerns is based on two different phenomenological approaches which allows to capture highly nuanced and contextualized aspects of subjective experiences (J. Smith, Flowers, and Larkin 2009). The Interview procedure and categorization system are described in detail in Miebach, Wolfsgruber, Frommann, R. Buckley, et al. (2017) and Miebach, Wolfsgruber, Frommann, Fließbach, et al. (2018). For the coding process, we used a deductive category assignment approach similar to qualitative content analysis (Mayring 2014). Participant's responses were coded using a binary coding system (i.e. 0=theme absent; 1=theme present).

Neuropsychological assessment

The Neuropsychological assessment included a set of different clinical measurements for global memory and cognitive performance specifically designed for early diagnosis of AD dementia. The test battery included the Free and Cued Selective Reminding Test (FCSRT) (Ivnik et al. 1997) and the German version

of the neuropsychological test battery of the Consortium to Establish a Registry for Alzheimer's disease (CERAD-plus (Morris et al. 1989)) with various sub-tests (e.g. verbal fluency, Boston Naming Test, Mini Mental State Exam, Word List learning, Constructional praxis, Word List recall, word list recognition, constructional praxis recall, TMT-A, TMT-B, the symbol digit modalities test (SDMT) (A. Smith 1982)). Depressive symptoms were assessed with the 15-item version of the Geriatric Depression Scale (GDS (Yesavage et al. 1983)) and the Patient Health Questionnaire (PHQ-9) (Kroenke et al. 2010)

Group characteristics and demographical differences

Analysis for the group differences were performed using IBM SPSS Version 22 (Corp 2013). Group differences were observed for age, education, interview duration, GDS and PHQ-9 scores. Memory-clinic patients were slightly older (M=72.91 yr) compared to MDD (M=69.43 yr) and the interview duration was significantly longer (M=18.41 min) in comparison with HC (M=14.32 min) and the MDD-Group (M=12.07 min). HC were younger, performed significantly better on the MMSE (M=29) and exhibited lower levels of depressive symptomatology (GDS; M=0.62) compared to MDD and the Memory-clinic patients. The depressive group exhibited elevated levels of depressive symptomatology, significantly above the GDS cut-off for depression (M=7.00) and the PHQ-9 cut-off for moderate depression (M=10.89). Depressive patients also had significantly fewer years of education (M=12.57) compared to HC (M=15.10) and Memory-clinic patients (M=15.50).

8.5.2 Classification

We have used the models from section 8.4 to classify the patients. In doing so, we classify on the textual data and on the clinical and neuropsychologically generated data. Models 8.4.1 to 8.4.3 are designed for the classification with the self-generated data, while the models 8.4.4 and 8.4.5 are designed for textual classification.

Clinical and neuropsychological feature classification

In this experiment, we used the clinical ratings of subjective cognitive complaints based on the qualitative interview as one feature set. We also used the neuropsychological test results including data about objective cognitive performance and measurements of depression as another feature set. First of all, we used both feature sets independently for classification. We then combined both sets of features and used the combination for classification. Table 8.1 shows the results of the different classifiers applied to the 3 feature sets. We discovered that the combined features are always more successful than both feature sets on their own.

Model	Neuropsych.	Clinical	Combined
8.4.1	0.747	0.706	0.794
8.4.2	0.754	0.723	0.800
8.4.3a	0.870	0.821	0.881
8.4.3b	0.933	0.928	0.949

Table 8.1: F-scores of the classifiers with regard to the different data sets.

Patient talks classification

In this experiment, we analyze the texts of the cognitive complaint interviews and use them for classification (leave one out cross validation). We only used the text content of the patients and removed the doctor's text data from the interview protocols because the doctor asks all patients similar questions. Table 8.2 shows the results of the 2 methods we used for classification. It can be seen that the baseline classifier fastText can hardly classify the texts. However, if the texts are abstracted, as it is the case with text2voronoi, an improvement is achieved.

Model	F-score
8.4.4	0.520
8.4.5	0.340

Table 8.2: Results of the textual classification experiment of the 3 patient groups (MC/MDD/HC)

8.5.3 Feature analysis

Now that we have applied different classifiers in different experiments, we want to find out which of the used features were actually needed. Here, we have used the following approaches.

Genetic feature search

As explained in Section 8.4.3, we have also carried out a genetic search (GeneticSVM) of the parameters to find the smallest possible subset, which provides best results. We found out that only a fraction of the features are required to perform a good classification.

Experiment	**Subset**
Neuropsych.	47.30%
Clinical	40.63%
Combined	40.58%

Table 8.3: Subset analysis of the features using model 8.4.3.

In addition to the genetic search, we have also carried out two other approaches, as described in Section 8.4.3. Figure 8.1 shows the process of this analysis. Again, it is obvious that few features are enough to get the best results. We achieve the best score with only 64 features.

Decision tree

A good way to analyze the features is to use decision trees, as it follows simple and comprehensible heuristics. The graphic representation as a tree diagram also illustrates hierarchically consecutive decisions. We have used the Python package *sklearn* (Pedregosa et al. 2011) to perform these analyses. In our best performing experiment, we have the following patient distribution:

- $[21, 23, 21]$ - (Control patients, Memory-Clinic patients, Depressive patients)

Figure 8.2 shows that feature 41 (SDMT - neuropsychological score developed to identify individuals with neurological impairment) is at the top of the tree.

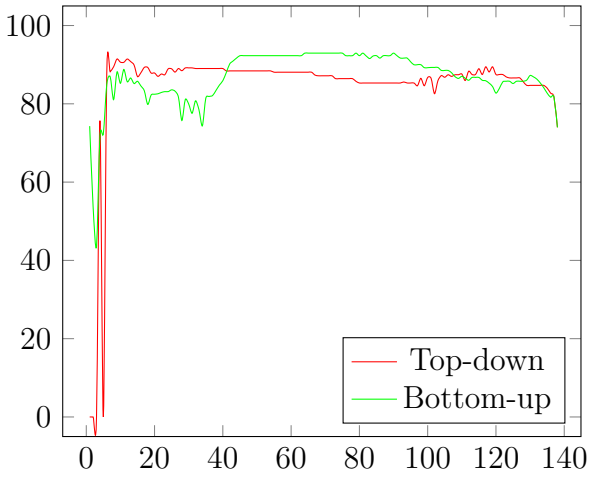


Figure 8.1: F-scores based on the number of features in the example of the combined experiment (see Section 8.5.2).

If the value of SDMT is less than or equal to 44.5, the patient distribution is divided into:

- $[0, 20, 18]$ - Only memory-clinic or depressive patients.
- $[21, 3, 3]$ - Mostly healthy controls.

Thus, we divide all control patients into a separate group.

The next important parameter is feature 45 (GDS - neuropsychological measurement for depression) (Yesavage et al. 1983). This divides the group of diseases ($[0, 20, 18]$) into the following patient distribution:

- $[0, 17, 4]$ - Mostly Memory-clinic patients
- $[0, 3, 14]$ - Mostly depressive patients

This means that we could group 52 (14+17+21) patients correctly with these 2 features alone, but 13 (3+4+3+3) wrong. You can also see that these features belong to the neuropsychological features. This also makes sense, as these values also lead to better classifications (see example 8.5.2). The further down the tree is examined at, the more precise the distributions will be. However, given the number of features (138) and the small amount of patients (65), this is rather overfitting.

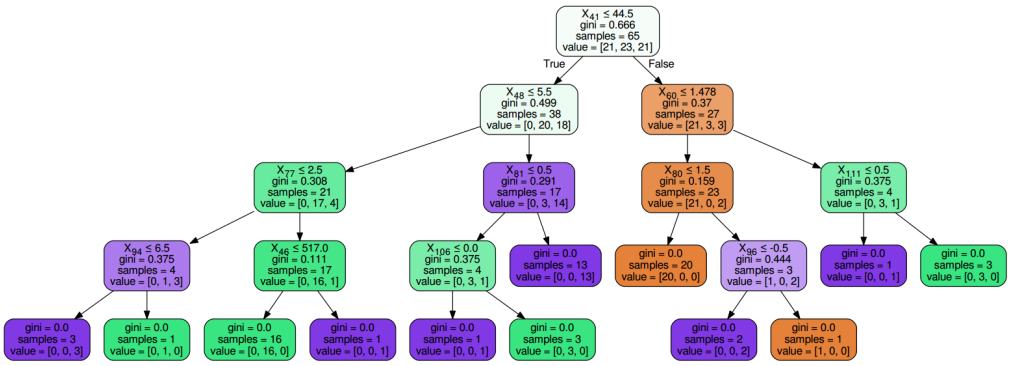


Figure 8.2: Result of the decision tree based on the combined example of experiment 8.5.2.

Distance correlation

To measure the interdependence between the features as described in Section 8.5.1 we calculated distance correlation between pairs of features. For this we used the R package *energy* (Rizzo and Szekely 2017) utilized by TextImager (Uslu, Hemati, et al. 2017). Interdependent features are an indicator for redundant data. These redundant features are less helpful for classification. Figure 8.3 shows the heatmap of the pairwise dependencies. Each cell represents the distance correlation of the features X and Y, with the green shading indicating the dependency (darker = more dependent). The diagonal is green which indicates that every feature is correlated to itself. The green squares also provide important information. The first dependency square at the top left in Figure 8.3, for example, contains only neuropsychological features based on the MMSE. The mini-mental state test (MMSE) is a brief screening tool for Alzheimer dementia and impairment in global cognition (M. F. Folstein, S. E. Folstein, and McHugh 1975). The MMSE test includes items assessing orientation, word recall and registration, attention and calculation, and language and visuospatial abilities. As a logical consequence we observed a high dependency between the different subscores of the MMSE.

As mentioned above, 58% of the sample named more than one cognitive complaint. As a result, the categories were coded for a second time. A high dependency between these features is therefore a consequence of the interview procedure. These dependencies can be seen in Figure 8.3 by the large green

square at the bottom right.

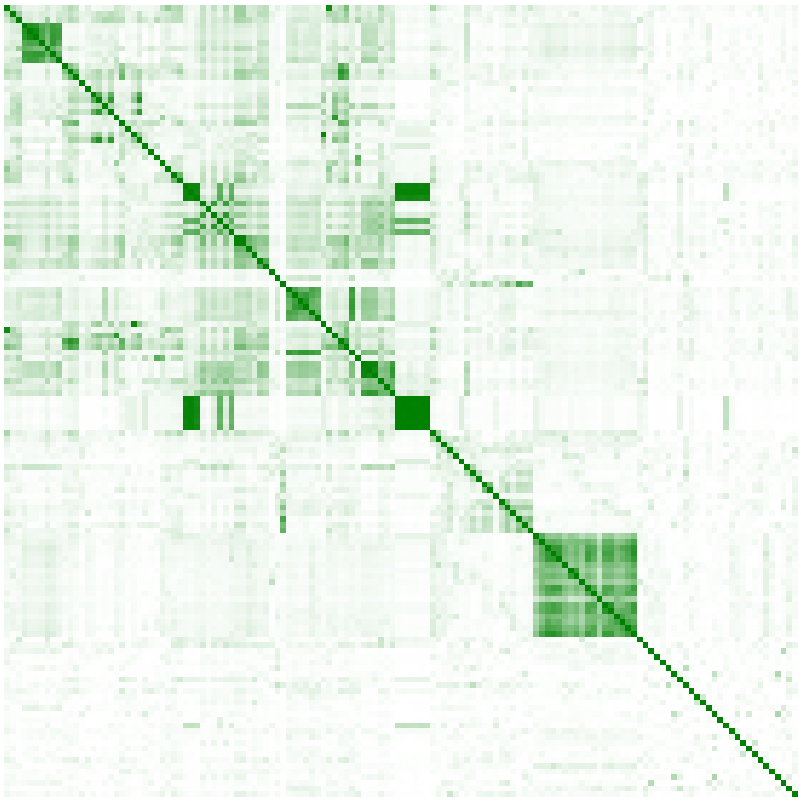


Figure 8.3: Visual depiction of pairwise dependence of the features.

8.6 Discussion

The present study is the first to combine a qualitative text-analytical approach for cognitive complaints with an automatic classification system for three different diagnostic groups. Recently, only a few studies have explored the use of automated learning methods within the neuropsychological literature. In this proof of principle study we used a machine learning approach based on neuropsychological and interview generated cognitive complaint categories for the classification of memory-clinic patients, depressive patients and normal healthy older adults.

We aimed to replicate the diagnostic value of the recently proposed complaint categories using an automatic classification method instead of current statistical methods used in clinical research (Rachel F. Buckley et al. 2015;

Miebach, Wolfsgruber, Frommann, R. Buckley, et al. 2017; Miebach, Wolfsgruber, Frommann, Fließbach, et al. 2018). Cognitive complaints were elicited with a semistructured interview comparable with a typical clinical routine interview.

The current study results revealed that machine learning techniques can accurately classify patients measured via neuropsychological test battery and via clinical rating of cognitive complaints. We found that the classification with self-generated characteristics extracted by a qualitative approach works much better than with the recorded texts in the patient conversations.

This result makes sense because patients talk about many different topics in the diagnostic interview and the content of these texts is not reliable for determining a disease. Therefore the interpretation of cognitive complaints relies on expertise of some kind which is not ideal for a wide distribution across studies.

In the case of the second experiment, the neuropsychological data outperformed the clinical ratings based on interview data. This could be explained by the heterogeneous sample including patients with mild cognitive impairment as well as patients with only subjective cognitive decline and depression. In line with current literature, the combination of neuropsychology and the clinical rating reached the best diagnostic accuracy (Molinuevo et al. 2016). A replication in a larger sample with focus on the complaint categories is needed to extract features which are truly relevant for AD-pathology.

Given the present results, we believe it is much more likely that a measure incorporating both, qualitative text based and quantitative neuropsychological methods will be able to identify the preclinical AD profile. Recent studies used composite scores calculated based on z-transformed subscales of different SCD assessments to predict the tau-pathology in the enthorinal cortex of healthy older adults (Rachel F Buckley et al. 2017).

However, in the case of textual classification, an improvement is achieved when the text is transformed into a more abstract model (text2voronoi). The experiments also show that the neural network-based approaches are usually somewhat better than the SVMs. However, the best solutions can be found with GeneticSVM and even only a subset of all features. As a result, we found

out that a few features are enough to get a good classification. However, these features (SDMT; GDS) are established clinical screening tools for the measurement of memory impairment and depressive symptomatology (Yesavage et al. 1983; A. Smith 1982). A feature analysis only including the cognitive complaint categories should be an important next step with a higher clinical impact in the field of AD research. We analyzed them and found out that there are some dependencies among the features. There is a need of alternative ways for the operationalization and the diagnosis of AD-relevant cognitive complaints. Using a semistructured interview based on qualitative categories seems to be promising regarding the clinical evaluation of memory complaints in non-demented elderly. Further improvement of the complaint glossary and the rating scale is needed for the detection of preclinical AD. Therefore machine learning approaches could be promising for reducing and refining neurospychological assessments. This information can save a lot of work, since the dependent features barely improve the classification.

8.7 Conclusion

In this paper we have used different classifiers in various patient diagnosis experiments. We have shown that a good classification can be achieved by using cognitive complaint categories based on clinical interview and neuropsychological data from standardized test batteries. We found that the combination of these data sets leads to the best results with an F-score of 80%. In addition, we have applied a number of different approaches to find the optimal subset of features that provide the best classification. In this case we even achieve an F-score of 94.874,363,520,300,38%. However, classification at text level is not yet particularly successful. In future work we aim at studying different abstractions of texts (as provided, for example, by text2voronoi) in order to detect expressive linguistic features that allow for automatically assessing the diseases under consideration.

9 fastSense: An Efficient Word Sense Disambiguation Classifier

The content of this chapter (Uslu, Mehler, Baumartz, et al. 2018) was published in Proceedings of the 11th edition of the Language Resources and Evaluation Conference.

9.1 Abstract

The task of *word sense disambiguation* (WSD) is to determine the meaning of an ambiguous word in a given context. Despite its importance for most NLP tasks, WSD can still be considered unsolved. The reason is that we currently lack tools for WSD that handle big data – "big" in terms of the number of ambiguous words and in terms of the overall number of senses to be distinguished. This desideratum is exactly the objective of fastSense, an efficient neural network-based tool for word sense disambiguation introduced in this paper. We train and test fastSense by means of the disambiguation pages of the German Wikipedia. In addition, we evaluate fastSense in the context of Senseval and SemEval. By reference to Senseval and SemEval we additionally perform a parameter study. We show that fastSense can process huge amounts of data quickly and also surpasses state-of-the-art tools in terms of F-measure.

9.2 Introduction

One of the core tasks in natural language processing is *word sense disambiguation*. Without disambiguation, we just consider a word as a combination of

characters and not the meaning behind it. Without properly disambiguating the lexical constituents of a text, it is almost impossible to process its content automatically. Our goal is to solve this problem and to make it applicable to large amounts of data. To this end, we present a neural network-based classifier for WSD called fastSense. We take sequences of words as input and compute a sense label per ambiguous word in that sequence as output. This approach was motivated by the classifier called fastText (Joulin et al. 2016). As the name suggests, fastText is designed to perform text classifications as quickly as possible. However, fastText is not suitable for disambiguating words. In addition, the neural network used by fastText does not support training of multi-labels. Therefore, we implemented our own word embedding-based neural network by analogy to the architecture of fastText. This allows us to apply fastSense to WSD efficiently even on big data. In order to test the time complexity of our approach, we created a disambiguation corpus from the German Wikipedia with over 50,000,000 training and test sets. We use the disambiguation pages and the link structure of Wikipedia to match words with their corresponding Wikipedia senses. In this paper, we deal with the German Wikipedia. In terms of size or space complexity, its sense model is far beyond what is normally studied, for example, in the framework of Senseval or SemEval. However, in order to show that our approach is language independent, we additionally perform multiple tests related to Senseval and SemEval. These tests show that our model can keep up with state-of-the-art tools.

The paper is structured as follows: In Section 9.3, we contrast fastSense with related approaches to WSD. In Section 9.4 we introduce the architecture of fastSense. In Section 9.5, we explain the experiments carried out to evaluate fastSense and show the achieved results. In Section 9.5.3, we discuss our findings and in Section 9.6, we give a summary of the paper.

9.3 Related Work

Our approach is motivated by fastText (Joulin et al. 2016). This relates to the very efficient and successful way by which fastText allows for classifying data. The main purpose of fastText is text classification. Its architecture is similar to `word2vec` (Mikolov, K. Chen, et al. 2013): both approaches are based on

a bag-of-words model. Further, both of them use a single hidden layer. The difference between `word2vec` and fastText is that the latter requires to define a label for any input text, while `word2vec` uses context windows of lexical units to predict single words or vice versa. We transpose fastText to word sense disambiguation in order to efficiently determine the meaning of ambiguous words even in cases in which we face big data. By this we mean scenarios in which hundreds of thousands of different words are ambiguous.

fastSense is characterized by its simplicity, speed and quality. This distinguishes it from similar tools. For instance, Mihalcea and Csomai (2007), Ferragina and Scaiella (2010), Ratinov et al. (2011), Ratinov et al. (2011), Agerri, Bermudez, and Rigau (2014), and Moro, Raganato, and Navigli (2014) present approaches to *Entity Linking*. More specifically, they link tokens in texts to knowledge databases such as DBpedia, Wikipedia or WordNet to identify instances of entities. These approaches are similar to ours, with the difference that we focus on ambiguous words, while the latter approaches also link words that have only one meaning. The disadvantage of these approaches is their speed. For large amounts of data, they may take weeks to produce an output (see Table 9.2 for an estimation of this time effort). Mihalcea (2007) uses a technique similar to the one presented here to build a sense-tagged Wikipedia corpus using the link structure of Wikipedia to match senses. However, this corpus has not been used to disambiguate ambiguous words according to Wikipedia's disambiguation pages, but to compare them with the data of Senseval 2. Mihalcea, Tarau, and Figa (2004) use a PageRank algorithm operating on semantic networks to perform WSD. The underlying network is spanned by means of semantic relations of synsets, entailment and other WordNet relations. The PageRank algorithm assigns scores to words and chooses the disambiguating synset of highest score. Yuan et al. (2016) present two WSD algorithms, achieving the best results by means of a semi-supervised algorithm combining labeled sentences with unlabeled ones and propagating labels based on sentence similarity. Tripodi and Pelillo (2016) describe an approach to WSD based on evolutionary game theory, in which words tend to adapt senses of their neighborhood so that WSD is reformulated as a kind of constraint satisfaction. Zhong and H. T. Ng (2010) present a framework for English all-words WSD. It disambiguates each content word of a given sentence using a linear kernel-based SVM (Joachims 2002). Iacobacci, Pilehvar,

and Navigli (2016) show that the use of word embeddings achieves an improvement in WSD compared to standard features. Chaplot, Bhattacharyya, and Paranjape (2015) propose a graph based unsupervised WSD system which requires WordNet, a dependency parser and a POS-Tagger. They model WSD as a maximum-a-posteriori inference query operating on a Markov random field. Raganato, Bovi, and Navigli (2017) define WSD in terms of a sequence learning problem. This is done by means of a bidirectional LSTM-based neural network (Hochreiter and Schmidhuber 1997). Melamud, Goldberger, and Dagan (2016) present `context2vec` which is also based on bidirectional LSTMs for learning disambiguating word contexts.

Unlike these approaches, we present a method that can handle big data: in terms of the number of senses to be distinguished *and* in terms of the number of units to be disambiguated. On the one hand, knowledge driven approaches using, for example, WordNet and related resources are limited in terms of the number of senses distinguished by them. GermaNet, for example, distinguishes 33,630 senses of 13,445 ambiguous words – that is much less than considered by us. On the other hand, approaches that rely on algorithms like PageRank or classifiers like SVMs or LSTMs are limited in terms of their time efficiency: it is a computational challenge to maintain, for example, SVMs for each of the 825,179 senses of the 221,965 ambigous words of the German Wikipedia which, however, are easily covered by our approach. Thus, we are in need of a flexible, easy-to-compute, but efficient method for WSD as presented in the next section.

9.4 Model architecture

During training, fastSense requires text as input and the corresponding senses as output (see Figure 9.1). Its single hidden layer is an embedding layer in which word indexes from the input layer are converted into word vectors. More specifically, the number of hidden nodes corresponds to the dimension of the pre-trained word embedding vectors so that the weights of edges between input and hidden nodes correspond to the respective coordinates of the latter vectors. We computed the word embeddings by means of `word2vec` (Mikolov, K. Chen, et al. 2013) using Wikipedia as the underlying corpus. The embed-

dings are then merged in the hidden layer according to the *global averaging pooling principle* (Lin, Q. Chen, and Yan 2013).

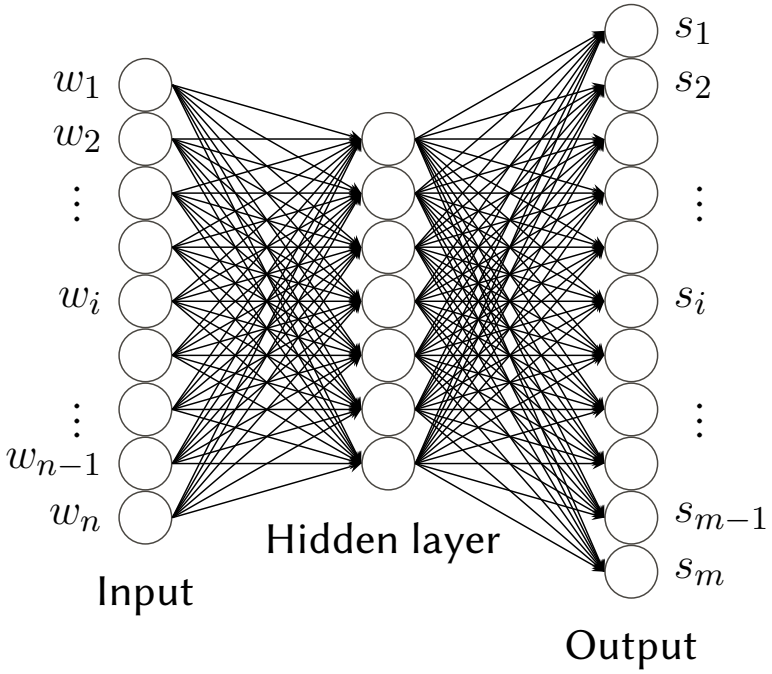


Figure 9.1: Model architecture of fastSense.

To support multi-label training we used the *sigmoid* function as an activation function of the output layer. For the sake of optimizing, `Adamax`, a special variant of `Adam` (Kingma and Ba 2014), is a very efficient choice in practice. It uses the *infinity norm*, which makes it possible to stabilize the training over longer periods of time and, thus, to achieve faster and better results. To prevent overfitting, we used *Dropout* (Srivastava et al. 2014) as regularization method. Dropout removes nodes during each training session, ignores them and does not train with them. After the training process, the nodes are reinserted with their original weights.

To apply this model to WSD, we additionally developed a method for post-processing the output of the neural network. Usually, the sense of highest probability is selected as output. However, since the output layer contains all senses of all ambiguous words, it is unlikely that the target sense of a word x to be disambiguated equals the top ranked sense. Thus, we do not necessarily select the label of highest probability, but go through the list of rank-ordered candidates until the first occurrence of x tagged by a correspond-

ing sense number is reached. This sense unit is then produced as the output of disambiguating x. As an example, consider processing the ambiguous word *bank* as depicted in Figure 9.2: when observing an occurrence of this word in a sentence about a *financial* topic, a classifier like fastText will likely suggest labels, such as *finance*, *money* or *financial institute* because of the fact that the input sentence is about such a topic. However, what we are looking for is the *sense of the word* and not the most strongly associated topic label. Thus, we descend the sorted output of fastSense given the input sentence until we reach a candidate sense prefixed by *bank* (i.e., *bank_2*) that is taken to predict the sense of the word in this sentence. In this way, fastSense can be used as a tool for WSD. Conversely speaking, we reconstructed WSD as a kind of topic labeling that is performed by fastSense by analogy to fastText.

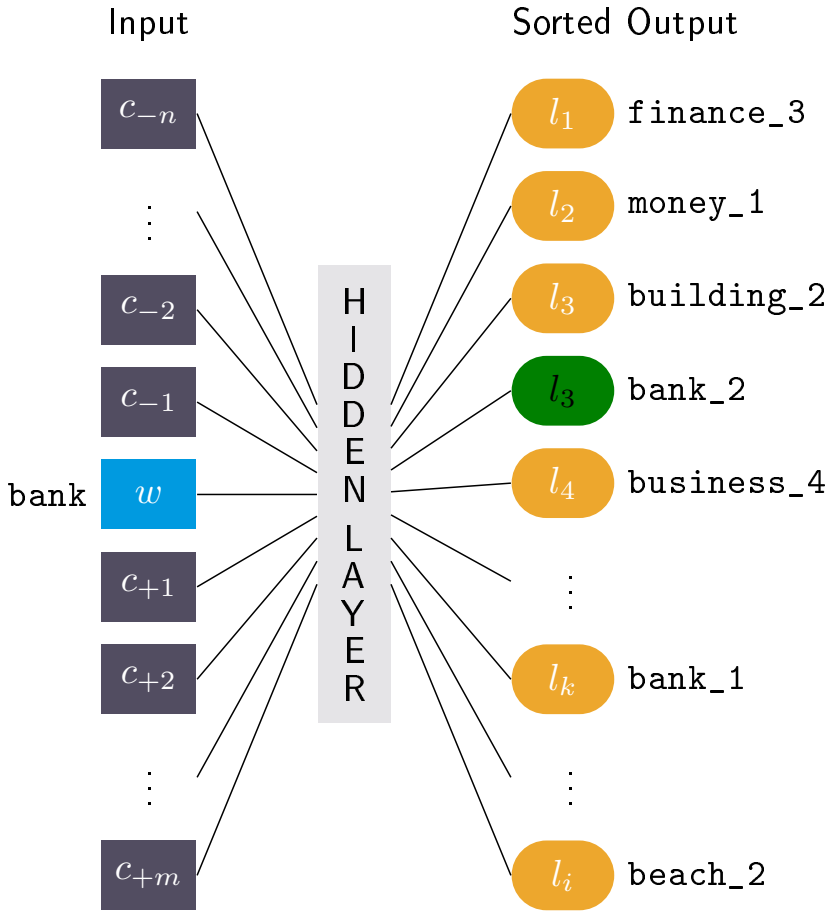


Figure 9.2: fastSense in work: *bank_2* is selected as the first occurrence prefixed by *bank*.

9.5 Experiment

We perform two disambiguation experiments. The first one uses the German Wikipedia to demonstrate the efficiency of fastSense. The second one is based on Senseval and SemEval. It aims at comparing fastSense with state-of-the-art tools. Table 9.1 lists the parameters used for these evaluations.

Short	**Description**
POS	Part of speech is considered.
Lemma	Lemma information is considered.
Token	Token information is considered.
WP	POS information is added.
x-Nb	x neighbors (left and right) are considered as context.
MinContext(k)	Any input text must contain at least k tokens to be used in training or testing.

Table 9.1: Parameters used for evaluating fastSense.

9.5.1 Wikipedia-based Disambiguation

In order to show that our approach allows for capturing large amounts of data, we created a corpus using the disambiguation pages of the whole German Wikipedia. For pre-processing this data we used TextImager (Hemati, Uslu, and Mehler 2016). Every word listed on a disambiguation page in Wikipedia corresponds to a different meaning of the corresponding lemma (page title). In total, we processed 221,965 disambiguation pages related to 825,179 senses. On average, this gives 3.72 senses per word. The disambiguation page *Bank*[1], for example, distinguishes 42 senses. For classification, we take each paragraph in Wikipedia that contains one of the target words to be disambiguated as a sample for training semantic representations of these words. This results in a corpus of 55,796,534 instances (49.9 GB) . Using this corpus, we trained fastSense on 51,067,054 instances (46GB) and tested it on the remaining 4,729,480 instances (3.9GB). The test takes only 20 minutes on a single thread. This runtime can be further reduced linearly by distributing fastSense over differ-

[1] https://de.wikipedia.org/wiki/Bank_(Begriffskl%C3%A4rung)

ent threads. The results of the Wikipedia-based evaluation are shown in Table 9.3.

We compared several entity linking tools with fastSense in terms of time expenditure. With the same hardware, these tools take 6 to 188 days to process our test set. The effort was estimated based on a subset of elements as documented in Table 9.2. Obviously, fastSense outperforms these competitors. However, since these tools link to different resources (e.g., DBpedia, WordNet or Wikipedia), this comparison only holds for time effort.

Tools	1	500	1,000	5,000	4,729,480
Wikifier	0:01	8:20	16:41	1:24:06	≈55 days
Illinois	3:28	5:05	6:53	24:40	≈14 days
IXA	0:03	28:20	58:49	4:47:32	≈188 days
Babelfy	0:01	0:55	1:50	-	≈6 days
TAGME	1:19	3:20	5:42	28:40	≈18 days
fastSense	0:07	0:07	0:10	0:13	20:46

Table 9.2: Runtime-related evaluation regarding similar tools using 1, 500, 1,000 and all test instances.

Type	Min-Context	F1-score
fastSense	1	0.735
fastSense	2	0.778
fastSense	5	0.810
fastText	1	0.071
MFS Baseline	1	0.627

Table 9.3: Wikipedia-based evaluation of fastSense in comparison to fastText and the most frequent sense (MFS) baseline. We used a learning rate of 0.050, a hidden layer size of 10 and 5 training epochs.

9.5.2 Senseval and SemEval related Disambiguation

SemCor (Mihalcea 2016) provides texts with semantically annotated WordNet senses, which are automatically mapped to WordNet. We trained on SemCor 3.0 for performing Senseval and SemEval related tests. Because of the small amount of data provided by this corpus (234,136 disambiguated words), we

were able to perform a parameter study to search for the best performing parameter settings. Candidates for feature selection are POS, token, lemma and combinations thereof (see Table 9.1). Next, we tested different word context sizes (Context), word-n-grams (NGrams), learning rates (LR), dimensions of the hidden layer (Dim) and the number of epochs (Epoch). After each optimization step, we used the best performer of the preliminary round (see Table 9.4).

1. Type	SE2
Token WP	**0.703**
Lemma WP	0.697
POS	0.662
Token	0.701
Lemma	0.699

2. Context	SE2
S	0.703
1-Nb	**0.732**
2-Nb	0.718
3-Nb	0.720
4-Nb	0.706

3. NGrams	SE2
1	0.723
2	0.730
3	**0.732**

4. LR	SE2
0.025	0.718
0.05	0.724
0.1	**0.732**
0.5	0.724

5. Dim	SE2
5	0.710
10	**0.732**
25	0.711
50	0.712

6. Epoch	SE2
5	0.727
10	0.732
15	**0.735**
25	0.724
50	0.719

Table 9.4: Parameter study based on Senseval 2 and SemCor 3.0.

After completion we applied the optimal settings on Senseval 2 (English all-words) (SE2) and Senseval 3 (English all-words) (SE3). We also tested fastSense on SemEval-2007 Task 17 Subtask 1 (SE7) and Subtask 3 (SE7'), SemEval-2013 Task 12 (SE13) and SemEval-2015 Task 13 (SE15).

Since no information about lemmas or POS was given in the SE7 test sets, we carried out these experiments only on token basis. The results are listed in Table 9.5 and are compared to those of state-of-the-art tools in Table 9.6.

Input	**SE2**	**SE3**	**SE7**	**SE7'**	**SE13**	**SE15**
Token WP	0.735	0.735	–	–	0.662	0.732
Token	–	–	0.876	0.624	–	–

Table 9.5: F1-scores of the Senseval/SemEval-related tasks.

Model	SE2	SE3	SE7	SE7'	SE13	SE15
Iacobacci, 2016	0.634	0.653	**0.894**	0.578	**0.673**[2]	0.715[2]
Tripodi, 2016	0.660	0.647	0.828	0.565	–	–
Yuan, 2016	**0.736**	0.692	0.828	0.642	0.670	–
Chaplot, 2015	0.605	0.586	–	0.506	–	–
Zhong, 2010	0.625[1]	0.650[1]	0.879[1]	0.565[1]	0.653[2]	0.695[2]
Raganato, 2017	0.720	0.702	–	**0.648**	0.669	0.724
Melamud, 2016	0.718[2]	0.691[2]	–	0.613[2]	0.656[2]	0.719[2]
fastSense	0.735	**0.735**	0.876	0.624	0.662	**0.732**

[1] Iacobacci, Pilehvar, and Navigli (2016)
[2] Raganato, Camacho-Collados, and Navigli (2017)

Table 9.6: Comparison of state-of-the-art WSD tools on the Senseval 2, 3 and SemEval 7, 13 and 15 tasks using SemCor for training.

9.5.3 Discussion

We successfully used Wikipedia as a disambiguation corpus and show that fastSense can handle such a large amount of data (see Table 9.3). fastSense not only stands out for its speed, but also for the quality of its classification. We outperform the baseline considerably and show that similar approaches cannot keep up with fastSense in terms of runtime. Thus fastSense is a step towards performing WSD in relation to large amounts of data.

Since the SemCor data is many times smaller than the data derived from Wikipedia, we were able to carry out a parameter study in the Senseval- and SemEval-related experiments. Most interesting is our finding concerning the size of the context window. Using one neighbor of a word as context (1-Nb) and word-3-grams perform best. We also found that token in combination with POS-related information (see the parameter list in Table 9.1 and Table 9.4) perform best.

Note that fastSense is comparably fast: it takes only 20 minutes for disam-

biguating 4,729,480 instances on a single thread – that is, ca. 3,941 senses per second. Further, as mentioned in Section 9.4, fastText is less suited for WSD; accordingly, it performs worse compared to fastSense (see Table 9.3). In this sense, though being based on a related architecture, fastSense better fits the needs of WSD.

9.6 Conclusion

We presented a novel approach to word sense disambiguation called fastSense. We tested this model in the framework of Senseval and SemEval tasks as well as in terms of a big data-experiment based on the German Wikipedia. We achieve an F-score of up to 81.00% using the Wikipedia-based data. Further, we achieved 73.47% and 73.48% on Senseval 2 and 3, 87.57% and 62.40% on SE7 and SE7' and also 66.20% and 73.20% on SE13 and SE15. We show that fastSense cannot only work with huge data sets, but also surpasses state-of-the-art tools. Future work will address a parameter study on the Wikipedia-based data sets derived from different language releases. We will also account for dependency structures of sentences to gain an additional source of information for WSD. Our tool and all Wikipedia-based training and test data used in this paper will be made available open source.

10 Towards a DDC-based Topic Network Model of Wikipedia

The content of this chapter (Uslu, Mehler, Niekler, et al. 2018) was published in Proceedings of 2nd International Workshop on Modeling, Analysis, and Management of Social Networks and their Applications (SOCNET 2018).

10.1 Abstract

This paper presents a network-theoretical approach to modeling the semantics of large text networks. By example of the German Wikipedia we demonstrate how to estimate the structuring of topics focused by large corpora of natural language texts. Algorithms of this sort are needed to implement distributional semantics of textual manifestations in large online social networks. Our algorithm is based on a comparative study of short text classification starting from two state-of-the-art approaches: *Latent Dirichlet Allocation* (LDA) and *neural network language models* (NNLM). We evaluate these models by example of (i) OAI metadata, (ii) a TREC dataset and (iii) the Google Snippets dataset to demonstrate their performance. We additionally show that a combination of both classifiers is better than any of its constitutive models. Finally, we exemplify our text classifier by plotting the topic structuring of all articles of the German Wikipedia.

10.2 Introduction

In this paper, we develop a simple algorithm for modeling the semantics of large text networks. This is done by example of the German Wikipedia. Our

aim is to model the structure and networking of topics as manifested by large corpora of natural language texts. Algorithms serving this task are needed to implement a distributional semantics of textual manifestations in online social networks. One may want to know, for example, what topics are focused in a certain period of time in Twitter. Alternatively, one may want to know which fields of knowledge are either preferred or underrepresented in media such as Wikipedia or Wiktionary (Mehler, Rüdiger Gleim, Hemati, et al. 2017). In order to answer questions of this sort, it is necessary to determine the topic distribution of each individual text aggregate of the focused media and to decide how the resulting distributions are to be networked. This is the task of the present paper.

Our algorithm for modeling the thematic structure of large text corpora utilizes a well-established topic classification, that is, the *Dewey Decimal Classification* (DDC). More specifically, we build on a comparative study of approaches to short text classification. Short texts (e.g. tweets) refer to situations in which only snippets (e.g., metadata, abstracts, summaries or only single sentences such as titles) are available as input for classification instead of full texts. One example of this is digital libraries working on OAI (Open Archives Initiative) metadata (Waltinger et al. 2011). It also concerns text mining in online social media by example of chat messages, news feeds, tweets (Sriram et al. 2010), or turn-taking in online discussions (Forsythand and Martell 2007). In all these cases the central information to be extracted is what the snippets are about in order to classify them thematically (P. Wang et al. 2016), to disambiguate, to classify their constituents (Gaizauskas et al. 2014) or to enrich them by means of external knowledge resources. The requirement to handle big data streams is another reason to process snippets instead of full texts even if being accessible. In each of these cases, classifiers are influenced more by the sparseness of the lexical content of short text. Therefore, one needs both fast and accurate classifiers that are expressive enough to overcome the problem of lexical sparseness.

In this paper, we present a network model of topic structuring that is based on a comparative study of text snippet classification starting from two state-of-the-art approaches: *Latent Dirichlet Allocation* (LDA) and *neural network language models* (NNLM). In the latter case we experiment with fastText (Joulin et al. 2016), which has been developed to overcome problems of time-

consuming deep learners. We test each of these approaches separately and also test a variant in which fastText is additionally fed with topics generated by LDA. We have found that both classifiers classify with similar quality. Feeding fastText with LDA-based topics has not accomplished any improvements. However, the combination of both classifiers has enabled us to improve the overall quality of classification.

As a gold standard of topic modeling we use the DDC, which is the most common thematic classification system in (digital) libraries. One advantage of this approach is that it provides access to extensive training and test data. In addition to that we consider two tasks of short text classification in order to enable comparisons with state-of-the-art approaches: the first uses the TREC (Text Retrieval Conference) dataset (Voorhees and Tice 2000), the second the Google Snippets dataset (Phan, L.-M. Nguyen, and Horiguchi 2008). As a result of these evaluations we receive a classifier that allows for determining the topic distribution of all articles of the German Wikipedia so that we can finally model the networking of these topics. In this way, we exemplify how to map text corpora on networks of topics described by them.

The paper is organized as follows: Section 10.3 discusses related work of text classification. Section 10.4 describes the series of topic classifiers used in the experiments of Section 10.5. In Section 10.6, the best performer of this evaluation is applied in order to visualize the thematic structure of Wikipedia. Finally, in Section 10.7 we draw a conclusion and give an outlook on future work.

10.3 Related Work

Since our paper deals with the DDC-related classification of short texts, we consider two areas of related work: text snippet classification and topic modeling used for content analysis of online social networks.

By exploring OAI Metadata, Waltinger et al. (2011) present an SVM-based classifier that considers all three levels of the DDC. A basic restriction of this approach relates to the fact that it only processes OAI records of a certain minimal length. In contrast to this, we do not consider such a lower bound so

that we face a more realistic scenario in which the topic of a snippet is highly underrepresented by its vocabulary. Thus, unlike Waltinger et al. (2011), we consider all 2nd-level DDC categories: in the case of English texts this 2-level approach even deals with a larger set of target classes than the 3-level approach of Waltinger et al. (2011) (who are considering only 88 classes in total). Likewise, we aim at overcoming problems of computational complexity as exemplified by the approach of Brück, Eger, and Mehler (2016). This research shows that DDC-related text categorization, especially by example of short texts, has been a desideratum so far.

The classification of text snippets, regardless of the classification scheme, has made significant progress with the utilization of neural networks for text classification. P. Wang et al. (2016) show that the projection of similar text snippets onto a matrix can be a very helpful input to training a convolutional neural network that outperforms approaches based on other neural networks (Kalchbrenner, Grefenstette, and Blunsom 2014; Y. Kim 2014), LDA (Phan, L.-M. Nguyen, and Horiguchi 2008; M. Chen, Jin, and D. Shen 2011) or SVMs (Silva et al. 2011). These approaches concentrate on single aspects like syntactic rules, topic modeling of text snippets or semantic similarity measurement. Our case study examines sources of information that have not previously been investigated together in the context of classifying text snippets. This includes

1. information about n-grams,
2. information provided by dataset-external semantic knowledge as given by topic models derived from general corpora, and
3. information provided by NLP tools about tokens, lemmas and parts of speech.

We integrate these information sources into our model and compare the performance of a neural network and an SVM-based approach as two competing instances of our model.

The usage of topic models and thematic classifications as an input to graph structures has been explored in different ways. Mostly, the connections in such graphs are built by topical similarities of the documents (J. Chang and D. Blei 2009; Lafferty and David M. Blei 2006). In this way, one can observe,

for example, which sources or authors are highly connected in the resulting graph. On the other hand, social networks can be analyzed with respect to topical preferences manifested by their textual content (McCallum, Corrada-Emmanuel, and X. Wang 2005; Cha and Cho 2012; Gretarsson et al. 2012; Tang et al. 2009). Our approach also adds the network perspective regarding topic distributions. However, we additionally explore the networking of topics as a function of the polysemy of the underlying textual aggregates.

10.4 Models of Topic Classification

In this section we describe the models that we used for topic-related text classification: based on LDA (Sec. 10.4.1), on neural networks (Sec. 10.4.2), on neural networks fed by LDA-based topics (Sec. 10.4.3), on neural networks fed by vectors representing word significance distributions (Sec. 10.4.4) and based on a combination of a SVM and a NNLM-based classifier (Sec. 10.4.5).

10.4.1 LDA-based classification (SVM-LDA)

Topic models, as the *Latent Dirichlet Allocation* (LDA) model, utilize large text corpora to infer a latent distribution of words over a given number of topics so that each document can be described as a mixture of those topics when exploring co-occurrences of their lexical constituents (David M Blei, A. Y. Ng, and Jordan 2003). The parameters of the LDA model, ϕ (word-topic distribution) and θ (document-topic distribution) can be estimated using either a variational inference scheme or Gibbs samplers on a training set of documents (Griffiths and Steyvers 2004). One of the great benefits of topic models is the generalization of the model. The topic structure of documents which do not belong to the training set can be inferred using the fixed model parameters even if additional unknown vocabulary is included. In this way, each document of a corpus can be described in terms of its topic distribution regarding the parameters of a topic model that has been generated by means of a reference corpus.

In text classification, a vector space model is often used to derive elementary features for documents. The famous tf-idf scheme, entropy-based measures or

the pointwise mutual information can be used as alternatives to weight the terms in the document vectors. In Brück, Eger, and Mehler (2016) lexical features are weighted using such term weights. The resulting feature vectors are used to train a *support vector machine* (SVM) using a *Negative Euclidean Distance Kernel* (NDK) on a dataset of 4,000 German DDC classified documents. This approach achieves 0.723 in F-score with respect to the dataset.

Our approach uses additional information besides the tf-idf weights including the extraction of uni-, bi-, and trigrams and the additional use of topics as features within an SVM-classification scheme. That is, we informationally enrich each document in the training set. Unigram stop words are deleted from the set of features and words of the document collection were stemmed. Since a topic encodes an associated vocabulary context (e.g., the word-topic distribution), each document holds general information about other documents containing similar topics. This information can be useful in classification tasks if we augment the lexical features with the topic structure for a category. Our hope is to enhance the results of Brück, Eger, and Mehler (2016) by the use of such topic model-related features. The here described approach uses the LDA model of David M Blei, A. Y. Ng, and Jordan (2003) to infer topics on the dataset.

We considered a novel strategy to augment the lexical features with topic information. An LDA-model with 100 topics is inferred on "general" language data and the topic distributions of documents from both, training and test datasets, are determined with respect to this model.[1] This gives us an additional topic distribution on each document in the training and test sets. The language resource to build our model is based on corpora from the Wortschatz[2] project. We chose 3 million sentences from news data which were crawled in 2015 from German and English websites to build the respective models. We did not use Wikipedia-based data because of the possible domain similarity to our OAI-datasets in Section 10.5.

Additionally, we apply the tf-idf weighting scheme to the document term vectors (uni-, bi- and trigrams) in order to reduce the influence of general vocab-

[1] Our experiments showed that 100 topics provided the best topic solution for the described experiments in terms of F1 performance of the final classifier. We tested 20, 50, 75, 100, 250 and 500 as values for the amount of topics to infer.

[2] `http://wortschatz.uni-leipzig.de/en/download`

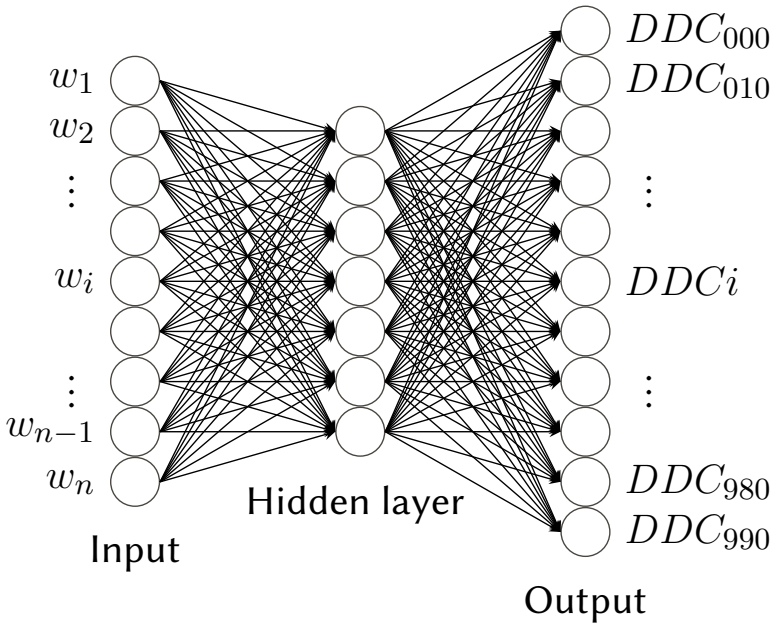


Figure 10.1: Architecture of Model 10.4.2.

ulary. Then, we append the topic distribution for a document as a vector of probabilities to its vector of lexical features. To train the SVM, we used the R-version of liblinear with an L2-regularized logistic regression and estimated the C-parameter heuristically (Fan et al. 2008; Helleputte 2015).

10.4.2 Neural network-based classification (NN)

For the neural network-based approaches, we started with the simple but very efficient classifier of Joulin et al. (2016) called fastText (see Figure 10.1 for a visual depiction of this model in our context). fastText uses a *bag-of-words* (bow) model and defines the occurrences of words in a document as input of the neural network. Since the order of words is ignored in the bow-model, fastText uses n-grams to capture some information about the local order. To avoid being forced to use default parameter settings, we have written a parameter analyzer, which searches the parameter space for better performing settings (according to a hill-climbing algorithm). Since the input corpora were not pre-processed, we applied various NLP tools to obtain additional information about tokens, lemmas and parts of speech. We also used pre-trained word embeddings to initialize the neural network.

10.4.3 Neural Network based classification combined with LDA (NN-LDA)

Since fastText only accepts text as input, we adapted its architecture so that we can process the textual content in conjunction with the topic distribution of a document. To this end, we extended the neural network underlying fastText to include not only input nodes for words, but also for each topic provided by the model of Section 10.4.1. Thus, when considering a distribution of 100 LDA-based topics, we added 100 input nodes to the neural network, which are activated according to the topic values of the input document. In this way, our extension of fastText is additionally fed by numerical values signaling membership to topics derived from LDA.

10.4.4 Neural network-based classification combined with GSS (NN-GSS)

Taking profit of the fact that we adapted fastText to additionally accept numeric values as input, we calculated the GSS coefficient (Galavotti, Sebastiani, and Simi 2000) for each pair of words in the input corpus and first-level categories of the DDC. In this way, each word of the input corpus is mapped onto a 10-dimensional feature vector whose dimensions denote the association of the word with respect to the given target category. Under this regime, the classifier uses feature vectors of GSS coefficients instead of the words themselves. This results in a neural network with $n \times m$ input nodes, where n is the size of the vocabulary of the input corpus and $m = 10$ the number of top-level DDC classes.

10.4.5 Combining both worlds (NN-SVM-LDA)

By means of an error analysis we found that the SVM and the NN-based classifiers make different errors, although achieving similar classification qualities. Therefore, we calculated a scoring for each document with respect to each target category based on the two best performing classifiers from the SVM- and NN-world, respectively and experimented with two methods to combine their

scorings:

1. voting for the target class as a function of the maximum score (not to be confused with majority voting) or
2. by means of the average score.

10.5 Classification Experiment

We test the models of Section 10.4 by example of four different data sets. Two of these samples represent OAI-based datasets (one in German and one in English) which were used regarding two classification tasks. The first task was to classify the first level of the DDC (10 classes in the English corpus (EN-10) and 10 classes in the German corpus (DE-10)). The second task was to classify the second level of the DDC (93 classes in the English corpus (EN-All) and 88 classes in the German corpus (DE-All)). The German corpus consists of 595,493 records with an average of 37.24 words per document. The English corpus consists of 1,222,948 records with an average of 50.69 words per document. Each corpus was randomly divided into training (70%) and test (30%) sets. In order to ensure comparability with state-of-the-art systems for classifying text snippets, we also evaluated our models using the TREC 2003 Question Answering dataset (X. Li and Roth 2002) and the Google Snippet dataset (Phan, L.-M. Nguyen, and Horiguchi 2008) as used in P. Wang et al. (2016).

10.5.1 Classification

For the SVM-based classification using LDA-features we trained one SVM-model for each dataset and task. The results are shown in Table 10.1. The SVM-LDA model outperforms the models described in Brück, Eger, and Mehler (2016) and Waltinger et al. (2011). Furthermore, it performs as good as the NN-based model described in P. Wang et al. (2016). In examining the impact of all features, we find that the n-gram features have an impact similar to features provided by the topic model. The combination of both feature sets does not improve overall performance. In detail, the classification for the DE-10

Corpus	Features	N-gram	F-scores
EN-10	NN: token-based	3	0.748
EN-10	SVM-LDA	1-3	0.771
EN-All	NN: token-based	3	0.698
EN-All	SVM-LDA	1-3	0.717
DE-10	NN: token-based	1	0.814
DE-10	NN: lemma + POS	2	0.816
DE-10	NN-GSS	–	0.792
DE-10	NN-LDA: lemma + POS + topics	2	0.795
DE-10	NN (lemma + POS) + SVM-LDA	1-3	0.820
DE-10	SVM	1-3	0.814
DE-10	SVM-LDA	1-3	0.815
DE-All	NN: lemma + POS	2	0.757
DE-All	NN: token-based	2	0.753
DE-All	SVM-LDA	1-3	0.750

Table 10.1: F-scores of text snippet classification based on four different corpora.

dataset results in the following F1-scores for the different feature configurations: *1. unigrams: 0.786; 2. unigrams + topics: 0.805; 3. n-grams: 0.814; 4. n-grams + topics: 0.815.* In general, it can be shown that using topic model features improves the quality of the classification, albeit to a limited extent. From a classification point of view, n-grams and LDA-based topics seem to encode related information within the feature space. This may give rise to future research.

In the case of classifying with neural networks, we carried out a parameter study to detect optimal parameter settings. To this end, we examined the following parameters:

- Learning rate (0.025 - 0.1)
- n-grams (1 - 5)
- Dimension (50 - 100)
- Epochs (500 - 10,000)

The results are shown in Table 10.1. It shows that SVM-LDA performs better than its NN-based counterparts in the case of the English data sets, while

Method	Google Snippets	TREC
SVM-LDA (Section 10.4.1)	0.960	0.971
NN (Section 10.4.2)	0.962	0.974
P. Wang et al. (2016)	0.851	0.972

Table 10.2: Comparison of our models to the best performing model in P. Wang et al. (2016).

the NN-based (lemma + POS) classifier outperforms its competitors in the case of the German data. However, the difference to SVM-LDA is very small. Additionally feeding the NN with LDA topics (NN-LDA) performs worse than NN-GSS (DE-10). Further, lemma-level features perform very little better than token-level ones (DE-10 and DE-All).

Next, we selected the best classifiers of both areas (SVM and NN) and further analyzed their classification quality. Although both classifiers perform similarly (81.4% and 81.6%), they make different mistakes. When always knowing the right class of a snippet and then selecting the classifier voting for it, we would achieve an F-score of 89.6% as a kind of an upper bound of an algorithmic combination of SVM-LDA and NN (lemmas + POS). However, we cannot presuppose this knowledge. Thus, we need to apply one of the combinations of Section 10.4.5. This produces an the F-score of 82% in the DE-10 experiment using the method of averaging scores.

Finally, we compared the best performers of Table 10.1 with those documented by P. Wang et al. (2016): Table 10.2 shows that we also outperform these competitors by example of the Google and the TREC data. Obviously, our approach is more than just competitive.

10.5.2 Discussion

Although we worked with the complete DDC corpus (as described at the beginning of this section) and therefore had to classify many small texts, we achieved rather promising classification results. This holds for both the SVM-LDA and the NN-based classifier. Both classifiers outperform the approach of Waltinger et al. (2011) (being based on a classical SVM) and the one of Brück, Eger, and Mehler (2016) (using a newly invented kernel function), even

when using the full dataset rather than using only a subset of texts of a certain minimal length. In addition, both our classifiers outperform their competitors described in P. Wang et al. (2016) (see Table 10.2).

In the case of SVM-LDA, we show that information provided by LDA has a positive impact on classification. The different errors generated by SVM-LDA and NN indicate that there is a high potential in the combination of both approaches. However, the neural network achieved worse results when directly using topic information provided by the LDA (NN-LDA – see Table 10.1). Therefore, information about topics as provided by LDA should be integrated into neural networks in other ways than by the one used here so that one can make better use of this information. The very same can be said about using GSS-weighted vectors (NN-GSS). Experiments of J. Kim, Rousseau, and Vazirgiannis (2015) and Y. Kim (2014) show the potential of including word similarity information within a convolutional layer of a neural network. This type of semantic smoothing might also be interesting to explore similarities of documents that are used simultaneously for training the network. In this way, we may help to better integrate topic models and neural networks. This will also be an object of future research. In any event, we are now in a position to guess for any piece of text – down to the level of single words – what topic class of the DDC it likely belongs to. In this way, we have a very powerful topic model that can be used to study the topic distribution and topic networking of online social networks and related media.

10.6 A bird's eye view of topic networks

In this section, we experiment with the best performing (non-combined) topic classifier of Section 10.5, that is, NN (lemma + POS, DE-All), to model inter-topic structures. This is done by example of a complete release of the German Wikipedia (download: January 20th, 2017). That is, each of the 1,760,875 Wikipedia articles in this release is mapped onto a subset of DDC categories and each of the 53,122,347 links between these articles is mapped onto arcs between nodes denoting these categories. We address two tasks:

- *Topic distribution and thematic dominance:* Firstly, we try to determine for each article of this release what topics it deals with. This means that

we assume a multi-label classification scenario in which the same article possibly manifests several topics to varying degrees (measured by the strength μ of classification).

- *Topic linkage:* Secondly, we use this information to generate a network that shows how these topics are interlinked. Through this network we provide two types of information: about the salience of topics and about topics being jointly manifested by articles.

- *Visualization:* Our visual depiction of this topic network is based on the following statements:

    1. The more articles describing the same topic and the stronger they do, the more salient this topic becomes and the bigger its visual depiction.

    2. The more articles related to the topic A are linked with articles related to the topic B, the larger the visual representation of the arc from A to B.

The result of this visualization procedure is depicted in Figure 10.2 (a). It demonstrates that articles are usually so ambiguous (in terms of our classifier) that applying this algorithm of network induction to *all* Wikipedia articles ultimately brings us close to a completely connected topic network. Thus, in order to reveal more structure, we additionally experiment with varying thresholds of minimal classificatory membership by analogy to α-cuts in fuzzy set theory. This is demonstrated in Figure 10.2 (b): it shows that for a threshold of maximum class membership, we arrive at an extremely sparse network in which only a tiny fraction of topic-to-topic links survive. At this level, inter-topic structure almost diminishes: a single highly salient category emerges, that is, DDC class 790 (*Recreational & performing arts*). Note that in Figure 10.2 (b), salience of vertices is *also* a function of α: only those categorizations are counted per DDC class whose membership value μ is at least α; the same constraint also concerns the linkage of topic nodes.

Now the question is raised how the network of Figure 10.2 (b) passes over into the one of Figure 10.2 (a): how does it move from crisp to fuzzy categorization? In order to answer this question, we compute networks according to our algorithm of network induction by taking only those mappings of articles

x to DDC categories A into account, whose class membership $\mu_A(x)$ satisfies the inequality $\mu_A(x) \geq \alpha$ while reducing α stepwise from 1 to 0.01 (in steps of 0.01). Then, for each of these α values we induce a separate network for which we compute a subset of graph invariants as depicted in Figure 10.3:

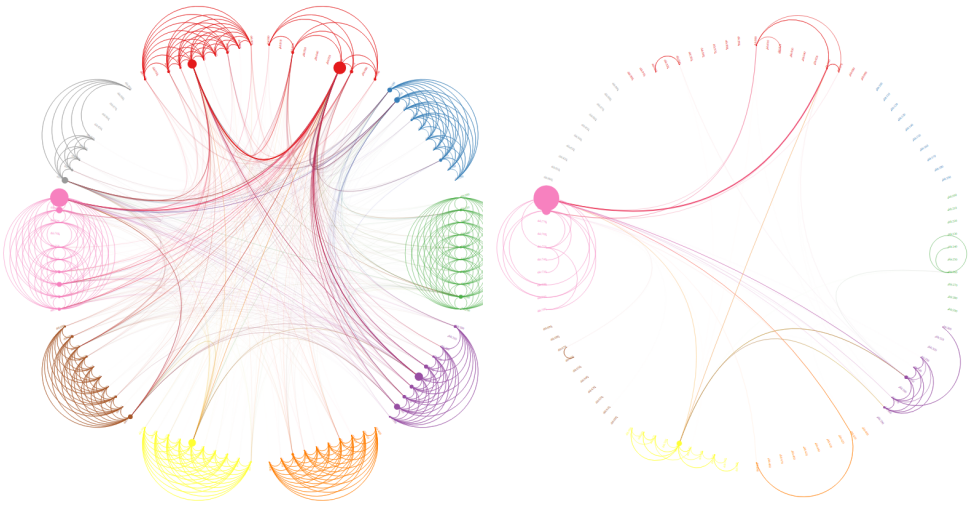
1. The unweighted C (Watts and Strogatz 1998) and the weighted cluster value C_d^w of directed networks (Barrat, Barthélemy, and Vespignani 2007) estimating the probability with which nodes linked from the same node are themselves connected, taking into account the weights of these arcs.

2. The proportion of vertices belonging to the largest strongly lcc_s and weakly lcc_w connected component.

3. The cohesion value coh, that is, the proportion of existing arcs in relation to the number of possible arcs.

Finally, we plot aggregated values of graph invariants, that is, the product of C_d^w and C on the one hand and of coh and lcc_s on the other. We observe that compared to C_d^w (C), the values of $C_d^w \cdot coh$ $(C \cdot coh)$ are significantly smaller. This indicates that although clustering rapidly increases even for smallest decreases of maximum α, clustering rather concerns a small subset of vertices. At the same time, we observe that by weighting C_d^w with lcc_s, clustering does not decrease by far to the same extent (the same holds, though to a higher degree, for $C \cdot lcc_s$). This suggests that adding arcs as a result of reducing α contributes more to the connectivity than to the clustering of the underlying networks. In other words: increasing the level of allowable ambiguity rather leads to connected topic networks than to networks exhibiting many local (triadic) clusters. If we compare the distribution of C_d^w as a function of α with C, we observe that for smaller values of $\alpha = 0.4$ C_d^w starts shrinking as C continues to grow: for this threshold value, smaller weights of edges begin to overlay higher edge weights. In other words, at this level, the categorization quickly becomes too much blurred. In any event, we also observe that under our model of topic classification, articles tend to be highly polysemous so that one rapidly approximates a highly connected graph ($lcc_w \sim 1$) that also exhibits high cluster ($C > 0.8$) and cohesion values ($coh > 0.2$).

Obviously, this analysis provides both (i) a bird's eye view on topic structuring



(a) alpha=0 (b) alpha=1

Figure 10.2: Comparison of the Wikipedia based DDC network with alpha = 0 and 1.

● DDC 0, ● DDC 1, ● DDC 2, ● DDC 3, ● DDC 4, ● DDC 5, ● DDC 6, ● DDC 7, ● DDC 8, ● DDC 9

as manifested by text networks as large as Wikipedia and (ii) an assessment of its ambiguity. The latter is done by analyzing the transition dynamics starting from clear classifications to highly ambiguous ones, taking into account clustering and connectivity.

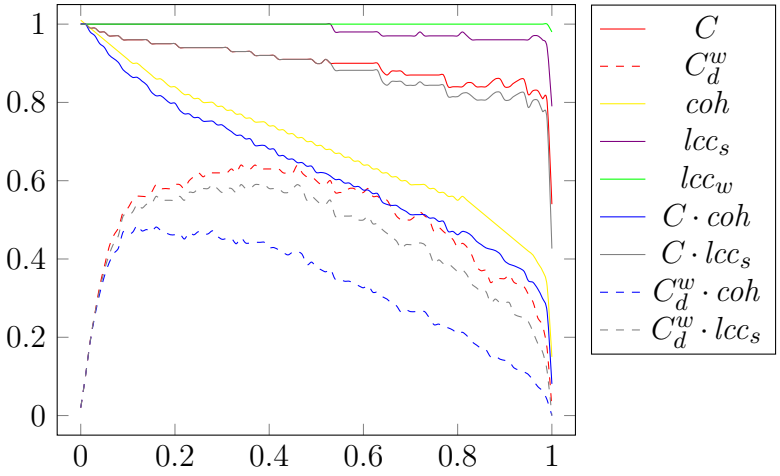


Figure 10.3: Distribution of graph invariants of topic networks as a function of minimal class membership α.

10.7 Conclusion

In this paper, we developed a simple algorithm for analyzing and visualizing the topic structure of large text networks. To this end, we experimented with a series of classifiers in the context of three evaluation scenarios. This included an SVM-based classifier exploring topics derived from LDA, a NNLM-based classifier (i.e, fastText) as well as combinations thereof. Using the best performer of these experiments, we have shown how to generate a bird's eye view of the salience and linkage of topics as manifested by hundreds of thousands of texts. In this context, we observed a very high degree of thematic ambiguity, which makes it necessary to search for more precise, less ambiguous classifiers. This will be the task of future work. Nevertheless, our paper shows a way to automatically visualize the thematic dynamics of textual aggregates as produced by large online social networks.

11 Computing Classifier-based Embeddings with the Help of text2ddc

The content of this chapter (Uslu, Mehler, and Baumartz 2019) was published in Proceedings of the 20th International Conference on Computational Linguistics and Intelligent Text Processing (CICLing 2019).

11.1 Abstract

We introduce a method for computing classifier-based semantic spaces on top of text2ddc. To this end, we optimize text2ddc, a neural network-based classifier for the *Dewey Decimal Classification* (DDC). By using a wide range of linguistic features, including sense embeddings, we achieve an F-score of 87.4%. To show that our approach is language independent, we evaluate text2ddc by classifying texts in six different languages. Based thereon, we develop a topic model that generates probability distributions over topics for linguistic input at the word (sense), sentence and text level. In contrast to related approaches, these probabilities are estimated with text2ddc, so that each dimension of the resulting embeddings corresponds to a separate DDC class. We finally evaluate this *C*lassifier-b*a*sed *S*emantic spac*e* (CaSe) in the context of text classification and show that it improves the classification results.

11.2 Introduction

We present a model for calculating neural network-based *Classifier-Induced Semantic Space*s (CaSe) using the Dewey Decimal Classification (DDC), that is, an international standard for topic classification in libraries. Based on this model, input units on the sense-, word-, sentence- or text level can be mapped onto the same feature space to compute, for example, their semantic similarity (Bär et al. 2012; Pilehvar and Navigli 2015). Such an approach is needed whenever multiresolutional semantic information has to be processed to interrelate, for example, units of different levels of linguistic resolution (e.g., words or phrases to texts). Contrary to related approaches (Landauer and Dumais 1997; David M Blei, A. Y. Ng, and Jordan 2003) we use classifiers to define the dimensions of CaSe, which are directly labeled by the underlying target class. This has the advantage that embeddings of linguistic units in semantic spaces can be interpreted directly in relation to the class labels.

To this end, we generate several DDC corpora by exploring information from Wikidata, Wikipedia and the Integrated Authority File (*Gemeinsame Normdatei* – GND) of the German National Library. Since Wikipedia is offered for a wide range of languages, such corpora can be created for different languages. In this paper, we focus on Arabic, English, French, German, Spanish, and Turkish while performing a deeper analysis by example of the German corpus.

Our *Classifier-Induced Semantic Space* (CaSe) utilizes a classifier that is a further development of a feedforward neural network (FNN) which has been evaluated previously in text classification and automatic disambiguation (Uslu, Mehler, Baumartz, et al. 2018; Uslu, Mehler, and Baumartz 2019). In this paper, we optimize this classifier with respect to feature selection, extend it to the 3rd level of the DDC (comprising up to 641 target classes) and train it for six languages. To this end, we consider a wide range of pre-processing steps including lemmatization, part of speech (POS) tagging, word sense disambiguation (WSD) and sense embeddings.

Our approach is in line with Pilehvar and Navigli (2015) and, thus, disambiguates input words to obtain sense representations as input for calculating sense embeddings. To this end, we disambiguate the entire German Wikipedia

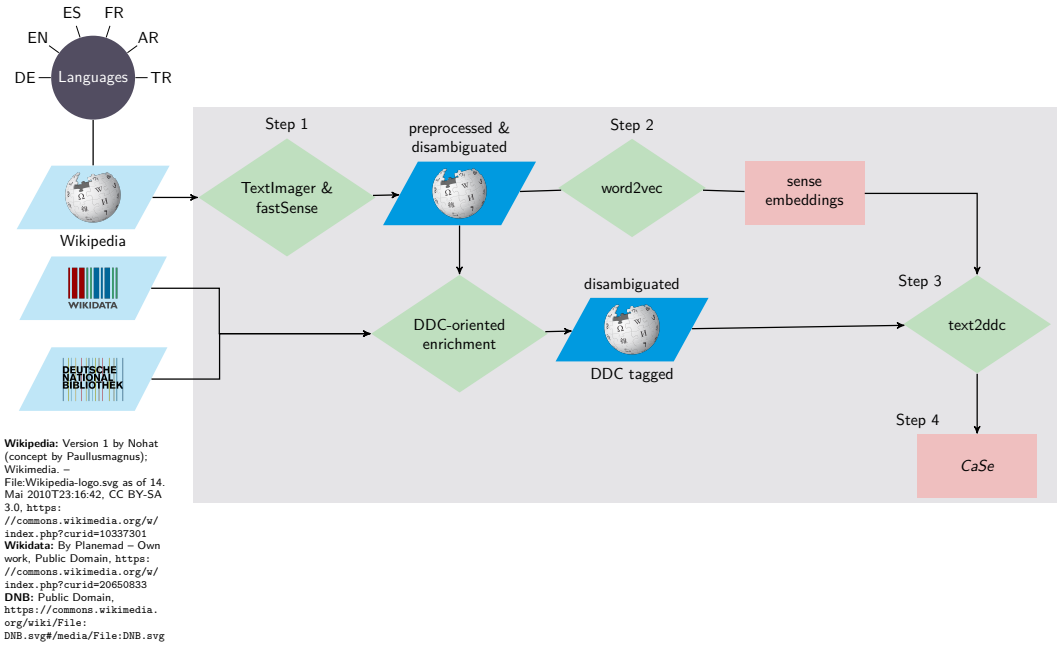


Figure 11.1: Flowchart of the experimental framework.

and calculate sense embeddings using word2vec (Mikolov, K. Chen, et al. 2013). Based on this approach, we achieve the best classification results. This is shown for the 2nd (including up to 98 classes) and the 3rd level of the DDC (including up to 641 classes).

Using the trained and evaluated DDC-related classifier, we derive a novel model of CaSe. For each sense-, word-, or text-level input CaSe generates a probability distribution over the DDC classes (of either the 2nd or 3rd level). In this way, each input unit is mapped onto an n-dimensional feature vector whose dimensions are uniquely labeled by the corresponding DDC classes. In order to demonstrate the expressiveness of CaSe, we conduct two classification tasks and show that using CaSe-based feature vectors improve both classifications.

The paper is organized as follows. Section 11.3 discusses previous literature related to this article. In Section 11.4 we describe the models and corpora used for the experimental framework. The main studies and results are presented in Section 11.5, followed by a discussion of the outcome and an error analysis in Section 11.6. Section 11.7 provides some final conclusions and directions for future work.

11.3 Related Work

In this section, we review related work based on similar approaches, methods or experiments.

Waltinger et al. (2011) and Brück, Eger, and Mehler (2016) introduce SVM-based topic classifiers regarding the DDC as the target scheme. However, these approaches require a minimum number of words and thus reduce the number of classes. In previous work (Anonymous, 2018), we introduced a topic classifier which outperforms these approaches and even considers the third level of the DDC. Text classification, regardless of the classification scheme, has made significant progress by means of neural networks. Kalchbrenner, Grefenstette, and Blunsom (2014) and Y. Kim (2014) show, for example, that convolutional neural networks perform comparable to the state-of-the-art in text classification. Joulin et al. (2016) present an FNN-based classifier consisting of only one hidden layer. They show that this classifier is competitive, even though it is many times faster than its competitors.

Zhang and LeCun (2015) also experiment with neural networks in text classification by considering different classification tasks. In the present study, we adopt their scenarios to evaluate our DDC-based CaSe.

The best-known topic model is probably the one being based on Latent Dirichlet Allocation (LDA) as introduced by David M Blei, A. Y. Ng, and Jordan (2003). Irrespective of its outstanding applications, a central problem of this approach is that topics being detected are not directly labeled. This makes it difficult to interpret the resulting topic distributions assigned, for example, to input texts. There exist many approaches to automatically labeling topics as detected by LDA by means of heuristic methods (Mei, X. Shen, and Zhai 2007; Magatti et al. 2009). In this way, topic labels are derived from the underlying training corpus. In contrast to this, our topic model refers to the DDC as a pre-established standard used by libraries worldwide for topic classification. Thus, any of the topic labels used by our CaSe can be directly interpreted and related to other classifications for which the DDC has also been used.

To this end, we perform a feature analysis which is related to J. Li and Jurafsky (2015) who analyze the effect of sense embeddings on tasks in natural language processing (NLP). J. Li and Jurafsky (2015) compute sense embeddings based

on a manually generated dictionary (e.g., WordNet). The embeddings are then assigned to tokens of input texts to disambiguate them. Several classification tasks are conducted to evaluate this approach. They show that sense embeddings lead to improvements compared to word-related embeddings.

Vial, Lecouteux, and Schwab (2017), Q. Li, T. Li, and B. Chang (2016), and Pelevina et al. (2017) also generate sense embeddings by heuristically utilizing classical word embeddings. Q. Li, T. Li, and B. Chang (2016) use synonyms with only one sense and try to learn sense embeddings based thereon. Vial, Lecouteux, and Schwab (2017) compute sense embeddings using the normalized sum of all word embeddings that occur in a dictionary for the current sense. Pelevina et al. (2017) create a graph based on word embeddings and use cluster algorithms to create sense embeddings.

In this paper, we learn sense embeddings similar to the approach of Iacobacci, Pilehvar, and Navigli (2015). Iacobacci, Pilehvar, and Navigli (2015) disambiguate the complete Wikipedia with the help of Babelfy[1] and calculate sense embeddings using word2vec (Mikolov, K. Chen, et al. 2013). In summary, by looking at different NLP tasks, these approaches show that sense embeddings are effective and usually outperform their word-based competitors.

CaSe have been introduced by Leopold (2007) who uses *support vector machine*s (SVM) to induce the dimensions of the semantic space. Contrary to this approach, we compute CaSe by means of sense embeddings combined with neural networks.

In this way, we extend previous models of semantic spaces by providing a largely language-independent (Wikipedia- and Wikidata-based) approach that differentiates between the 2nd and the 3rd DDC level to provide semantic spaces of different granularity. This also means that CaSe generates low-dimensional spaces that are much more compact than feature spaces derived from semantic networks.

[1] www.babelfy.org

mouse (computer)	mouse (rodent)	ball (sport)	ball (dance)
remote control	cat	hockey puck	masquerade ball
keyboard	rat	bat	clown
Wii remote control	snake	batsman	costume ball
scanner	tick	strike	dinner
video camera	cavy	ball carrier	dance night
touchpad	ant	basket	stag party

Table 11.1: Nearest neighbors of ambiguous words (both translated from German to English) using cosine similarity operating on the space of sense embeddings.

Language	articles	avg tokens/article	classes (2nd)	classes (3rd)
German	15,136	1,228	98	641
English	10,991	2,799	95	603
Arabic	9,910	788	96	591
Turkish	7,998	539	94	566
French	12,406	1,740	96	616
Spanish	13,221	2,053	95	620

Table 11.2: Training & test corpora for six languages, number of articles, average number of tokens per article, number of target classes on the 2nd and 3rd level for which the apparatus of Figure 11.1 was trained.

11.4 Model architecture

The overall architecture of the experimental framework is depicted in Figure 11.1. It consists of four steps: in step 1 we use TextImager (Hemati, Uslu, and Mehler 2016) for pre-processing and fastSense (Uslu, Mehler, Baumartz, et al. 2018) to disambiguate the German Wikipedia. Currently, the focus of fastSense is on the disambiguation of nouns. The disambiguated Wikipedia is then used in Step 2 to create sense embeddings by means of word2vec (Mikolov, K. Chen, et al. 2013). The aim is to obtain disambiguated articles and sense embeddings for training a DDC classifier in Step 3 and thus generating text2ddc. For this we enrich the disambiguated Wikipedia articles with DDC information using Wikidata and GND. Next, in Step 4, we utilize text2ddc to generate CaSe for a given input text: in this way, each input unit on the sense-, word- or text-level can be mapped onto an n-dimensional feature vector whose di-

mensions correspond to DDC classes.

The following subsections describe the models and data used to generate CaSe.

11.4.1 Step 1 & 2: Word Sense Disambiguation

Word Sense Disambiguation (WSD) is performed by means of fastSense (Uslu, Mehler, Baumartz, et al. 2018). fastSense is trained on the entire German Wikipedia. For this purpose, the Wikipedia's link structure is explored to resolve ambiguous words. The context information at paragraph level of any ambiguous token is given as input. In this way, the neural network based classifier learns which senses are more likely to be associated in a given context. fastSense achieves an F-score of over 80% in disambiguating the German Wikipedia and state-of-the-art results concerning various SemEval and Senseval WSD tasks. Furthermore, fastSense is very time efficient compared to related taggers. It disambiguates the entire German Wikipedia in just a few hours. Using the resulting corpus, we created sense embeddings by means of word2vec (Mikolov, K. Chen, et al. 2013). Table 11.1 exemplifies the results by example of two ambiguous words and their nearest neighbors computed by our sense embeddings.

11.4.2 Step 3: Classifier

We introduce a classifier called text2ddc, to classify an input on the sense-, word, sentence- or document-level regarding the DDC as the target classification. text2ddc is based on a FNN and is able to use word embeddings as additional input. We extend text2ddc by alternatively using sense embeddings combined with a disambiguated corpus and many more features (see Table 11.4).

11.4.3 Step 4: Classification scheme

We use the 2nd and 3rd level of the Dewey Decimal Classification (DDC) as two alternative classification schemes. The DDC includes three levels of the-

matic resolution: the first level distinguishes 10 main topics, each of which is subdivided into maximally 10 topics on the 2nd level (summing up to 99 classes), which in turn are subdivided into maximally 10 topics on the 3rd level (summing up to 915 classes). For example, DDC 500 denotes *Natural Sciences* and contains the subtopics *Physics* (DDC 530), *Chemistry* (DDC 540) and *Life Sciences* (DDC 570). On the 3rd level one finds more specific classes such as *Magnetism* (DDC 538) and *Light & Related Radiations* (DDC 535). References to the DDC are provided by Wikidata and the GND of the German National Library. Since many Wikipedia articles refer to Wikidata or the GND, we were able to explore these articles as training examples of the corresponding DDC classes. In addition, translations provided by both Wikipedia and Wikidata enable the creation of language-specific training corpora by evaluating translation relationships between articles assigned to the DDC and articles for which these assignments do not exist. Table 11.2 lists the statistics of the training and test corpora that we generated for six target languages.

11.5 Experiments

In this section, we report on the results of evaluating Step 3 (text2ddc) and Step 4 (CaSe) of our algorithm depicted in Figure 11.1.

11.5.1 Evaluating text2ddc

We evaluate text2ddc with regarding the question which features are most successful in DDC-oriented text classification. To this end, we split the corpora described in Section 11.4.3 into 80%/20% for training/testing. We used the following standard parameters to train the FNN underlying text2ddc:

- number of hidden layers: 1
- hidden layer dimension: 100
- learning rate: 0.1
- update rate: 100

Content	Epoch 100	Epoch 10,000
Article	0.8	0.822
Section	0.56	0.744
Paragraph	0.658	0.707
Sentence	0.698	0.592

Table 11.3: F-scores considering different contents of the corpus.

- minimal number of word occurrence: 1
- number of negatives sampled: 5

text2ddc by example of the German Wikipedia

We started by testing which parts of an article have the greatest impact on classification when being used for training. That is, for each article in the training corpus, we alternatively trained text2ddc by means of the first sentence, the first paragraph, the first section and the complete article. We did

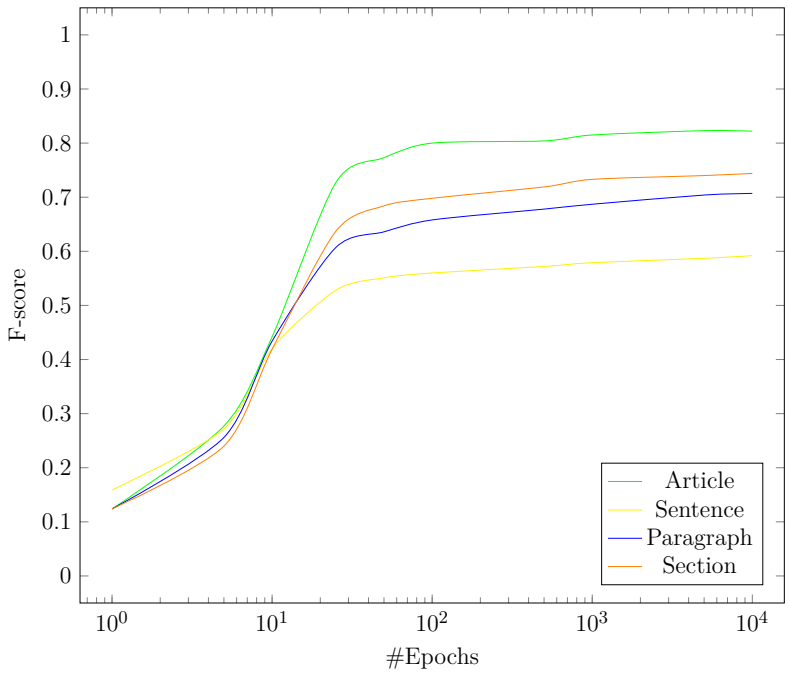


Figure 11.2: The effect of different units of the logical document structure (first sentence, first paragraph etc.) and of different numbers of epochs on F-score.

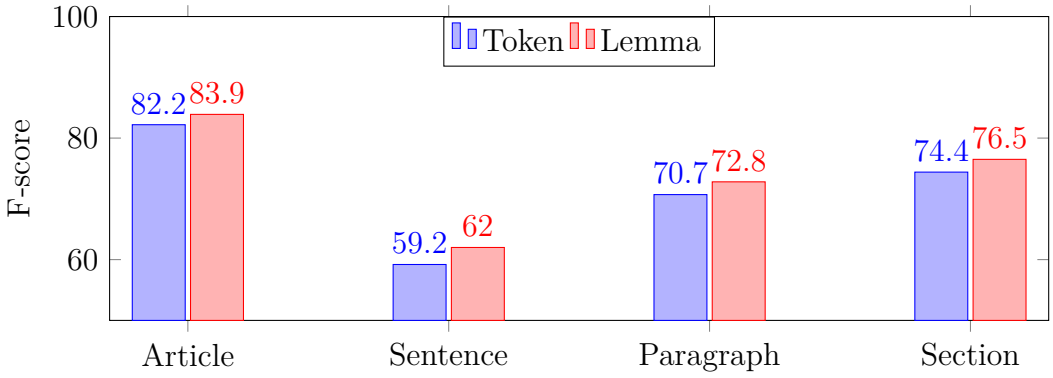


Figure 11.3: The effect of lemmatization on classification.

so by additionally testing the number of training epochs as a test parameter. Figure 11.2 shows that using the entire article as input achieves best results. However, the use of the first paragraph or section as input leads to comparable results. In any event, the larger the input, the better the results. We also see that 100 epochs are minimally required to achieve an F-score of at least 50%.

Now that we detected the optimal input unit and the minimal number of epochs, we tested additional features to improve the classification. This includes lemma information of the input text streams. Figure 11.3 shows that lemmatization improves the results in all experiments. This makes sense because in this way, one gets more information about the association of DDC classes and text content. For example, `prays`, `prayed` and `praying` are wordforms of the same lexeme *pray*. Under this regime, rare occurrences can still be covered because of being mapped onto the corresponding lemma, even if, for example, the wordform *praying* was never observed in training, it can be processed later on, when other wordforms of the same lemma *pray* have been observed. In addition, we also consider lexical information from different sources to measure their effect on classification. This includes Wikipedia categories and Wikidata properties. That is, we used the Wikipedia categories assigned to articles in the training corpus as additional input for training text2ddc and alternativly did the same using the Wikidata properties of the corresponding Wikidata article. Figure 11.4 shows that Wikipedia categories improve the classification, while Wikidata properties worsen it. Therefore, we excluded Wikidata properties from the subsequent evaluation.

Nr	Features	Epoch 100	1000
1	Lemma	80.8%	
2	1 + Categories	82.1%	
3	2 + Wikidata	81.4%	
4	2 + Embeddings	84.9%	85.5%
5	4 + Disambiguation	84.9%	86.7%
6	5 + No functors	85.0%	87.1%
7	6 + Subwords	85.3%	86.7%
8	6 + Word n-grams	84.7%	

Table 11.4: Feature analysis by example of the German Wikipedia.

Language	DDC 2	DDC 3
German	87.4%	78.1%
English	79.8%	72.6%
Arabic	79.8%	68.8%
Turkish	78.9%	67.5%
French	79.4%	68.1%
Spanish	79.7%	70.5%

Table 11.5: F-scores for different languages for 2nd and 3rd level DDC.

Next, we examined the impact of embeddings, disambiguation and function words. Since the results of these features are very close to each other (see Table 11.4), we also carried out these experiments with 1,000 epochs. This shows that word embeddings improve the classification, but by means of disambiguation we can perform even better. Moreover, the removal of function words also improves the classification. Next, we experimented with sub-word units and n-gram features, showing that they do not improve classification. In fact, word n-grams (2 and 3-grams) even worsen the results (see Table 11.4).

After determining which approach based on the standard parameters achieves the best results, we conducted a parameter study to find the optimal hyper-parameters. This includes the following parameters:

- learning rate: 0.2
- update rate: 150
- minimal number of word occurrences: 5

Input	DBpedia	AG News
Text without CaSe	97.89%	89.88%
Text + CaSe (DDC 2)	98.00%	90.18%
Text + CaSe (DDC 3)	98.06%	90.33%

Table 11.6: F-scores in the DBpedia and AG News classification tasks.

- epochs: 10,000

In this way, we have increased the F-score to 87.4%.

Tackling language independence

By means of the language-specific corpora that we generated for different languages (see Section 11.4.3) we additionally trained text2ddc for Arabic, English, French, Spanish, and Turkish. Table 11.5 shows that though text2ddc performs worse in the case of the latter five languages compared to German, the results for the 2nd level of the DDC are nevertheless close to 80%. This makes it possible to determine the topic of linguistic units (senses, words, sentences etc.) of any of these languages to a remarkable degree. This also demonstrates that our approach is largely language independent at least what concerns languages that are sufficiently manifested by language specific releases of Wikipedia.

Third Level DDC

In order to perform a more detailed analysis, we also analyzed text2ddc with respect to the 3rd level of the DDC. This increased the number of target classes, however, any topic vector is enriched by providing more detailed information. Table 11.2 shows the frequency distributions of the DDC classes and the number of examples for each language. Table 11.5 shows the corresponding results: we observe a drop in F-score when switching to the 3rd level, while in the case of the German Wikipedia we still perform at about 78% – a result that is similar the one that we observed for the other languages on the 2nd level of the DDC.

11.5.2 Evaluating CaSe

To show that our DDC-based topic model improves classification, we performed two classification tasks. To this end, we consider two data sets: the DBpedia Ontology Classification Dataset and the AG's news corpus[2]. The DBpedia Ontology Classification Dataset is created by selecting 14 non-intersecting classes from DBpedia 2014[3]. Zhang and LeCun (2015) constructed a topic classification dataset using the AG's news corpus by selecting the 4 largest classes.

For each input text of these two datasets we generated CaSe-related feature vectors. That is, each text is represented by a feature vector of DDC classes of either the 2nd (DDC 2) or 3rd level (DDC 3). Next, we trained a FNN-based classifier that uses these feature vectors as input in addition to the input texts. That is, the FNN is trained in such a way that the words of the input text are presented to the input layer in conjunction with 98 (DDC 2) or 641 (DDC 3) input neurons representing the corresponding DDC classes. For any input word the corresponding input neuron is activated with weight 1. The DDC input neurons are activated using the output values generated by text2ddc for the given text and DDC class. To be independent of the classifier, this experiment was conducted by means of StarSpace (Wu et al. 2017), a text classification framework developed by Facebook's research team. Table 11.6 shows the results and the impact of CaSe. The improvements over purely the text-based classifier are not very large, but with such a high classification quality (in the case of DBpedia over 97%), every percentage is important. These two experiments document an impact of CaSe on text classification.

11.6 Discussion

We have increased the performance of the DDC-related classification to over 87% for German. This is remarkable considering that our corpus was automatically generated from Wikipedia articles in conjunction with data from Wikidata and GND. That is, we developed CaSe as an approach that maps any lexical or textual input onto probability distributions over DDC classes. In

[2] www.di.unipi.it/ gulli/AG_corpus_of_news_articles.html
[3] www.wiki.dbpedia.org/data-set-2014

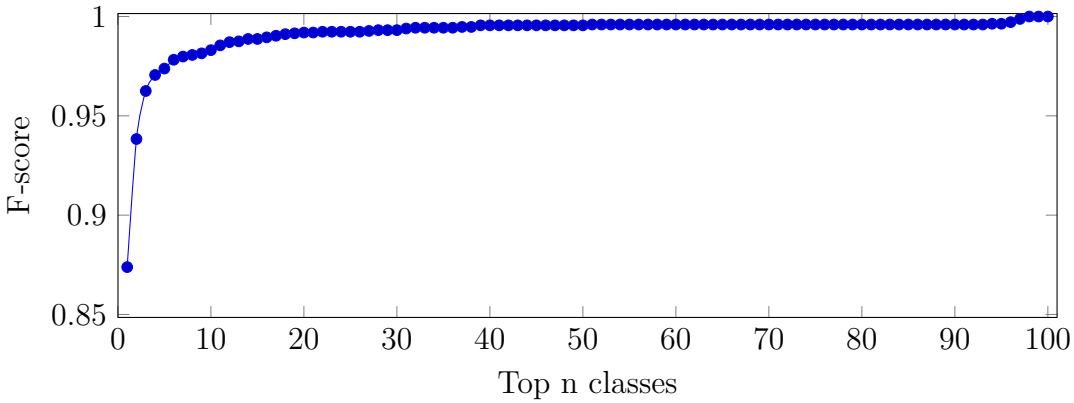


Figure 11.4: Analysis of the F-score by considering the top n classes.

this way, it is possible to automatically label the topic of this input, referring to interpretable subject names as provided by DDC. On the other hand, topic vectors provided by CaSe can also be used as additional input for classification experiments. We have shown that these vectors improve classification by example of two tasks based on DBpedia and AG News data.

11.6.1 Error analysis

We additionally performed an error analysis of the DDC classification. We examined which classes are most often misclassified. On the other hand, we also investigated the performance when considering the top n predicted classes. In the previous experiments we only selected the class with the highest score. Here we discovered that we already achieve a score of over 96% when considering the top 3 classes (see Figure 11.4). We also discovered that the most common errors occur since some classes are quite similar. One of the most common mistakes is the tagging of DDC 590 (Zoological sciences) instead of DDC 560 (Paleontology). This is very comprehensible, because animals are mentioned in both topics. Another common mistake is the classification of DDC 530 (Astronomy & allied sciences) instead of DDC 520 (Physics). Thus, we recognize that the remaining errors are not necessarily actual errors, but rather similar topics which still might have a high probability.

We also analyzed the average number of words in a successful and unsuccessful classification (see Table 11.7). Here we discovered that the unsuccessful classifications always contain fewer words on average than the successful ones.

Language	Successful	Unsuccessful
German	3,364.40	3,066.72
English	3,031.78	2,503.29
Arabic	880.36	684.28
Turkish	570.44	395.32
French	2,119.06	1,768.47
Spanish	1,808.43	1,237.92

Table 11.7: Tokens per example. A comparison between successful and unsuccessful classification.

The more data the classifier receives, the better the classification.

11.7 Conclusion

We have presented a neural network based classifier to categorize DDC classes. For this we have used various features and resources to achieve the best possible classification. This includes POS tagging, lemmatizing and disambiguating the German Wikipedia. Together with this information, we have managed to achieve a classification quality of over 87%. Considering the top three classes, we even exceed 96%. For a given text, the classifier generates a probability distribution over the DDC classes and thus a vector. This vector can be used as input for other classification tasks and we have shown that improvements can be achieved.

We have also trained the classifier in English, Spanish, French, Arabic and Turkish in order to be able to use these vectors in different languages. In future work we would optimize the classifiers of the different languages in the same way we did for the German version. We offer the classifier (text2ddc) and the DDC topic model (CaSe) as API for all above mentioned languages.

12 text2wiki - a Dynamic Open Topic Model by means of the Wikipedia Category System

The content of this chapter will be submitted in Scientometrics, an International Journal for all Quantitative Aspects of the Science of Science, Communication in Science and Science Policy.

12.1 Abstract

Topic models are becoming more and more important due to the increasing amount of textual data in the world wide web. In this way, a semantic value is assigned to the texts and thus prepared for further applications. As the availability of data increases, so does the thematic universe, which must be taken into account. Therefore we present text2wiki, an open topic model based on the constantly evolving category system of Wikipedia. We have trained models in different semantic levels and have shown that we can analyze the entire Wikipedia in just a few hours. text2wiki not only convinces with its efficiency, but also with its quality, as we achieve F-scores of over 80%. We have also analyzed the dynamics of the Wikipedia category system over a period of one year and observed that the rate of change is over 10%.

12.2 Introduction

Topic modeling is becoming increasingly important as more and more digital data becomes available and automatic analysis of this data is imperative. Topic analysis can be used in a variety of areas. For example, the topic of literatures or scientific papers can be analyzed for automatic classification in libraries or scientific conferences. They can also be used to analyze the distribution of topics in texts or corpora. There are already well-established topic models available, but most of them are unsupervised and therefore not based on an existing thematic concept. Accordingly, these models are not provided with a label, making them difficult to interpret. Then there are supervised topic models requiring the topics and the underlying texts as prerequisites. These models usually utilize an existing thematic model, which on the one hand covers only a small part of the thematic universe and on the other hand does not adapt over time. As a result, many of these models are considered obsolete and do not include more recent topics. We call these *closed topic models* (CTM). In this paper we present `text2wiki`, an *open topic model* that is constantly maintained by exploring Wikipedia's category system. text2wiki continuously searches for categories to be added or deleted according to a selection procedure. Wikipedia is a perfect source for topic analysis as it contains millions of articles labeled with categories. Every day, thousands of new articles are created, which are provided with up-to-date categories. Working with Wikipedia as basis for text2wiki requires highest efficiency for our models. Therefore, we rely on an established algorithm that allows to analyze Wikipedia within just a few hours. As a special bonus, this topic model is also language-independent, since we can create the model in all supported Wikipedia languages. However, since the category system is created by the social tagging of the users, the categories can also be unclean, requiring cleanup first. For this purpose we have pursued various approaches, whereby we offer thematical analysis in different degrees of detail. In addition, we divided the Wikipedia category system into 3 sections: *space*, *time* and *topic*. This gives us a multimodal topic model that analyzes a text in terms of its location, the time period it is about and the theme. To test the potential of text2wiki in an application, we used it on the training and test data of a closed topic model and added additional categories. Here we have shown that the attached categories increase the classification results by up to 3%. Last but not least, we also analyzed how dynamic the

Wikipedia category system is and how it changes over time.

12.3 Related Work

There are already numerous studies dealing with the analysis of topics. Probably the best known topic model is LDA, an unsupervised method that assigns topic probabilities to a text. The number of topics to be found must be specified in advance and these will not be labeled afterwards. Thus, the result of this approach is difficult to interpret and cannot be used for various applications.

There are also approaches that have dealt with labeled topic models. Mehler and Waltinger (2009), Uslu, Mehler, and Baumartz (2019), Waltinger et al. (2011), and Brück, Eger, and Mehler (2016) have been working on a topic classifier based on the thematic universe of the Dewey Decimal Classification. J. Raiman and O. Raiman (2018) and Medeiros et al. (2018) worked on a topic model based on Wikipedia's category system. However, they used a predefined and very small part of the categories to get a basic overview of the topic distribution.

The thematic universe that the above approaches have dealt with is defined in advance and does not adapt to today's world. Mehler and Waltinger (2009) introduce the term *open topic model* (OTM), which describes a thematic universe that is constantly adapted and expanded by social tagging, as it is the case with the Wikipedia category system. There are also several research approaches that deal with the category system of Wikipedia. Schönhofen (2009) for example use Wikipedia's category system to improve further classifications. However, he used a different approach by searching for Wikipedia article titles in documents and associated these texts with the corresponding Wikipedia categories. Subsequently, he uses these categories to solve classification tasks. Türker et al. (2019) calculates word embeddings by adding Wikipedia categories to the articles and thus incorporating additional thematic information into the embeddings. He also shows that he achieves better results in various classification tasks by using these embeddings. Syed, Finin, and Joshi (2008) use Wikipedia articles and the category and article link graphs to predict concepts for a document. He uses a search engine library that suggests the most

No. of categories	285,549
No. of articles	3,901,210
Avg. categories per article	4.466
No. of token	852,617,641
Size	5.1 GB

Table 12.1: Detailed information about the German Wikipedia (01. September 2018).

similar Wikipedia articles for a given text. From these articles and the associated categories, concepts are assigned to the input text. C.-Y. Chen and Ma (2018) examined the influence of Wikipedia categories on Chinese embeddings and the creation of a larger evaluation data set. Kittur, Chi, and Suh (2009) and Farina, Tasso, and Laniado (2011) map the Wikipedia category system to top level categories in order not to lose the clarity of the constantly growing Wikipedia. The category system is thereby reduced to 11 and 21 categories, resulting in a loss of semantic precision.

Many of these approaches have already taken first steps towards analysis and categorization according to the Wikipedia scheme. However, in most cases there were several shortcomings. Either the category system was downsized so much that the semantic information was greatly reduced or all categories are worked with and assigned according to certain articles, thus preventing individual topic distribution. To solve these problems, we create an artificial neural network using a filtered Wikipedia category system and thus are able to predict thousands of categories.

12.4 Model

12.4.1 Corpus

In this section we describe the approach to create the underlying training corpus of text2wiki.

12.4.2 Category tree creation

We use the Wikipedia category system to create a category tree. Here we used the dumps from September 01, 2018. Table 12.1 lists detailed information about the number of given articles and categories of this dump. From these we have used the records with the namespace 14[1] to get all category description pages. We have linked these categories to the parent and child categories, resulting in a cyclical directed category graph. By means of the width search we resolved these circles and created an acyclic category tree. We have assigned a *height* to each category that corresponds to the shortest path to the root category. Then we collected all Wikipedia articles that belong to these categories and calculated the number of articles contained by each category and its subtree. For example, to identify the articles contained in the category *Mathematics*, the union of all articles assigned as category *Mathematics* or one of its subcategories, such as *Arithmetic*, is used. Thus a category contains all direct articles and all articles assigned to the subcategories. In addition, for the subsequent filtering of the category system, we have calculated the number of all subcategories of a category.

Category selection

Since the Wikipedia category system is created by social tagging of its users, the degree of precision of the categories is not predefined. The category system contains a total of 285,549 categories, whereby categories exist that are rarely used. For this reason, we filtered the category system using heuristic methods in order to have enough data records for each category and to filter categories that were too specific or not relevant. For this we calculated the average number of articles, the average number of subcategories and the average height of a category. All categories that have surpassed this average have been included in our filtered category system, resulting in 3,806 categories.

Multi modal categories

In addition, we subdivided the Wikipedia category system into 3 areas.

[1] https://en.wikipedia.org/wiki/Wikipedia:Namespace

- *Space*
- *Time*
- *Theme*

This results in 3 models that predict the place, time period and theme of a particular text. The temporal analysis is based on millennia, centuries and decades. In the spatial analysis, first it is distinguished whether the content of the text is terrestrial or extraterrestrial. In the case of terrestrial, further predictions are made at both continental and country levels.

Article Selection

When creating the datasets, the articles were labelled with the direct categories and additionally with all categories that were above these categories in our filtered category system. All articles were pre-processed using TextImager (Hemati, Uslu, and Mehler 2016), providing additional information to the texts, such as lemma and part of speech information. This results in a data set of 2,300,000 articles with an average of 39 categories.

Expand Category Selection

In order not to be limited to the 3,806 filtered categories, we have developed a way to adjust this number individually. We have expanded the categories by adding the subcategories of the leaves of our category tree. In order not to increase the number of categories enormously, we have implemented the following steps:

- 3,806 categories
- 7,500 categories
- 10,000 categories
- 15,000 categories
- 20,000 categories

Table 12.2 shows detailed information on the data sets for each size.

	3,806	7,500	10,000	15,000	20,000
No. of lines	1,757,799	1,757,799	1,757,799	1,757,799	1,757,799
Size	7.0 GB	8.6 GB	11.0 GB	12.0 GB	13.9 GB
Avg. no. of categories	39.65	136.67	179.27	253.81	317.32

Table 12.2: Detailed information about the data sets.

12.4.3 Model architecture

text2wiki is based on an artificial neural network using only one hidden layer. We relied on the model architecture of text2ddc (Uslu, Mehler, and Baumartz 2019), as it was designed for an efficient and precise classification. The core idea of the classifier comes from fastText (Joulin et al. 2016) and offers more features as input after a few modifications. This was necessary as we had different input formats available, such as a DDC vector.

12.5 Experiments

In this chapter, we describe the conducted experiments and document the resulting outcomes.

12.5.1 text2wiki

For the development of text2wiki we trained the classifier of Chapter 12.4.1 on the corpus of Chapter 12.4. In the process, we tested various input types to optimize the classification. By using the *softmax* function in the output layer we get a probability distribution across the categories. Since we are dealing with multi-label classification, we have used various evaluation metrics for analysis. First of all, we looked at the precision when selecting the top n categories. Figure 12.1 shows the course of the precision depending on the number of categories selected. We have taken into account no more than the average number of categories of an article. In the example of the model of size 3,806 we have compared the precision starting from 1 selected category up to 39 categories. For the model of size 20,000 we took up to 317 categories.

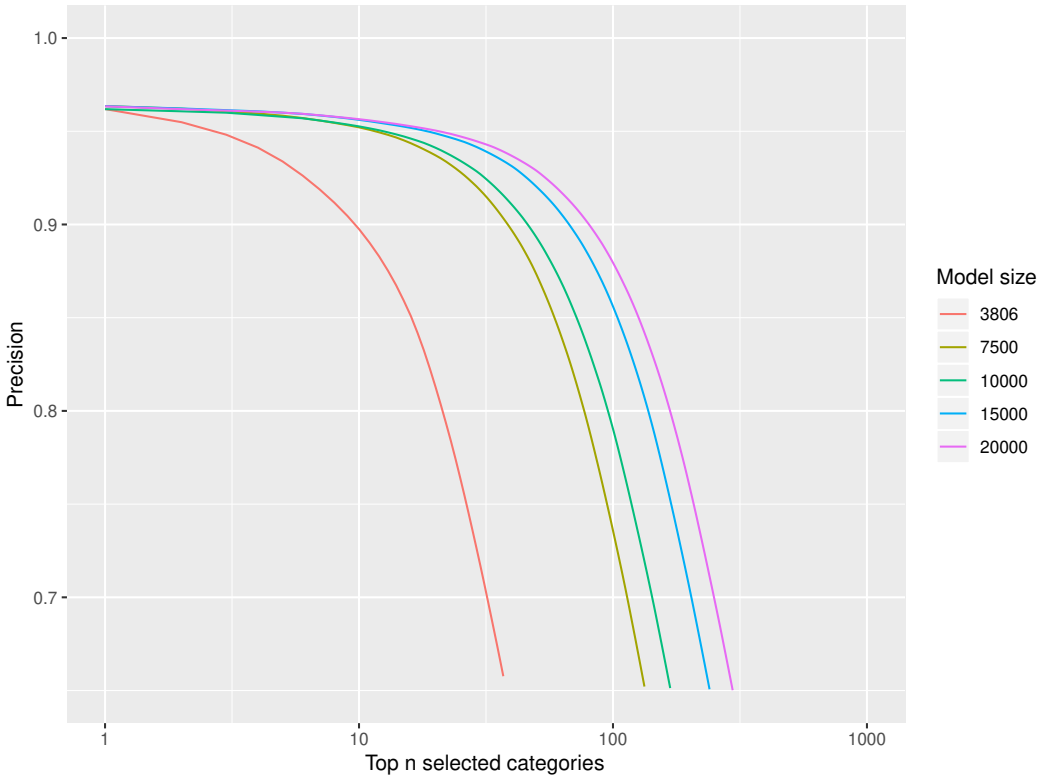


Figure 12.1: Comparison of the precision course according to the number of selected categories.

Table 12.3 lists some of these values and in most cases we reach an F-score above 90%, when considering the top 10 predicted categories.

Position	3,806	7,500	10,000	15,000	20,000
1	0.961	0.963	0.961	0.963	0.963
5	0.933	0.958	0.957	0.959	0.959
10	0.897	0.952	0.952	0.956	0.956
100		0.735	0.790	0.856	0.879
200				0.704	0.759
Avg. no. of categories	0.641	0.645	0.631	0.634	0.629

Table 12.3: Comparison of the precision by selecting the top n categories for each size.

Next, we examine the precision when selecting only categories that surpass a certain probability (see Figure 12.2). Here we distinguish between weighted and unweighted precision. In the case of weighted precision, not only the occurrence but also the probability of the category is taken into account. Taking

probabilities into account improves the precision, leading to the assumption that the more probable the class, the more likely it is to be classified correctly.

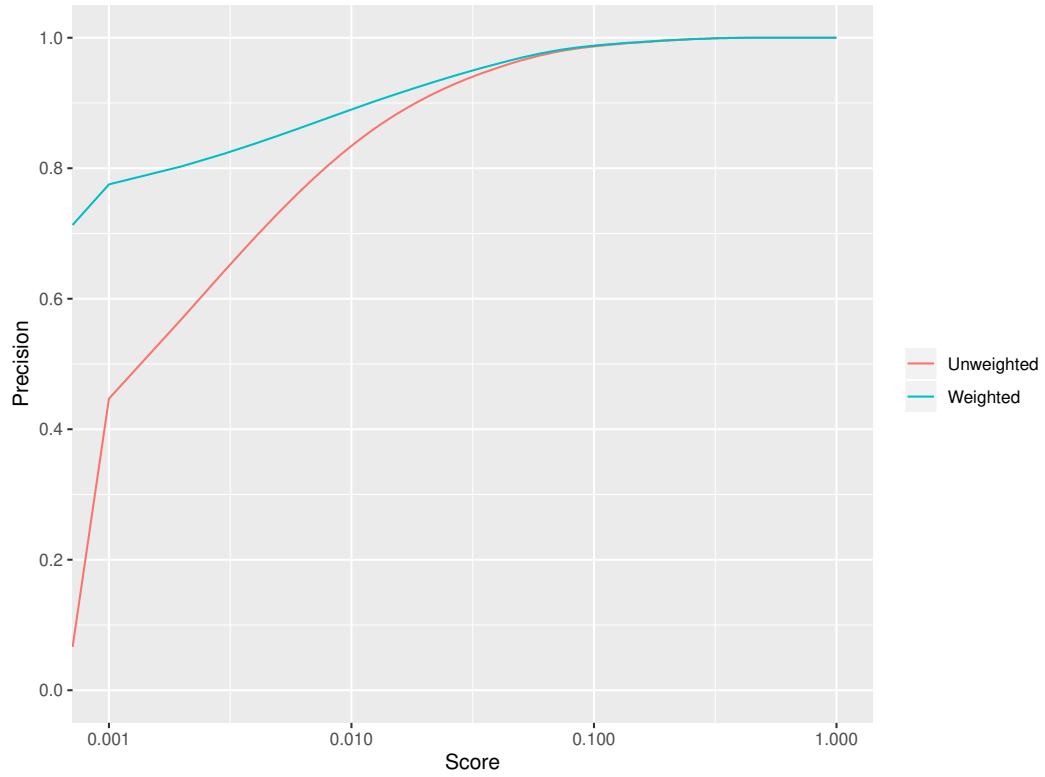


Figure 12.2: Weighted and unweighted precision when considering categories beyond a specific probability.

Last but not least, we examined all categories predicted with a probability greater than 0. Table 12.4 documents the results we achieved using the weighted evaluation method for different input parameters. Starting with 5 epochs, we analyzed the results using TextImagers pre-processing. Using the best input assignment, including word embeddings, lemma and POS information, we increased the number of epochs to 100. That alone has already improved the score a lot. By including DDC information using text2ddc and disambiguating the text with fastSense, we achieved our best result of 81.8%.

Using this evaluation method, we have also calculated the precision for the space and time categories. In Table 12.5 we see the results of all 3 models compared to each other.

No.	Features	Epoch	F1 score
1	Token	5	0.539
2	Lemma	5	0.556
3	2 + Embeddings	5	0.673
4	3 + POS	5	0.681
5	4	100	0.805
6	5 + text2ddc	100	0.812
7	13 + fastSense	100	0.818

Table 12.4: Weighted F-score for different inputs.

	Topic	Space	Time
F-score	0.818	0.952	0.853
Precision	0.719	0.917	0.766
Recall	0.998	0.999	0.999

Table 12.5: Results of text2wiki in topic, space and time analysis.

Subsequently, we applied the model to the different category sizes. Despite the increase of categories, we only had to sacrifice small losses in precision (see Table 12.6). Using this evaluation method we often predict many categories, resulting in a very high recall value. The number of mispredicted categories therefore increases as well. However, considering the probabilities of the categories, we achieve F-scores of almost 80%.

	3,806	7,500	10,000	15,000	20,000
F1-score	0.818	0.801	0.773	0.784	0.777
Weighted precision	0.719	0.695	0.660	0.674	0.666
Recall	0.998	0.996	0.995	0.994	0.992
Avg. no. of categories predicted	849	1,569	2,115	2,482	2,928

Table 12.6: Results of text2wiki for the different sizes.

Thanks to the multilingualism of Wikipedia, it is possible to develop language-independent text2wiki models for all supported languages. To exemplify this, we created models for 2 additional languages – English and French. In Table 12.7 we see the results of these models compared to the German version.

	DE	EN	FR
F1-score	0.818	0.750	0.718
Weighted precision	0.719	0.628	0.591
Recall	0.998	0.997	0.997

Table 12.7: Results of text2wiki for the languages German, English and French.

12.5.2 Use case scenario of text2wiki

After developing and optimizing text2wiki and providing it for different languages, we wanted to test the potential of the topic model. We have used text2wiki to enrich documents with additional thematic information and improve classification on this data. For this we used the data sets of text2ddc (Uslu, Mehler, and Baumartz 2019) and Google search snippets (Phan, L.-M. Nguyen, and Horiguchi 2008), which are based on plain text. The basic model of text2ddc has reached an F-score of 80.9% By adding the original Wikipedia categories of the articles, text2ddc could be improved to 82.1%. By incorporating the output categories of text2wiki (of size 3,806) we even reached 83.5%. In other words, not only do we get a good reproduction of the Wikipedia categories, we also improve the classification further using our filtered category system. In the example of the Google snippet classification, we achieved an improvement of more than 4% (see Table 12.8).

Features	**F-score**
text2ddc (basic)	0.808
text2ddc + Wikipedia categories	0.821
text2ddc + text2wiki categories	0.835
Google snippets (basic)	0.67
Google snippets + text2wiki categories	0.714

Table 12.8: Comparison of text2wiki categories with the original Wikipedia categories within text2ddc.

12.5.3 Dynamic category system

Since the Wikipedia category system is an open topic model, we wanted to measure the changes in the category system over time. Table 12.9 shows some

information from different periods of time of the German Wikipedia. The number of articles and categories is growing steadily. The category system filtered according to Chapter 12.4.2 is also constantly changing. Within one year, the number of categories increased by 237 from 3,806 to 4,043 categories. Thereby 122 categories have been removed and 359 new categories have been added, resulting in a total change of 481 categories. This means a rate of change of over 12% in one year.

Dump	01.09.18	01.07.19	01.08.19	01.09.19	01.10.19
No. of articles	2,326,425	2,435,499	2,446,630	2,457,130	2,468,349
No. of categories	285,549	313,127	316,221	319,043	321,522
Avg. articles per category	788.24	765.51	762.44	758.67	757.637
No. of topic categories	3,806	3,990	4,022	4,043	4,075
No. of space categories	208	208	208	208	208
No. of time categories	360	360	360	360	360

Table 12.9: Growth of Wikipedia and our filtered category system over time.

12.6 Discussion

We successfully used Wikipedia's category system as a basis to create text2wiki, an open topic model capable of handling big data. We optimized text2wiki to an F-score of over 80%, by using different pre-processing methods and features. We did this on a data set created by using various heuristics on the Wikipedia category tree. This allowed us to reduce all the 285,549 Wikipedia categories down to 3,806. In addition, we divided the Wikipedia category system into 3 sections: time, space and topic. Thus, text2wiki offers a semantic analysis beyond topic modeling. We also created models for more detailed category coverage with 7,500, 10,000, 15,000 and 20,000 categories. What we observed is that when we consider the average number of categories in each level, we obtain very similar precision values (see Figure 12.1 and Table 12.2). On the one hand, this means that the larger the classification system, the more the number of proposed categories, thus the more detailed the semantic information we obtain. On the other hand, this means that we hardly have to suffer any losses in terms of F-score.

We have also shown that the higher the probability of a predicted category, the higher the F-score. That means, if we only consider examples where text2wiki has assigned a probability of over 0.03 to a category, we even get an F-score of over 90% (see Figure 12.2). This can be considered when text2wiki maps a probability distribution over the categories to a text. This probability topic vector can then be used to address other tasks. In our case we improved text2ddc, a closed topic model based on the Dewey Decimal Classification system. We also did a classification using Google Search Snippet data and showed that using text2wiki leads to improvements. This demonstrates the potential of text2wiki and possible improvement options. Methods that rely on the categories of Wikipedia articles could also be applied to out-domain articles by using text2wiki to create the required categories. For example, as noted in Chapter 12.3, there are some methods that calculate word embeddings using Wikipedia categories. By using text2wiki, these approaches would no longer be limited to Wikipedia articles, but could be applied to any corpora.

Last but not least, we trained text2wiki in English and French to show its language independence. The classification quality in these models were not as good as for German. However, this seems to be caused by the fact that in the case of German we carried out many optimizations and pre-processing steps.

12.7 Future work

We introduce a neural network based open topic model called text2wiki. text2wiki was trained on a filtered version of the category system of the Wikipedia. This allows thousands of topics to be captured, whereby this number is constantly evolving by the tagging of the Wikipedia users. By using several features and pre-processing methods, we managed to achieve an F-score of over 80%. The classifier-based topic vector provided by text2wiki also improved other classification tasks. Due to its language independence, the potential of text2wiki is even broader. In future work we want to develop models for all supported Wikipedia languages. Subsequently, we will use these models to investigate thematic trends in different application areas. In addition, text2wiki can also be used to improve many other NLP tasks. The developed models will be

available open-source.

13 Summary

This chapter summarizes the goals and results of the dissertation and gives an outlook for future work.

13.1 Contribution

Many methods and tools have been presented in this dissertation addressing the main goal of automatic document analysis at semantic level. However, in order to achieve the main goal, we first had to develop a solid base to complete the big picture. Thus various methods and tools have been developed, covering different aspects of NLP. The interaction of these methods made it possible to achieve our goal successfully. Besides the automatic document analysis, we put great emphasis on the three principles (see Chapter 1.2.1) of *efficiency*, *applicability* and *language independence*. Thus all developments of this dissertation were ready for application. The size and language of the data to be analyzed is no longer an obstacle, at least for the supported Wikipedia languages.

A great contribution to this was `TextImager` (see Chapter 2), the tool responsible for the underlying architecture of various methods and the entire pre-processing of the texts. TextImager is designed as a multi-server and multi-instance cluster, enabling distributed processing of data, by using UIMAs cluster management services UIMA-AS[1] and UIMA-DUCC[2]. Furthermore the multi-service architecture of TextImager allows the integration of any NLP tool and their joint execution in a pipeline system. In addition, TextImager provides a web-based user interface that offers a range of interactive visual-

[1] uima.apache.org/doc-uimaas-what.html
[2] uima.apache.org/doc-uimaducc-whatitam.html

izations, depicting the results of the text analysis. The web interface does not require any programming expertise – by simply selecting the NLP components and entering the text, the analysis is started and subsequently visualized, allowing even technophiles to work with these tools.

In Chapter 3 we demonstrated the integration of the statistical framework `R` into the functionality and architecture of TextImager. Here we used the OpenCPU API to deploy R packages on our own R server. This allowed the combination of R packages with the state-of-the-art NLP components of TextImager. Thus the functions of the R packages received extracted information from TextImager, resulting in improved analyses. In addition, we added interactive visualizations to visualize the information derived from R.

Some of the visualizations developed within TextImager are particularly outstanding and have found application in many areas. An example of this is `PolyViz`, an interactive visualization system that allows the representation of a multi-part graph. In Chapter 4 we have exemplified PolyViz by means of two different use cases. In addition, the visualization was further developed allowing the individual groups to be clustered (see Figure 13.1).

Figure 13.2 shows the visual effects when interacting with a particular cluster. All associated nodes and edges inside and outside the groups are highlighted accordingly.

`SemioGraph`, a visualization technique for depicting multicodal graphs, was introduced in Chapter 5. The visual and interactive functions of SemioGraph were presented with an application for visualizing word embeddings. We showed that different embedding models can lead to completely different graphs. Thus Semiograph can help in finding word embeddings for specific NLP tasks.

Inspired by all the text visualizations within TextImager, the idea was born for `text2voronoi` (see Chapter 6). Here we presented a novel approach to image-driven text classification based on a Voronoi tesselation of linguistic features. This classification approach has been applied to automatic patient diagnosis and we have shown that we even exceed the traditional bag-of-words model. This approach makes it possible to subsequently analyze the underlying features, thus taking a first step towards resolving the black box. In Chapter

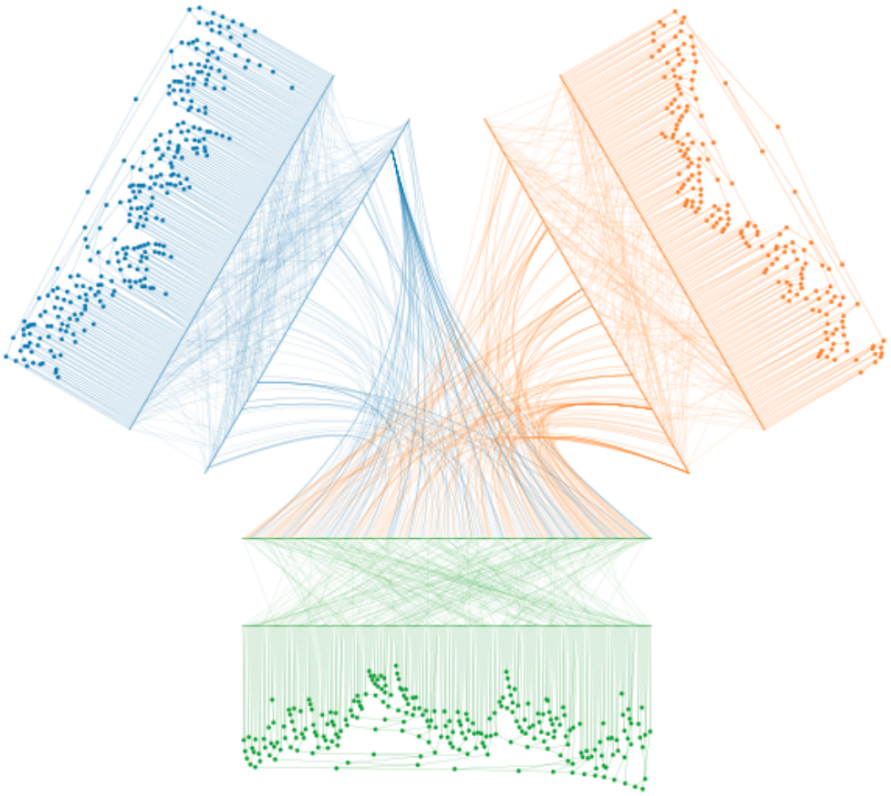
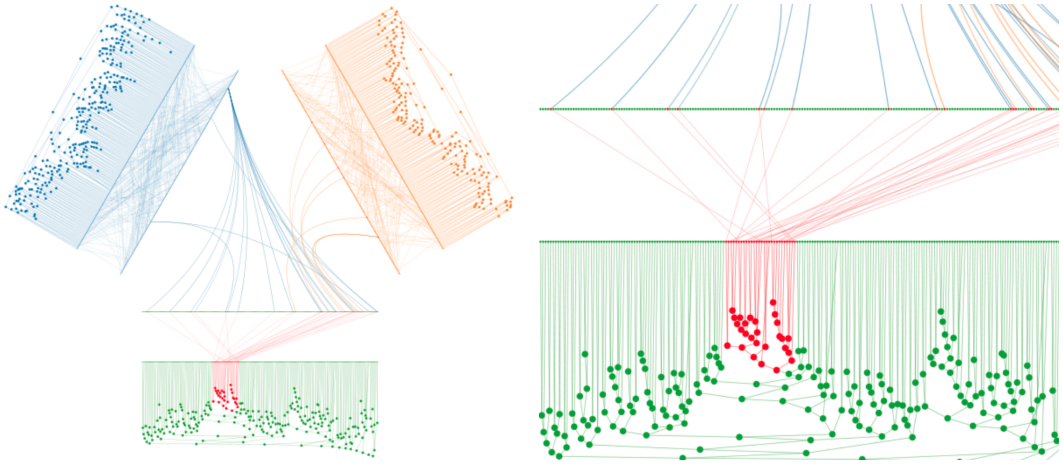


Figure 13.1: Further development of PolyViz including cluster analysis.



(a) Overview (b) Zoomed in

Figure 13.2: Visual highlighting when interacting with PolyViz.

8, we conducted further analyses on the automatic classification of memory clinic patients and depressed patients.

We applied text2voronoi to literary works in Chapter 7, resulting in the web

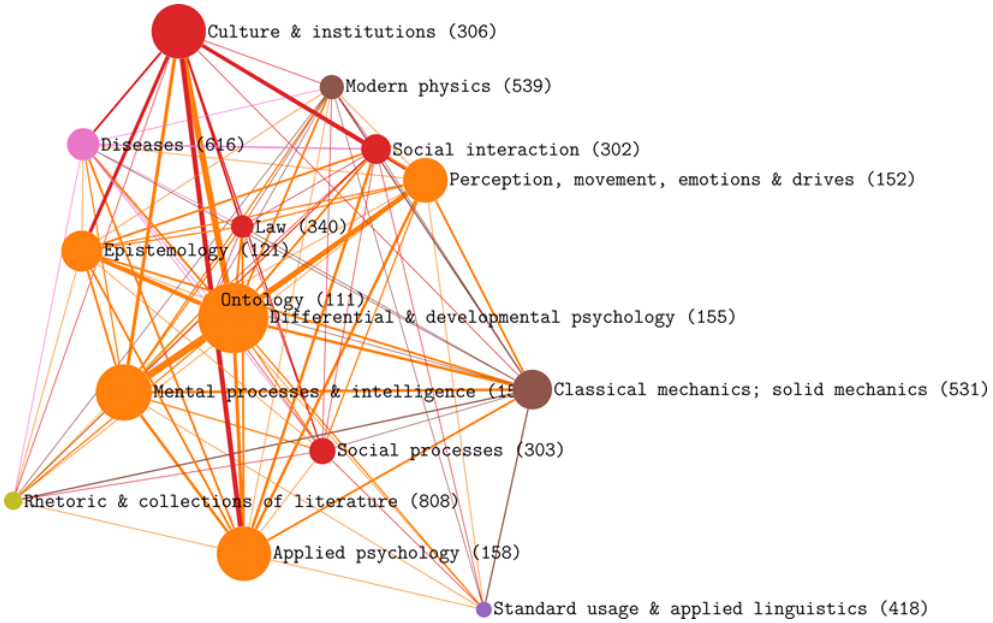
tool called `LitViz`. Here we enabled the comparison of Voronoi diagrams for different literatures, and thus the visual comparison of the linguistic styles of the underlying authors.

Now that we are proficient in the pre-processing and analysis of texts, we have come one step closer to our goal of semantic document analysis. Next, we have examined the dissolution of senses at the word level in Chapter 9. Here we presented `fastSense`, a word sense disambiguation tool designed for big data. In order to accomplish this, we created a disambiguation corpus based on Wikipedia's 221,965 disambiguation pages related to 825,179 senses. This resulted in more than 50 million data records, which required almost 50 GB of storage space. Not only have we shown that fastSense can handle such a large amount of data without any problems, but we have also shown that we can keep up with our competitors and even outperform them in shared tasks.

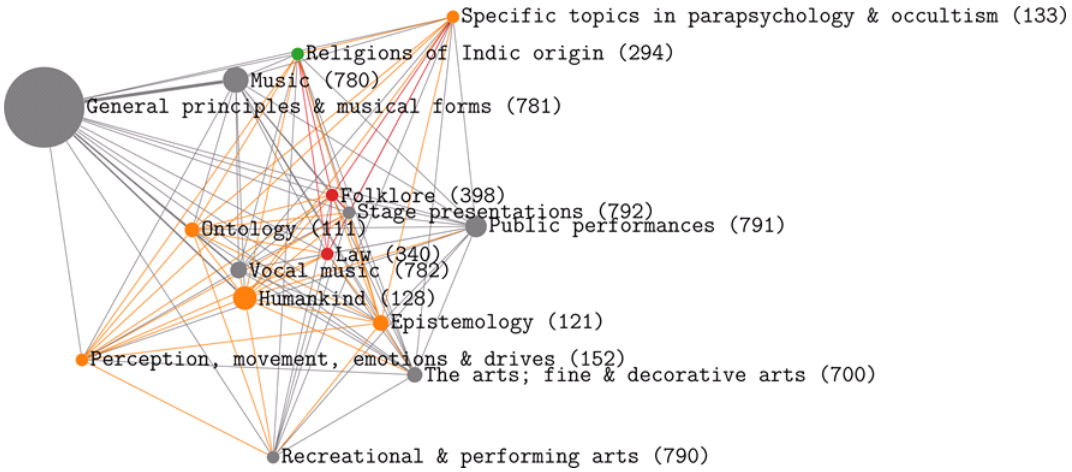
Now that we can assign senses to words, we have come another step closer to semantic document analysis. The more information we can obtain from a text and its words, the more precisely we can analyze its content, as confirmed in chapters 10 to 12. Starting with Chapter 10, we presented a network-theoretical approach to modeling the semantics of large text networks exemplified by the German Wikipedia. We showed how to assess the structuring of topics that are addressed in large corpora of natural language texts. To this end, we have developed an algorithm called `text2ddc` for modeling the thematic structure of a text, based on an established topic classification, namely the `Dewey Decimal Classification`. Using this model, we have shown how to create a bird's eye view of the salience and linkage of topics as manifested by millions of documents. Thus we have created a possibility to visualize the thematic dynamics of document networks automatically. However, the training and test data we had in this chapter consisted mainly of short text snippets. In Chapter 11 we created DDC corpora by examining information from Wikidata, Wikipedia and the Integrated Authority File (Gemeinsame Normdatei - GND). In this way, we were not only able to increase the amount of data, but also create data sets for many previously inaccessible languages. We also optimized text2ddc to achieve a score of 87.4% for the 98 classes of the second DDC level. The pre-processing of TextImager and the disambiguation by fastSense had a big impact on this. For any given document, text2ddc is able to calculate a probability distribution across the DDC classes, resulting

in a labeled topic model, based on an internationally established classification system.

text2ddc was also used to address other research questions. For example, Mehler, Uslu, Rüdiger Gleim, et al. (2019) use text2ddc to analyze literary works by well-known authors according to their thematic content. Figure 13.3 shows a comparison between the authors Sigmund Freud and Friedrich Nietzsche.



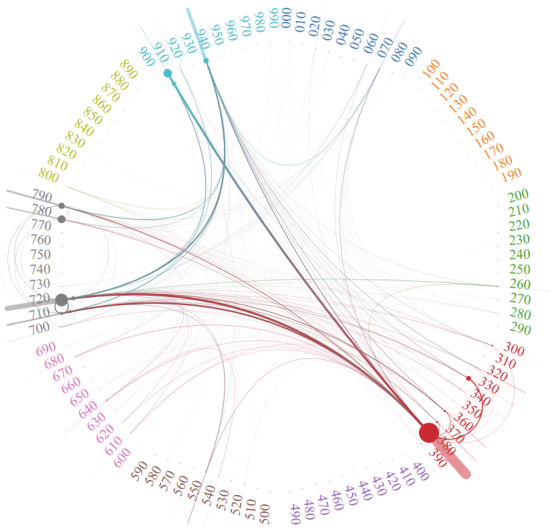
(a) Sigmund Freud



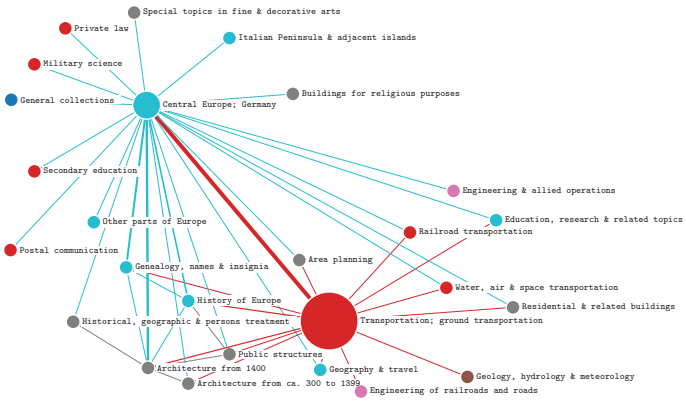
(b) Friedrich Nietzsche

Figure 13.3: Comparison of the thematic composition of two authors (Mehler, Uslu, Rüdiger Gleim, et al. 2019).

Mehler, Rüdiger Gleim, Gaitsch, et al. (2019) use text2ddc to explore city wikis. Here they analyzed which topics were discussed in different cities and how they relate to each other. Figure 13.4 shows the thematic progression of Munich and Dresden, whereby in the case of Munich, PolyViz was used to illustrate the distribution.



(a) Munich



(b) Dresden

Figure 13.4: Thematic progression in city wikis (Mehler, Rüdiger Gleim, Gaitsch, et al. 2019).

The classifier-induced semantic space of text2ddc was also used to improve further NLP methods. This also includes `text2wiki`, a framework for automatic tagging according to the Wikipedia category system. Again we have a classifier-induced semantic space, but this time it is based on the Wikipedia

category system. A big advantage of this model is the precision and depth of the covered themes and the constantly evolving category system. Thus the criteria of an open topic model have also been fulfilled. To demonstrate the benefits of text2wiki, we subsequently used the topic vectors provided by text2wiki in order to improve text2ddc, allowing both systems to improve each other. The synergy between the created tools in this dissertation was decisive for the success of each tool. Figure 13.5 shows a more detailed view of the interaction model from Section 1.3. For the sake of clarity, we have highlighted the focal points of this dissertation. Since the main focus of the dissertation is the semantic analysis of multiple documents, we highlighted the 3 tools fastSense, text2ddc and text2wiki. In addition, we have highlighted the outgoing edges of these tools. Here again, a directed edge indicates the usage of the source tool in the target tool. We have successfully developed the methods for semantic text analysis and then applied them to further methods. For example, fastSense has contributed to the improvement of text2ddc. Moreover, fastSense in combination with text2ddc contributed to the improvement of text2wiki. text2wiki in turn could be used to improve further classifications. Even the tools which are not highlighted in Figure 13.5 strongly contributed to this. TextImager always offered a strong underlying architecture and extracted information through pre-processing. Due to the visualizations, the data could always be analyzed and errors could be detected prematurely. By using SemioGraph, specific word embeddings could be found that are suitable for the respective task. Ultimately, the semantic models also supported each other and led to improvements.

13.2 Future work

The contributions of this dissertation still offer room for improvements that can be addressed in future work. Firstly, the individual tools can be further optimized and secondly, these tools offer numerous opportunities for further research.

One could, for example, improve the topic model based on the DDC by obtaining more training data. In text2ddc we combined Wikipedia, Wikidata and GND to provide Wikipedia articles with DDC information. However,

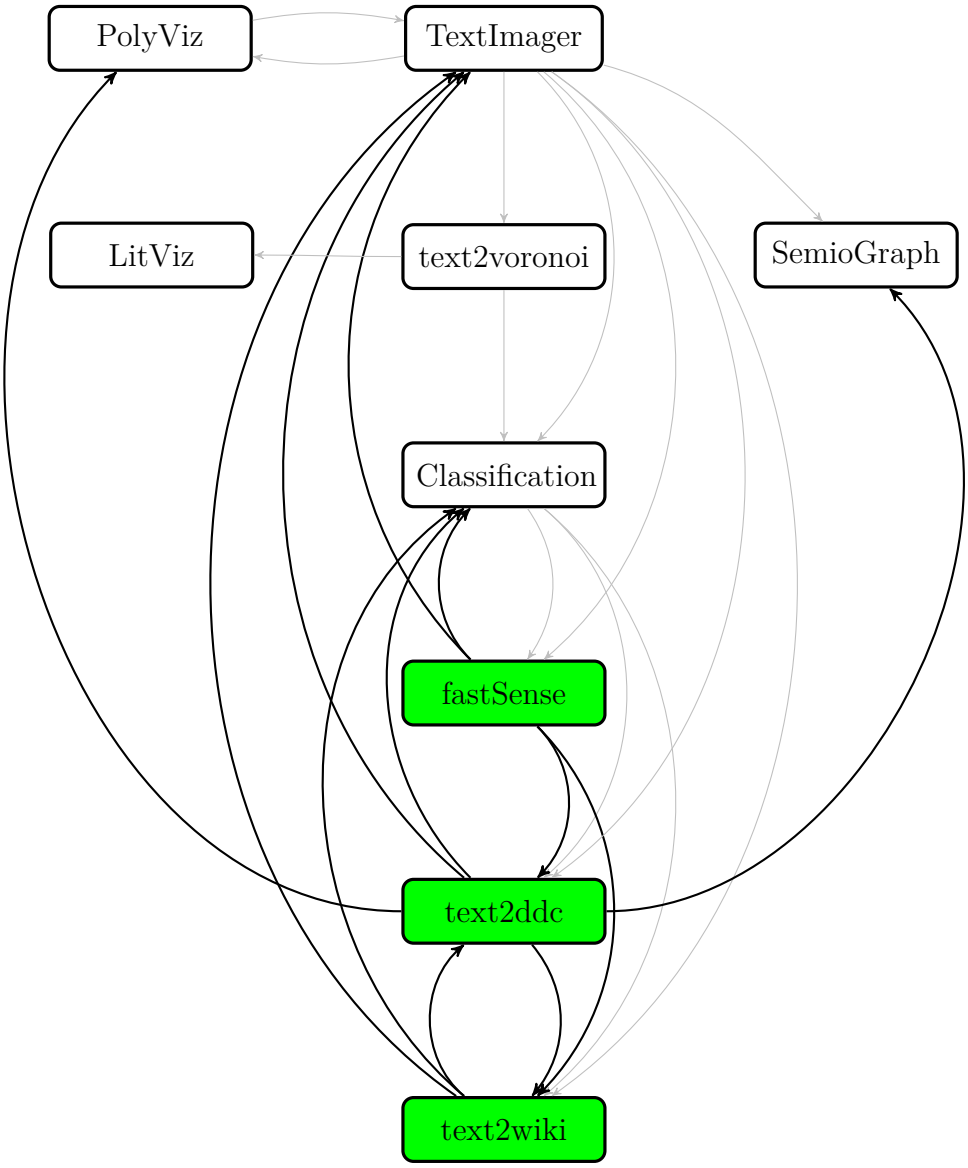


Figure 13.5: A more detailed view of the interaction model of Section 1.3, highlighting the emphasis of this dissertation.

here we were limited to the annotations from GND, leaving many Wikipedia articles without a DDC tag. One idea would be to map the DDC classes to Wikipedia categories and thus find new training data, since every Wikipedia article is provided with Wikipedia categories. Since Wikipedia categories are also available in other languages, this has the advantage of creating more data for low-resource languages. Another advantage would be to improve the 3rd level of DDC, as we could now cover all classes of this level and also generate much more data, as we would no longer be dependent on GND.

Another approach that could benefit from further work would be fastSense. In fastSense we determined ambiguous words based on the Wikipedia disambiguation pages and created a data set from the corresponding articles. We used Wikipedia as a basis as it offers a large data source and also a very detailed division of the senses. In some cases, however, this division is either too precise, which can lead to more than 40 meanings for a word, or too coarse, where certain meanings may be missing. This is the case since Wikipedia is a collaborative encyclopedia that allows users to divide words into their senses with any degree of precision. Here it might be helpful to incorporate resources created and maintained by experts, such as the Duden[3]. Another advantage of additional resources is not to be limited to nouns, as in the case of Wikipedia. A big challenge that has to be addressed here is the mapping of these resources to the senses and articles of Wikipedia. However, this would make it possible to improve and extend the underlying model of fastSense without losing the benefits of Wikipedia. Another feature fastSense could benefit from is the automatic recognition of multiword expressions. In the conducted experiments (Wikipedia and shared tasks) the words to be disambiguated are already specified. In practice, however, these words have to be found first. Here we are currently working with a simple search using regular expressions. Newer methods working with neural networks could be used to optimize this process.

In the recent past, transformer architectures (Vaswani et al. 2017) have stepped into the machine learning world. There are already numerous research articles available, achieving significant improvements by using transformer architectures (Radford et al. 2018; Tripathi et al. 2019; Prato, Charlaix, and Rezagholizadeh 2019; Zhao et al. 2019). The biggest drawback of this architecture is its slow performance, however, initial approaches to accelerating this architecture are already in progress (Winata et al. 2019; Murray et al. 2019; Gong et al. 2019). If one day the transformer architectures will be able to efficiently analyze the millions of documents we are dealing with in this dissertation, we could also apply them to our data sets and improve the models.

Besides the improvement of the developed tools, many new research possibilities for future work were created. The two developed topic models text2ddc and text2wiki are particularly responsible for this. By using our efficient closed

[3] https://www.duden.de/

and open topic models, it is possible to analyze different corpora according to their topic distribution, whereby the size of the corpora is no longer an issue. The CTM is particularly well suited for measuring and comparing the temporal progression of topics. For example, Wikipedia could be analyzed thematically at different time periods and the temporal progression could be examined. This could also be applied to any other platform or social media in order to identify certain trends. One could also analyze these temporal thematic shifts and forecast possible future trends.

Comparability is not that straightforward with text2wiki since the thematic universe is constantly evolving. A particular time could be selected as view point for the topics. Another approach could be to look at a certain text corpus from the topical viewpoints of different time periods. This would enable a kind of time travel from a thematic point of view.

Bibliography

Agerri, Rodrigo, Josu Bermudez, and German Rigau (2014). "IXA pipeline: Efficient and Ready to Use Multilingual NLP tools." In: *LREC*. Vol. 2014, pp. 3823–3828.

Albert, Marilyn S et al. (2011). "The diagnosis of mild cognitive impairment due to Alzheimer's disease: Recommendations from the National Institute on Aging-Alzheimer's Association workgroups on diagnostic guidelines for Alzheimer's disease". In: *Alzheimer's & dementia* 7.3, pp. 270–279.

Amariglio, Rebecca E. et al. (2012). "Subjective cognitive complaints and amyloid burden in cognitively normal older individuals". In: *Neuropsychologia* 50.12, pp. 2880–2886. ISSN: 0028-3932.

Asturiano, Vasco (2018). *force-graph – Force-directed graph rendered on HTML5 canvas*. `https://github.com/vasturiano/force-graph`. Accessed: 2018-11-22.

Balash, Y. et al. (2013). "Subjective memory complaints in elders: depression, anxiety, or cognitive decline?" In: *Acta Neurologica Scandinavica* 127.5, pp. 344–350. ISSN: 1600-0404.

Bär, Daniel, Chris Biemann, Iryna Gurevych, and Torsten Zesch (2012). "UKP: Computing Semantic Textual Similarity by Combining Multiple Content Similarity Measures". In: *Proc. of SemEval '12*. Stroudsburg, pp. 435–440.

Barrat, Alain, Marc Barthélemy, and Alessandro Vespignani (2007). "The Architecture of Complex Weighted Networks". In: *Large Scale Structure and Dynamics of Complex Networks*. Ed. by Guido Caldarelli and Allesandro Vespignani. New Jersey: World Scientific, pp. 67–92.

Battista, Petronilla, Christian Salvatore, and Isabella Castiglioni (2017). "Optimizing neuropsychological assessments for cognitive, behavioral, and functional impairment classification: A machine learning study". In: *Behavioural neurology* 2017.

Bede, Peter (2017). *From qualitative radiological cues to machine learning: MRI-based diagnosis in neurodegeneration*.

Beelen, Kaspar (2015). *Visualizing Parliamentary Discourse with Word2Vec and Gephi*. https://blog.history.ac.uk/2015/08/visualizing-parliamentary-discourse-with-word2vec-and-gephi. Accessed: 2018-11-22.

Benito-León, Julián, Alex J Mitchell, Saturio Vega, and Félix Bermejo-Pareja (2010). "A population-based study of cognitive function in older people with subjective memory complaints". In: *Journal of Alzheimer's Disease* 22.1, pp. 159–170.

Berg, Mark de, Marc van Kreveld, Mark Overmars, and Otfried Schwarzkopf (2000). *Computational Geometry*. Berlin/Heidelberg: Springer.

Biemann, Chris, Gregory R Crane, Christiane D Fellbaum, and Alexander Mehler (2014). "Computational humanities-bridging the gap between computer science and digital humanities (Dagstuhl Seminar 14301)". In: *Dagstuhl reports*. Vol. 4. 7. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik.

Bird, Steven, Loper Edward, and Ewan Klein (2009). *Natural Language Processing with Python*. O'Reilly Media Inc. URL: http://14.139.183.250:8080/bitstream/handle/123456789/120/Natural%5C%20Language%5C%20Processing%5C%20with%5C%20Python%5C%20%5C%282009%5C%29.pdf?sequence=1.

Blei, David M, Andrew Y Ng, and Michael I Jordan (2003). "Latent dirichlet allocation". In: *Journal of machine Learning research* 3.Jan, pp. 993–1022.

Bostock, Michael, Vadim Ogievetsky, and Jeffrey Heer (2011). "D^3 data-driven documents". In: *IEEE transactions on visualization and computer graphics* 17.12, pp. 2301–2309.

Brinker, Klaus et al. (1988). *Linguistische Textanalyse*. Dt. Blindenstudienanst.

Brück, Tim vor der, Steffen Eger, and Alexander Mehler (2016). "Complex Decomposition of the Negative Distance kernel". In: *CoRR* abs/1601.00925. URL: http://arxiv.org/abs/1601.00925.

Buckley, Rachel F. et al. (2015). "Phenomenological characterization of memory complaints in preclinical and prodromal Alzheimer's disease". In: *Neuropsychology* 29.4, p. 571. ISSN: 1931-1559.

Buckley, Rachel F et al. (2017). "Region-specific association of subjective cognitive decline with tauopathy independent of global β-amyloid burden". In: *JAMA neurology* 74.12, pp. 1455–1463.

Buckley, Rachel et al. (2013). "Factors affecting subjective memory complaints in the AIBL aging study: biomarkers, memory, affect, and age". In: *International Psychogeriatrics* 25.8, pp. 1307–1315.

Burghardt, Manuel, Julian Pörsch, Bianca Tirlea, and Christian Wolff (2014). "WebNLP: An Integrated Web-Interface for Python NLTK and Voyant". In: *Proceedings of the 12th edition of the KONVENS conference : Hildesheim, Germany, October 8-10,2014*. Ed. by Josef Ruppenhofer and Gertrud Faaß. Hildesheim: Universitätsbibliothek Hildesheim, pp. 235–240. URL: https://epub.uni-regensburg.de/35712/.

Cao, N. and W. Cui (2016). *Introduction to Text Visualization*. Atlantis Briefs in Artificial Intelligence. Atlantis Press. ISBN: 9789462391857. URL: https://books.google.de/books?id=6b0hjwEACAAJ.

Castilho, Richard Eckart de and Iryna Gurevych (2014). "A broad-coverage collection of portable NLP components for building shareable analysis pipelines". In: *Proceedings of the Workshop on Open Infrastructures and Analysis Frameworks for HLT*, pp. 1–11.

Cha, Youngchul and Junghoo Cho (2012). "Social-network analysis using topic models". In: *Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval*. ACM, pp. 565–574.

Chang, Jonathan (2015). *lda: Collapsed Gibbs Sampling Methods for Topic Models*. R package version 1.4.2. URL: https://CRAN.R-project.org/package=lda.

Chang, Jonathan and David Blei (2009). "Relational topic models for document networks". In: *Artificial Intelligence and Statistics*, pp. 81–88.

Chang, Jonathan, Sean Gerrish, Chong Wang, Jordan L Boyd-Graber, and David M Blei (2009). "Reading tea leaves: How humans interpret topic models". In: *Advances in neural information processing systems*, pp. 288–296.

Chaplot, Devendra Singh, Pushpak Bhattacharyya, and Ashwin Paranjape (2015). "Unsupervised Word Sense Disambiguation Using Markov Random Field and Dependency Parser." In: *AAAI*, pp. 2217–2223.

Chen, Chi-Yen and Wei-Yun Ma (2018). "Word embedding evaluation datasets and wikipedia title embedding for Chinese". In: *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*.

Chen, Po-Hao, Hanna Zafar, Maya Galperin-Aizenberg, and Tessa Cook (2018). "Integrating natural language processing and machine learning algorithms

to categorize oncologic response in radiology reports". In: *Journal of digital imaging* 31.2, pp. 178–184.

Chen, Mengen, Xiaoming Jin, and Dou Shen (2011). "Short Text Classification Improved by Learning Multi-granularity Topics". In: *Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence - Volume Volume Three*. IJCAI'11. Barcelona, Catalonia, Spain: AAAI Press, pp. 1776–1781.

Chetelat, Gael et al. (2010). "Relationship between atrophy and b–amyloid deposition in Alzheimer disease". In: *Annals of neurology* 67.3, pp. 317–324. ISSN: 1531-8249.

Chollet, François et al. (2015). *Keras*. `https://github.com/keras-team/keras`.

Corp, IBM (2013). "IBM SPSS statistics for windows, version 22.0". In: *Armonk, NY: IBM Corp*.

Csáji, Balázs Csanád (2001). "Approximation with artificial neural networks". In: *Faculty of Sciences, Etvs Lornd University, Hungary* 24, p. 48.

Csardi, Gabor and Tamas Nepusz (2006). "The igraph software package for complex network research". In: *InterJournal* Complex Systems, p. 1695. URL: `http://igraph.org`.

Cui, Licong, Samden D Lhatoo, Guo-Qiang Zhang, Satya Sanket Sahoo, and Alireza Bozorgi (2012). "EpiDEA: extracting structured epilepsy and seizure information from patient discharge summaries for cohort identification". In: *AMIA Annu Symp Proc.* Pp. 1191–1200.

Cunningham, Hamish et al. (2011). *Text Processing with GATE (Version 6)*. ISBN: 978-0956599315. URL: `http://tinyurl.com/gatebook`.

Deerwester, Scott, Susan T Dumais, George W Furnas, Thomas K Landauer, and Richard Harshman (1990). "Indexing by latent semantic analysis". In: *Journal of the American society for information science* 41.6, pp. 391–407.

DeVault, David, Ron Artstein, et al. (2014). "SimSensei Kiosk: A Virtual Human Interviewer for Healthcare Decision Support". In: *Proc. of the 2014 Int. Conf. on Autonomous Agents and Multi-agent Systems (AAMAS '14)*. Paris, France, pp. 1061–1068. URL: `http://dl.acm.org/citation.cfm?id=2617388.2617415`.

DeVault, David, Kallirroi Georgila, et al. (2013). "Verbal indicators of psychological distress in interactive dialogue with a virtual human". In: *Proc. of SIGDIAL*.

Dewey, Melvil (1876). *Decimal classification and relative index*.

Drew, Paul, John Chatwin, and Sarah Collins (2001). "Conversation analysis: a method for research into interactions between patients and health-care professionals". In: *Health Expectations* 4.1, pp. 58–70.

Eckart de Castilho, Richard and Iryna Gurevych (2014). "A broad-coverage collection of portable NLP components for building shareable analysis pipelines". In: *Proceedings of the Workshop on Open Infrastructures and Analysis Frameworks for HLT, OIAF4HLT@COLING 2014, Dublin, Ireland, August 23, 2014*, pp. 1–11. DOI: 10.3115/v1/W14-5201. URL: https://doi.org/10.3115/v1/W14-5201.

Eder, Maciej, Jan Rybicki, and Mike Kestemont (2016). "Stylometry with R: a package for computational text analysis". In: *R Journal* 8.1, pp. 107–121. URL: http://journal.r-project.org/archive/2016-1/eder-rybicki-kestemont.pdf.

Eger, Steffen, Rüdiger Gleim, and Alexander Mehler (2016). "Lemmatization and Morphological Tagging in German and Latin: A comparison and a survey of the state-of-the-art". In: *Proc. of LREC 2016*. Portorož (Slovenia).

Fan, Rong-En, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin (2008). "LIBLINEAR: A Library for Large Linear Classification". In: *Journal of Machine Learning Research* 9, pp. 1871–1874.

Farina, Jacopo, Riccardo Tasso, and David Laniado (2011). "Automatically assigning Wikipedia articles to macrocategories". In: *Proc. of Hypertext*.

Feinerer, Ingo and Kurt Hornik (2015). *tm: Text Mining Package*. R package version 0.6-2. URL: https://CRAN.R-project.org/package=tm.

Ferragina, Paolo and Ugo Scaiella (2010). "Tagme: on-the-fly annotation of short text fragments". In: *Proceedings of the 19th ACM international conference on Information and knowledge management*. ACM, pp. 1625–1628.

Ferrucci, David and Adam Lally (Sept. 2004). "UIMA: An Architectural Approach to Unstructured Information Processing in the Corporate Research Environment". In: *Natural Language Engineering* 10.3-4, pp. 327–348. DOI: 10.1017/S1351324904003523.

Folstein, Marshal F, Susan E Folstein, and Paul R McHugh (1975). "Mini-mental state: a practical method for grading the cognitive state of patients for the clinician". In: *Journal of psychiatric research* 12.3, pp. 189–198.

Forsythand, Eric N and Craig H Martell (2007). "Lexical and discourse analysis of online chat dialog". In: *Semantic Computing, 2007. ICSC 2007. International Conference on*. IEEE, pp. 19–26.

Friedman, Carol, Thomas C. Rindflesch, and Milton Corn (2013). "Natural language processing: state of the art and prospects for significant progress, a workshop sponsored by the National Library of Medicine". In: *Journal of Biomedical Informatics* 46.5, pp. 765–773.

Gaizauskas, Robert, Emma Barker, Monica Lestari Paramita, and Ahmet Aker (2014). "Assigning Terms to Domains by Document Classification". In: *Proceedings of the 4th International Workshop on Computational Terminology*, pp. 11–21.

Galavotti, Luigi, Fabrizio Sebastiani, and Maria Simi (2000). "Experiments on the use of feature selection and negative evidence in automated text categorization". In: *International Conference on Theory and Practice of Digital Libraries*. Springer, pp. 59–68.

Gardner, Matt et al. (2018). "AllenNLP: A Deep Semantic Natural Language Processing Platform". In: *CoRR* abs/1803.07640. arXiv: 1803.07640. URL: http://arxiv.org/abs/1803.07640.

Goldhahn, Dirk, Thomas Eckart, and Uwe Quasthoff (2012). "Building Large Monolingual Dictionaries at the Leipzig Corpora Collection: From 100 to 200 Languages." In: *LREC*. Vol. 29, pp. 31–43.

Golub, Koraljka, Johan Hagelbäck, and Anders Ardö (2018). "Automatic classification using DDC on the Swedish Union Catalogue". In: *18th European Networked Knowledge Organization Systems Workshop (NKOS 2018), Porto, Portugal, September 13, 2018*. CEUR-WS. org, pp. 4–16.

Gong, Linyuan et al. (2019). "Efficient Training of BERT by Progressively Stacking". In: *International Conference on Machine Learning*, pp. 2337–2346.

Goodfellow, Ian, Yoshua Bengio, and Aaron Courville (2016). *Deep learning*. MIT press.

Greimas, Algirdas Julien (1971). *Strukturale Semantik*. Springer.

Gretarsson, Brynjar et al. (2012). "Topicnets: Visual analysis of large text corpora with topic modeling". In: *ACM Transactions on Intelligent Systems and Technology (TIST)* 3.2, p. 23.

Griffiths, Thomas L and Mark Steyvers (2004). "Finding scientific topics". In: *Proceedings of the National academy of Sciences* 101.suppl 1, pp. 5228–5235.

Gülich, Elisabeth (2010). "Le rôle du corpus dans l'élaboration pluridisciplinaire d'un instrument de diagnostic linguistique: l'exemple de l'épilepsie". In: *Pratiques. Linguistique, littérature, didactique* 147-148, pp. 173–197.

Günther, Fritz, Carolin Dudschig, and Barbara Kaup (2015). "LSAfun: An R package for computations based on Latent Semantic Analysis". In: *Behavior Research Methods* 47.4, pp. 930–944. DOI: 10.3758/s13428-014-0529-0.

Guo, Weiwei and Mona Diab (2011). "Semantic Topic Models: Combining Word Distributional Statistics and Dictionary Definitions". In: *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. EMNLP '11. Edinburgh, United Kingdom: Association for Computational Linguistics, pp. 552–561. ISBN: 978-1-937284-11-4. URL: http://dl.acm.org/citation.cfm?id=2145432.2145496.

Gurevich, Pavel, Hannes Stuke, Andreas Kastrup, Heiner Stuke, and Helmut Hildebrandt (2017). "Neuropsychological testing and machine learning distinguish Alzheimer's disease from other causes for cognitive impairment". In: *Frontiers in aging neuroscience* 9.

Gyllensten, Amaru Cuba and Magnus Sahlgren (2015). "Navigating the semantic horizon using relative neighborhood graphs". In: *arXiv preprint arXiv:1501.02670*.

Helleputte, Thibault (2015). *LiblineaR: Linear Predictive Models Based on the LIBLINEAR C/C++ Library*. R package version 1.94-2.

Hemati, Wahed, Tolga Uslu, and Alexander Mehler (2016). "TextImager: a Distributed UIMA-based System for NLP". In: *Proceedings of the COLING 2016 System Demonstrations*. Osaka, Japan.

Hinrichs, Erhard W., Marie Hinrichs, and Thomas Zastrow (2010). "WebLicht: Web-Based LRT Services for German". In: *ACL 2010, Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, July 11-16, 2010, Uppsala, Sweden, System Demonstrations*, pp. 25–29. URL: https://www.aclweb.org/anthology/P10-4005/.

Hinrichs, Marie, Thomas Zastrow, and Erhard W. Hinrichs (2010). "WebLicht: Web-based LRT Services in a Distributed eScience Infrastructure". In: *Proceedings of the International Conference on Language Resources and Evaluation, LREC 2010, 17-23 May 2010, Valletta, Malta*. URL: http://www.lrec-conf.org/proceedings/lrec2010/summaries/270.html.

Hochreiter, Sepp and Jürgen Schmidhuber (1997). "Long short-term memory". In: *Neural computation* 9.8, pp. 1735–1780.

Hofmann, Markus and Ralf Klinkenberg (2013). *RapidMiner: Data mining use cases and business analytics applications*. CRC Press.

Hofmann, Thomas (1999). "Probabilistic latent semantic analysis". In: *Proceedings of the Fifteenth conference on Uncertainty in artificial intelligence*. Morgan Kaufmann Publishers Inc., pp. 289–296.

Honnibal, Matthew and Ines Montani (2017). "spacy 2: Natural language understanding with bloom embeddings, convolutional neural networks and incremental parsing". In: *To appear* 7.

Howes, Christine, Matt Purver, Rose McCabe, Patrick GT Healey, and Mary Lavelle (2012). "Helping the medicine go down: Repair and adherence in patient-clinician dialogues". In: *Proc. of the 16th SemDial Workshop on the Semantics and Pragmatics of Dialogue (SeineDial)*.

Howes, Christine, Matthew Purver, and Rose McCabe (2013). "Using conversation Topics for predicting Therapy Outcomes in schizophrenia". In: *Biomedical informatics insights* 6.Suppl 1, p. 39.

Howes, Christine, Matthew Purver, Rose McCabe, Patrick GT Healey, and Mary Lavelle (2012). "Predicting adherence to treatment for schizophrenia from dialogue transcripts". In: *Proceedings of the 13th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pp. 79–83.

Iacobacci, Ignacio, Mohammad Taher Pilehvar, and Roberto Navigli (2015). "Sensembed: Learning sense embeddings for word and relational similarity". In: *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. Vol. 1, pp. 95–105.

– (2016). "Embeddings for Word Sense Disambiguation: An Evaluation Study". In: *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 1: Long Papers*.

Ivnik, Robert J et al. (1997). "Free and cued selective reminding test: MOANS norms". In: *Journal of Clinical and Experimental Neuropsychology* 19.5, pp. 676–691.

Jäger, Gerhard (2006). "Convex meanings and evolutionary stability". In: *The Evolution of Language. Proc. of the 6th International Conference (EVOLANG6)*. Ed. by Angelo Cangelosi, Andrew D. M. Smith, and Kenny Smith. Rome, pp. 139–144.

Jänicke, Stefan, Greta Franzini, Muhammad Faisal Cheema, and Gerik Scheuermann (2015). "On Close and Distant Reading in Digital Humanities: A Survey and Future Challenges." In: *EuroVis (STARs)*, pp. 83–103.

Jessen, Frank et al. (2014). "A conceptual framework for research on subjective cognitive decline in preclinical Alzheimer's disease". In: *Alzheimer's & dementia* 10.6, pp. 844–852.

Joachims, Thorsten (1998). *Making large-scale SVM learning practical*. Tech. rep. Technical Report, SFB 475: Komplexitätsreduktion in Multivariaten Datenstrukturen, Universität Dortmund.

– (2002). *Learning to classify text using support vector machines*. Boston: Kluwer.

Jonker, Cees, Mirjam I Geerlings, and Ben Schmand (2000). "Are memory complaints predictive for dementia? A review of clinical and population-based studies". In: *International journal of geriatric psychiatry* 15.11, pp. 983–991.

Joulin, Armand, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov (2016). "Bag of Tricks for Efficient Text Classification". In: *CoRR* abs/1607.01759. URL: http://arxiv.org/abs/1607.01759.

Kalchbrenner, Nal, Edward Grefenstette, and Phil Blunsom (2014). "A convolutional neural network for modelling sentences". In: *arXiv preprint arXiv:1404.2188*.

Kallmeyer, Werner, Wolfgang Klein, Reinhard Meyer-Hermann, Klaus Netzer, and Hans-Jürgen Siebert (1974). "Lektürekolleg zur Textlinguistik. 2 Bde". In: *Frankfurt/M.*

Khalili, Ali, Sören Auer, and Axel-Cyrille Ngonga Ngomo (2014). "context–lightweight text analytics using linked data". In: *European Semantic Web Conference*. Springer, pp. 628–643.

Kim, Jonghoon, François Rousseau, and Michalis Vazirgiannis (2015). "Convolutional Sentence Kernel from Word Embeddings for Short Text Categorization." In: *EMNLP*, pp. 775–780.

Kim, Yoon (2014). "Convolutional neural networks for sentence classification". In: *arXiv preprint arXiv:1408.5882*.

Kingma, Diederik and Jimmy Ba (2014). "Adam: A method for stochastic optimization". In: *arXiv preprint arXiv:1412.6980*.

Kittur, Aniket, Ed H Chi, and Bongwon Suh (2009). "What's in Wikipedia?: mapping topics and conflict using socially annotated category structure". In: *Proceedings of the SIGCHI conference on human factors in computing systems*. ACM, pp. 1509–1512.

Komninos, Alexandros and Suresh Manandhar (2016). "Dependency based embeddings for sentence classification tasks". In: *Proceedings of the 2016 Con-*

ference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pp. 1490–1500.

Koppara, Alexander et al. (2015). "Cognitive performance before and after the onset of subjective cognitive decline in old age". In: *Alzheimers & Dementia: Diagnosis, Assessment & Disease Monitoring* 1.2, pp. 194–205.

Kroenke, Kurt, Robert L Spitzer, Janet BW Williams, and Bernd Löwe (2010). "The patient health questionnaire somatic, anxiety, and depressive symptom scales: a systematic review". In: *General hospital psychiatry* 32.4, pp. 345–359.

Kucher, Kostiantyn and Andreas Kerren (2015). "Text visualization techniques: Taxonomy, visual survey, and community insights". In: *Visualization Symposium (PacificVis), 2015 IEEE Pacific*. IEEE, pp. 117–121.

Lafferty, John D. and David M. Blei (2006). "Correlated topic models". In: *Advances in neural information processing systems*, pp. 147–154.

Lamba, Manika and Margam Madhusudhan (Aug. 2019). "Mapping of Topics in DESIDOC Journal of Library and Information Technology, India: A Study". In: *Scientometrics* 120.2, pp. 477–505. ISSN: 0138-9130. DOI: 10.1007/s11192-019-03137-5. URL: https://doi.org/10.1007/s11192-019-03137-5.

Landauer, Thomas K. and Susan T. Dumais (1997). "A Solution to Plato's Problem: The Latent Semantic Analysis Theory of Acquisition, Induction, and Representation of Knowledge". In: *Psychological Review* 104.2, pp. 211–240.

Leopold, Edda (2007). "Models of Semantic Spaces". In: *Aspects of Automatic Text Analysis*. Ed. by Alexander Mehler and Reinhard Köhler. Vol. 209. Studies in Fuzziness and Soft Computing. Berlin/Heidelberg: Springer, pp. 117–137.

Levy, Omer and Yoav Goldberg (2014). "Dependency-based word embeddings". In: *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. Vol. 2, pp. 302–308.

Li, Jiwei and Dan Jurafsky (2015). "Do multi-sense embeddings improve natural language understanding?" In: *arXiv preprint arXiv:1506.01070*.

Li, Li et al. (2005). "A robust hybrid between genetic algorithm and support vector machine for extracting an optimal feature gene subset". In: *Genomics* 85.1, pp. 16–23. ISSN: 0888-7543. DOI: https://doi.org/10.1016/j.ygeno.

2004.09.007. URL: http://www.sciencedirect.com/science/article/pii/S088875430400271X.

Li, Qi, Tianshi Li, and Baobao Chang (2016). "Learning Word Sense Embeddings from Word Sense Definitions". In: *Natural Language Understanding and Intelligent Applications*. Springer, pp. 224–235.

Li, Xin and Dan Roth (2002). "Learning question classifiers". In: *Proceedings of the 19th international conference on Computational linguistics-Volume 1*. Association for Computational Linguistics, pp. 1–7.

Li, Yang, Wen-Zhuo Song, and Bo Yang (2018). "Stochastic Variational Inference-Based Parallel and Online Supervised Topic Model for Large-Scale Text Processing". In: *Journal of Computer Science and Technology* 33.5, pp. 1007–1022.

Lin, Min, Qiang Chen, and Shuicheng Yan (2013). "Network in network". In: *arXiv preprint arXiv:1312.4400*.

Ling, Wang, Chris Dyer, Alan Black, and Isabel Trancoso (2015). "Two/Too Simple Adaptations of word2vec for Syntax Problems". In: *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Denver, Colorado: Association for Computational Linguistics.

Lingler, Jennifer Hagerty et al. (2006). "Making sense of mild cognitive impairment: a qualitative exploration of the patient's experience". In: *The Gerontologist* 46.6, pp. 791–800.

Lösch, Mathias, Ulli Waltinger, Wolfram Horstmann, and Alexander Mehler (2011). "Building a DDC-annotated Corpus from OAI Metadata". In: *Journal of Digital Information* 12.2.

Maaten, Laurens van der and Geoffrey Hinton (2008). "Visualizing Data using t-SNE". In: *Journal of Machine Learning Research* 9, pp. 2579–2605. URL: http://www.jmlr.org/papers/v9/vandermaaten08a.html.

Magatti, Davide, Silvia Calegari, Davide Ciucci, and Fabio Stella (2009). "Automatic labeling of topics". In: *Intelligent Systems Design and Applications, 2009. ISDA'09. Ninth International Conference on*. IEEE, pp. 1227–1232.

Manning, Christopher D. et al. (2014). "The Stanford CoreNLP Natural Language Processing Toolkit". In: *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, ACL 2014, June 22-27, 2014, Baltimore, MD, USA, System Demonstrations*, pp. 55–60. URL: https://www.aclweb.org/anthology/P14-5010/.

Mayring, Philipp (2014). "Qualitative content analysis: theoretical foundation, basic procedures and software solution". In:

Mcauliffe, Jon D and David M Blei (2008). "Supervised topic models". In: *Advances in neural information processing systems*, pp. 121–128.

McCallum, Andrew, Andres Corrada-Emmanuel, and Xuerui Wang (2005). "Topic and role discovery in social networks". In: *Computer Science Department Faculty Publication Series*, p. 3.

McKhann, Guy et al. (1984). "Clinical diagnosis of Alzheimer's disease Report of the NINCDS-ADRDA Work Group* under the auspices of Department of Health and Human Services Task Force on Alzheimer's Disease". In: *Neurology* 34.7, pp. 939–939.

Medeiros, Jerry Fernandes, Bernardo Pereira Nunes, Sean Wolfgand Matsui Siqueira, and Luiz André Portes Paes Leme (2018). "TagTheWeb: Using Wikipedia Categories to Automatically Categorize Resources on the Web". In: *European Semantic Web Conference*. Springer, pp. 153–157.

Mehler, Alexander (2011). "Social ontologies as generalized nearly acyclic directed graphs: A quantitative graph model of social tagging". In: *Towards an Information Theory of Complex Networks*. Springer, pp. 259–319.

Mehler, Alexander, Rüdiger Gleim, Tim vor der Brück, et al. (2016). "Wikidition: Automatic Lexiconization and Linkification of Text Corpora". In: *Information Technology*, pp. 70–79. DOI: http://dx.doi.org/10.1515/itit-2015-0035.

Mehler, Alexander, Rüdiger Gleim, Regina Gaitsch, Tolga Uslu, and Wahed Hemati (2019). "From Topic Networks to Distributed Cognitive Maps: Zipfian Topic Universes in the Area of Volunteered Geographic Information". In: *Complexity*: *Cognitive Network Science: A New Frontier*. accepted.

Mehler, Alexander, Rüdiger Gleim, Wahed Hemati, and Tolga Uslu (2017). "Skalenfreie online soziale Lexika am Beispiel von Wiktionary". In: *Proceedings of 53rd Annual Conference of the Institut für Deutsche Sprache (IDS), March 14-16, Mannheim, Germany*. Ed. by Stefan Engelberg, Henning Lobin, Kathrin Steyer, and Sascha Wolfer. Berlin: De Gruyter.

Mehler, Alexander, Rüdiger Gleim, Andy Lücking, Tolga Uslu, and Christian Stegbauer (2018). "On the Self-similarity of Wikipedia Talks: a Combined Discourse-analytical and Quantitative Approach." In: *Glottometrics* 40, pp. 1–45.

Mehler, Alexander, Tolga Uslu, Rüdiger Gleim, and Daniel Baumartz (2019). "text2ddc meets Literature - Ein Verfahren für die Analyse und Visualisierung thematischer Makrostrukturen". In: *Proceedings of the 6th Digital Humanities Conference in the German-speaking Countries, DHd 2019*. DHd 2019. Frankfurt, Germany.

Mehler, Alexander, Tolga Uslu, and Wahed Hemati (2016). "Text2Voronoi: An Image-driven Approach to Differential Diagnosis". In: *Proceedings of the 5^{th} Workshop on Vision and Language (VL'16) hosted by the 54^{th} Annual Meeting of the Association for Computational Linguistics (ACL), Berlin*.

Mehler, Alexander and Ulli Waltinger (2009). "Enhancing document modeling by means of open topic models: Crossing the frontier of classification schemes in digital libraries by example of the DDC". In: *Library Hi Tech* 27.4, pp. 520–539.

Mei, Qiaozhu, Xuehua Shen, and ChengXiang Zhai (2007). "Automatic Labeling of Multinomial Topic Models". In: *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD '07. San Jose, California, USA: ACM, pp. 490–499. ISBN: 978-1-59593-609-7. DOI: 10.1145/1281192.1281246. URL: http://doi.acm.org/10.1145/1281192.1281246.

Melamud, Oren, Jacob Goldberger, and Ido Dagan (2016). "context2vec: Learning Generic Context Embedding with Bidirectional LSTM." In: *CoNLL*, pp. 51–61.

Michel, Jean-Baptiste et al. (2011). "Quantitative analysis of culture using millions of digitized books". In: *Science* 331.6014, pp. 176–182.

Miebach, Lisa, Steffen Wolfsgruber, Ingo Frommann, Rachel Buckley, and Michael Wagner (2017). "Different Cognitive Complaint Profiles in Memory Clinic and Depressive Patients". In: *The American Journal of Geriatric Psychiatry*.

Miebach, Lisa, Steffen Wolfsgruber, Ingo Frommann, Klaus Fließbach, et al. (2018). "Cognitive Complaints in Memory Clinic Patients and in Depressive Patients: An Interpretative Phenomenological Analysis". In: *The Gerontologist*.

Mihalcea, Rada (2007). "Using Wikipedia for Automatic Word Sense Disambiguation." In: *HLT-NAACL*, pp. 196–203.

– (2016). *Rada Mihalcea*. http://web.eecs.umich.edu/~mihalcea/downloads.html. Accessed: 2016-12-05.

Mihalcea, Rada and Andras Csomai (2007). "Wikify!: linking documents to encyclopedic knowledge". In: *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*. ACM, pp. 233–242.

Mihalcea, Rada, Paul Tarau, and Elizabeth Figa (2004). "PageRank on semantic networks, with application to word sense disambiguation". In: *Proceedings of the 20th international conference on Computational Linguistics*. Association for Computational Linguistics, p. 1126.

Mikolov, Tomas, Kai Chen, Greg Corrado, and Jeffrey Dean (2013). "Efficient estimation of word representations in vector space". In: *arXiv preprint arXiv:1301.3781*.

Mikolov, Tomas, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean (2013). "Distributed Representations of Words and Phrases and their Compositionality". In: *Advances in Neural Information Processing Systems 26*, pp. 3111–3119. URL: http://papers.nips.cc/paper/5021-distributed-representations-of-words-and-phrases-and-their-compositionality.pdf.

Mimno, David (Apr. 2012). "Computational Historiography: Data Mining in a Century of Classics Journals". In: *J. Comput. Cult. Herit.* 5.1, 3:1–3:19. ISSN: 1556-4673. DOI: 10.1145/2160165.2160168. URL: http://doi.acm.org/10.1145/2160165.2160168.

Molinuevo, José L. et al. (2016). "Implementation of subjective cognitive decline criteria in research studies". In: *Alzheimer's & Dementia*. ISSN: 1552-5260.

Moretti, Franco (2013). *Distant reading*. Verso Books.

Moro, Andrea, Alessandro Raganato, and Roberto Navigli (2014). "Entity linking meets word sense disambiguation: a unified approach". In: *Transactions of the Association for Computational Linguistics* 2, pp. 231–244.

Morris, J. C. et al. (1989). "The consortium to establish a registry for Alzheimer's disease (CERAD): I. Clinical and neuropsychological assessment of Alzheimer's disease". In: *Neurology*. ISSN: 1526-632X.

Müller, Thomas, Helmut Schmid, and Hinrich Schütze (2013). "Efficient Higher-Order CRFs for Morphological Tagging". In: *Proc. of the 2013 Conference on Empirical Methods in Natural Language Processing*, pp. 322–332. URL: http://www.aclweb.org/anthology/D13-1032.

Murray, Kenton, Jeffery Kinnison, Toan Q Nguyen, Walter Scheirer, and David Chiang (2019). "Auto-Sizing the Transformer Network: Improving Speed, Efficiency, and Performance for Low-Resource Machine Translation". In: *arXiv preprint arXiv:1910.06717*.

Newman, David J and Sharon Block (2006). "Probabilistic topic decomposition of an eighteenth-century American newspaper". In: *Journal of the American Society for Information Science and Technology* 57.6, pp. 753–767.

Nguyen, Minh Hoai and Fernando De la Torre (2010). "Optimal feature selection for support vector machines". In: *Pattern recognition* 43.3, pp. 584–591.

Ooms, Jeroen (2014). "The OpenCPU System: Towards a Universal Interface for Scientific Computing through Separation of Concerns". In: *arXiv:1406.4806 [stat.CO]*. URL: http://arxiv.org/abs/1406.4806.

Opp, Joachim, Barbara Job, and Heike Knerich (2015). "Linguistische Analyse von Anfallsschilderungen zur Unterscheidung epileptischer und dissoziativer Anfälle". In: *Neuropädiatrie in Klinik und Praxis* 14.1.

Organization, World Health (1993). *The ICD-10 classification of mental and behavioural disorders: diagnostic criteria for research*. Vol. 2. World Health Organization.

Oussous, Ahmed, Fatima-Zahra Benjelloun, Ayoub Ait Lahcen, and Samir Belfkih (2018). "Big Data technologies: A survey". In: *Journal of King Saud University-Computer and Information Sciences* 30.4, pp. 431–448.

Pedregosa, F. et al. (2011). "Scikit-learn: Machine Learning in Python". In: *Journal of Machine Learning Research* 12, pp. 2825–2830.

Pelevina, Maria, Nikolay Arefyev, Chris Biemann, and Alexander Panchenko (2017). "Making sense of word embeddings". In: *arXiv preprint arXiv:1708.03390*.

Peng, Nanyun and Mark Dredze (2016). "Learning Word Segmentation Representations to Improve Named Entity Recognition for Chinese Social Media". In: *CoRR* abs/1603.00786. arXiv: 1603.00786. URL: http://arxiv.org/abs/1603.00786.

Pennington, Jeffrey, Richard Socher, and Christopher Manning (2014). "Glove: Global vectors for word representation". In: *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pp. 1532–1543.

Phan, Xuan-Hieu, Le-Minh Nguyen, and Susumu Horiguchi (2008). "Learning to Classify Short and Sparse Text & Web with Hidden Topics from Large-

scale Data Collections". In: *Proceedings of the 17th International Conference on World Wide Web*. WWW '08. New York, NY, USA: ACM, pp. 91–100.

Pilehvar, Mohammad Taher and Roberto Navigli (2015). "From senses to texts: An all-in-one graph-based approach for measuring semantic similarity". In: *Artificial Intelligence* 228, pp. 95–128.

Plaza, Laura, Mark Stevenson, and Alberto Díaz (2010). "Improving summarization of biomedical documents using word sense disambiguation". In: *Proceedings of the 2010 Workshop on Biomedical Natural Language Processing*. Association for Computational Linguistics, pp. 55–63.

Plug, Leendert, Basil Sharrack, and Markus Reuber (2009). "Seizure metaphors differ in patients' accounts of epileptic and psychogenic nonepileptic seizures". In: *Epilepsia* 50.5, pp. 994–1000.

– (2010). "Seizure, Fit or Attack? The use of diagnostic labels by patients with Epileptic or Non-Epileptic Seizures". In: *Applied Linguistics* 31.1, pp. 94–114.

Prato, Gabriele, Ella Charlaix, and Mehdi Rezagholizadeh (2019). "Fully Quantized Transformer for Improved Translation". In: *arXiv preprint arXiv:1910.10485*.

Prince, M et al. (2015). "Alzheimer's Disease International (2015). World Alzheimer Report 2015: The Global Impact of Dementia: An Analysis of Prevalence, Incidence, Cost and Trends". In: *Alzheimer's Disease International, London*.

R Development Core Team (2008). *R: A Language and Environment for Statistical Computing*. ISBN 3-900051-07-0. R Foundation for Statistical Computing. Vienna, Austria. URL: `http://www.R-project.org`.

Rabin, Laura A. et al. (2015). "Subjective Cognitive Decline in Older Adults: An Overview of Self-Report Measures used Across 19 International Research Studies". In: *Journal of Alzheimer's Disease* Preprint, pp. 1–25. ISSN: 1387-2877.

Radford, Alec, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever (2018). "Improving language understanding by generative pre-training". In: *URL https://s3-us-west-2. amazonaws. com/openai-assets/researchcovers/languageunsupervised/langua understanding paper. pdf*.

Raganato, Alessandro, Claudio Delli Bovi, and Roberto Navigli (2017). "Neural sequence learning models for word sense disambiguation". In: *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pp. 1167–1178.

Raganato, Alessandro, Jose Camacho-Collados, and Roberto Navigli (2017). "Word Sense Disambiguation: A Unified Evaluation Framework and Empirical Comparison". In: *Proc. of EACL*, pp. 99–110.

Raiman, Jonathan and Olivier Raiman (2018). "DeepType: Multilingual Entity Linking by Neural Type System Evolution". In: *CoRR* abs/1802.01021. arXiv: 1802.01021. URL: http://arxiv.org/abs/1802.01021.

Rastier, François (1974). "Systematik der Isotopien". In: *Lekturekolleg zur Textlinguistik. Bd* 2, pp. 153–190.

Ratinov, Lev, Dan Roth, Doug Downey, and Mike Anderson (2011). "Local and global algorithms for disambiguation to wikipedia". In: *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*. Association for Computational Linguistics, pp. 1375–1384.

Reuber, Markus, Chiara Monzoni, Basil Sharrack, and Leendert Plug (2009). "Using interactional and linguistic analysis to distinguish between epileptic and psychogenic nonepileptic seizures: A prospective, blinded multirater study". In: *Epilepsy and Behavior* 16.1, pp. 139–144. DOI: 10.1016/j.yebeh.2009.07.018.

Rizzo, Maria L and Gabor J Szekely (2017). "Package 'energy'". In:

Roberts, Judith L and Linda Clare (2013). "Meta-representational awareness in mild cognitive impairment: An interpretative phenomenological analysis". In: *Aging & Mental Health* 17.3, pp. 300–309.

Rockwell, Geoffrey and Stéfan Sinclair (2016). *Hermeneutica: Computer-Assisted Interpretation in the Humanities*. MIT: MIT Press.

Ruecker, Stan, Milena Radzikowska, and Stéfan Sinclair (2011). *Visual interface design for digital cultural heritage: A guide to rich-prospect browsing*. Ashgate Publishing, Ltd.

Rule, Alix, Jean-Philippe Cointet, and Peter S Bearman (2015). "Lexical shifts, substantive changes, and continuity in State of the Union discourse, 1790–2014". In: *Proceedings of the National Academy of Sciences* 112.35, pp. 10837–10844.

Savova, Guergana K et al. (2010). "Mayo clinical Text Analysis and Knowledge Extraction System (cTAKES): architecture, component evaluation and applications". In: *Journal of the American Medical Informatics Association* 17.5, pp. 507–513.

Schönhofen, Peter (2009). "Identifying document topics using the Wikipedia category network". In: *Web Intelligence and Agent Systems: An International Journal* 7.2, pp. 195–207.

Selting, Margret et al. (2009). "Gesprächsanalytisches Transkriptionssystem 2 (GAT 2)". In: *Gesprächsforschung: Online-Zeitschrift zur verbalen Interaktion* 10, pp. 353–402.

Silge, Julia and David Robinson (2016). "tidytext: Text Mining and Analysis Using Tidy Data Principles in R". In: *JOSS* 1.3. DOI: 10.21105/joss.00037. URL: http://dx.doi.org/10.21105/joss.00037.

Silva, João, Luísa Coheur, Ana Cristina Mendes, and Andreas Wichert (2011). "From symbolic to sub-symbolic information in question classification". In: *Artificial Intelligence Review* 35.2, pp. 137–154.

Sinclair, Stéfan and Geoffrey Rockwell (2012). "Introduction to Distant Reading Techniques with Voyant Tools, Multilingual Edition". In: *Digital Humanities 2012, DH 2012, Conference Abstracts, University of Hamburg, Hamburg, Germany, July 16-22, 2012*, p. 26. URL: http://www.dh2012.uni-hamburg.de/conference/programme/abstracts/introduction-to-distant-reading-techniques-with-voyant-tools-multilingual-edition.1.html.

Smilkov, Daniel et al. (2016). "Embedding projector: Interactive visualization and interpretation of embeddings". In: *arXiv preprint arXiv:1611.05469*.

Smith, A (1982). "Symbol digit modalities test (SDMT) manual (revised) Western Psychological Services". In: *Los Angeles*.

Smith, J., P. Flowers, and M. Larkin (2009). *Interpretative Phoneomological Analysis: theory, method and research*.

Song, Hongyan and Tianfang Yao (2009). "Improving topic extraction in chinese documents using word sense disambiguation". In: *2009 Fourth International Conference on Innovative Computing, Information and Control (ICICIC)*. IEEE, pp. 1106–1109.

Sriram, Bharath, Dave Fuhry, Engin Demir, Hakan Ferhatosmanoglu, and Murat Demirbas (2010). "Short Text Classification in Twitter to Improve Information Filtering". In: *Proceedings of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '10)*. Geneva, Switzerland, pp. 841–842. DOI: 10.1145/1835449.1835643.

Srivastava, Nitish, Geoffrey E Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov (2014). "Dropout: a simple way to prevent neural

networks from overfitting." In: *Journal of machine learning research* 15.1, pp. 1929–1958.

Stegbauer, Christian (2009). "Wikipedia". In: *Das Rätsel der Kooperation. Wiesbaden* 5.

Strauss, Anselm and Juliet M Corbin (1997). *Grounded theory in practice*. Sage.

Strötgen, Jannik and Michael Gertz (2010). "HeidelTime: High quality rule-based extraction and normalization of temporal expressions". In: *Proc. of the 5th International Workshop on Semantic Evaluation*, pp. 321–324.

Sueddeutsche Zeitung (2018). *Sueddeutsche Zeitung*. `www.sueddeutsche.de`. Accessed: 2018-11-22.

Sun, Xiaobing, Xiangyue Liu, Jiajun Hu, and Junwu Zhu (2014). "Empirical studies on the nlp techniques for source code data preprocessing". In: *Proceedings of the 2014 3rd International Workshop on Evidential Assessment of Software Technologies*. ACM, pp. 32–39.

Syed, Zareen, Tim Finin, and Anupam Joshi (2008). "Wikipedia as an ontology for describing documents". In: *UMBC Student Collection*.

Tang, Jie, Jimeng Sun, Chi Wang, and Zi Yang (2009). "Social Influence Analysis in Large-scale Networks". In: *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD '09. New York, NY, USA: ACM, pp. 807–816. ISBN: 978-1-60558-495-9.

Tripathi, Suraj, Chirag Singh, Abhay Kumar, Chandan Pandey, and Nishant Jain (2019). "Bidirectional Transformer Based Multi-Task Learning for Natural Language Understanding". In: *International Conference on Applications of Natural Language to Information Systems*. Springer, pp. 54–65.

Tripodi, Rocco and Marcello Pelillo (2016). "A game-theoretic approach to word sense disambiguation". In: *arXiv preprint arXiv:1606.07711*.

Türker, Rima, Lei Zhang, Maria Koutraki, and Harald Sack (2019). "Knowledge-based short text categorization using entity and category embedding". In: *European Semantic Web Conference*. Springer, pp. 346–362.

Uslu, Tolga, Wahed Hemati, Alexander Mehler, and Daniel Baumartz (2017). "TextImager as a Generic Interface to R". In: *EACL 2017*, p. 17.

Uslu, Tolga and Alexander Mehler (2018). "PolyViz: a Visualization System for a Special Kind of Multipartite Graphs". In: *Proceedings of the IEEE VIS 2018*. IEEE VIS 2018. accepted. Berlin, Germany.

Uslu, Tolga, Alexander Mehler, and Daniel Baumartz (2019). "Computing Classifier-based Embeddings with the Help of text2ddc". In: *Proceedings of the 20th International Conference on Computational Linguistics and Intelligent Text Processing, (CICLing 2019)*. CICLing 2019. La Rochelle, France.

Uslu, Tolga, Alexander Mehler, Daniel Baumartz, Alexander Henlein, and Wahed Hemati (2018). "fastsense: An efficient word sense disambiguation classifier". In: *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC-2018)*.

Uslu, Tolga, Alexander Mehler, and Dirk Meyer (2018). "LitViz: Visualizing Literary Data by Means of text2voronoi". In: *Proceedings of the Digital Humanities 2018*. DH2018. Mexico City, Mexico.

Uslu, Tolga, Alexander Mehler, Andreas Niekler, and Daniel Baumartz (2018). "Towards a DDC-based Topic Network Model of Wikipedia". In: *Proceedings of 2nd International Workshop on Modeling, Analysis, and Management of Social Networks and their Applications (SOCNET 2018), February 28, 2018*.

Uslu, Tolga, Lisa Miebach, et al. (2018). "Automatic Classification in Memory Clinic Patients and in Depressive Patients". In: *Proceedings of Resources and ProcessIng of linguistic, para-linguistic and extra-linguistic Data from people with various forms of cognitive/psychiatric impairments (RaPID-2)*. RaPID. Miyazaki, Japan.

van der Loo, M.P.J. (2014). "The stringdist package for approximate string matching". In: *The R Journal* 6 (1), pp. 111–122. URL: https://CRAN.R-project.org/package=stringdist.

Vaswani, Ashish et al. (2017). "Attention is all you need". In: *Advances in neural information processing systems*, pp. 5998–6008.

Vial, Loïc, Benjamin Lecouteux, and Didier Schwab (2017). "Sense Embeddings in Knowledge-Based Word Sense Disambiguation". In: *12th International Conference on Computational Semantics*.

Vincent, Luc (2007). "Google book search: Document understanding on a massive scale". In: *Ninth International Conference on Document Analysis and Recognition (ICDAR 2007)*. Vol. 2. IEEE, pp. 819–823.

Voorhees, Ellen M. and Dawn M. Tice (2000). "Building a Question Answering Test Collection". In: *Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. SIGIR '00. Athens, Greece: ACM, pp. 200–207. ISBN: 1-58113-226-3. DOI: 10.1145/345508.345577. URL: http://doi.acm.org/10.1145/345508.345577.

Voss, Jakob (2006). "Collaborative thesaurus tagging the Wikipedia way". In: *CoRR* abs/cs/0604036. arXiv: cs/0604036. URL: http://arxiv.org/abs/cs/0604036.

Waltinger, Ulli, Alexander Mehler, Mathias Lösch, and Wolfram Horstmann (2011). "Hierarchical Classification of OAI Metadata Using the DDC Taxonomy". In: *Advanced Language Technologies for Digital Libraries (ALT4DL)*. Ed. by Raffaella Bernardi, Sally Chambers, Bjoern Gottfried, Frederique Segond, and Ilya Zaihrayeu. LNCS. Berlin: Springer, pp. 29–40.

Wang, Peng et al. (2016). "Semantic expansion using word embedding clustering and convolutional neural network for improving short text classification". In: *Neurocomputing* 174, Part B, pp. 806–814. ISSN: 0925-2312.

Wattenberg, Martin (2001). "The shape of song". In: *Website http://www.turbulence. org/Works/song/mono. html*.

– (2002). "Arc diagrams: Visualizing structure in strings". In: *Information Visualization, 2002. INFOVIS 2002. IEEE Symposium on*. IEEE, pp. 110–116.

Watts, Duncan J and Steven H Strogatz (1998). "Collective dynamics of 'small-world'networks". In: *nature* 393.6684, pp. 440–442.

Wickham, Hadley (2014). "Tidy Data". In: *Journal of Statistical Software* 59.1, pp. 1–23. ISSN: 1548-7660. DOI: 10.18637/jss.v059.i10. URL: https://www.jstatsoft.org/index.php/jss/article/view/v059i10.

Wild, Fridolin (2015). *lsa: Latent Semantic Analysis*. R package version 0.73.1. URL: https://CRAN.R-project.org/package=lsa.

Winata, Genta Indra, Samuel Cahyawijaya, Zhaojiang Lin, Zihan Liu, and Pascale Fung (2019). "Lightweight and Efficient End-to-End Speech Recognition Using Low-Rank Transformer". In: *arXiv preprint arXiv:1910.13923*.

Wolfsgruber, Steffen et al. (2015). "Subjective cognitive decline is related to CSF biomarkers of AD in patients with MCI". In: *Neurology* 84.12, pp. 1261–1268. ISSN: 1526-632X.

Wu, Ledell et al. (2017). "StarSpace: Embed All The Things!" In: *CoRR* abs/1709.03856. arXiv: 1709.03856. URL: http://arxiv.org/abs/1709.03856.

Yang, Tze-I, Andrew Torget, and Rada Mihalcea (June 2011). "Topic Modeling on Historical Newspapers". In: *Proceedings of the 5th ACL-HLT Workshop on Language Technology for Cultural Heritage, Social Sciences, and Humanities*.

Portland, OR, USA: Association for Computational Linguistics, pp. 96–104. URL: https://www.aclweb.org/anthology/W11-1513.

Yesavage, Jerome A. et al. (1983). "Development and validation of a geriatric depression screening scale: a preliminary report". In: *Journal of psychiatric research* 17.1, pp. 37–49. ISSN: 0022-3956.

Yuan, Dayu, Ryan Doherty, Julian Richardson, Colin Evans, and Eric Altendorf (2016). "Word Sense Disambiguation with Neural Language Models". In: *arXiv preprint arXiv:1603.07012*.

Zhang, Xiang and Yann LeCun (2015). "Text Understanding from Scratch". In: *CoRR* abs/1502.01710. arXiv: 1502.01710. URL: http://arxiv.org/abs/1502.01710.

Zhao, Wei, Liang Wang, Kewei Shen, Ruoyu Jia, and Jingming Liu (2019). "Improving grammatical error correction via pre-training a copy-augmented architecture with unlabeled data". In: *arXiv preprint arXiv:1903.00138*.

Zhong, Zhi and Hwee Tou Ng (2010). "It makes sense: A wide-coverage word sense disambiguation system for free text". In: *Proceedings of the ACL 2010 System Demonstrations*. Association for Computational Linguistics, pp. 78–83.

Zihl, Josef, Simone Reppermund, Sonja Thum, and Kathrin Unger (2010). "Neuropsychological profiles in MCI and in depression: Differential cognitive dysfunction patterns or similar final common pathway disorder?" In: *Journal of psychiatric research* 44.10, pp. 647–654.

Appendix

Zusammenfassung

Viele Methoden wurden in dieser Arbeit vorgestellt, die sich mit dem Hauptziel der automatischen Dokumentenanalyse auf semantischer Ebene befassen. Um das Hauptziel zu erreichen, mussten wir jedoch zunächst eine solide Basis entwickeln, um das Gesamtbild zu vervollständigen. So wurden verschiedene Methoden und Werkzeuge entwickelt, die verschiedene Aspekte des NLP abdecken. Das Zusammenspiel dieser Methoden ermöglichte es, unser Ziel erfolgreich zu erreichen. Neben der automatischen Dokumentenanalyse legen wir großen Wert auf die drei Prinzipien (siehe Kapitel 1.2.1) von *Effizienz*, *Anwendbarkeit* und *Sprachunabhängigkeit*. Dadurch waren die entwickelten Tools für die Anwendungen bereit. Die Größe und Sprache der zu analysierenden Daten ist kein Hindernis mehr, zumindest für die im Bezug auf die von Wikipedia unterstützten Sprachen.

Einen großen Beitrag dazu leistete `TextImager` (siehe Kapitel 2), das Framework, dass für die zugrunde liegende Architektur verschiedener Methoden und die gesamte Vorverarbeitung der Texte verantwortlich ist. TextImager ist als Multi-Server und Multi-Instanz-Cluster konzipiert, sodass eine verteilte Verarbeitung von Daten ermöglicht wird. Hierfür werden die Cluster-Management-Dienste UIMA-AS[4] und UIMA-DUCC[5] verwendet. Darüber hinaus ermöglicht die Multi-Service-Architektur von TextImager die Integration beliebiger NLP-Tools und deren gemeinsame Ausführung. Zudem bietet der TextImager eine webbasierte Benutzeroberfläche, die eine Reihe von interaktiven Visualisierungen bietet, die die Ergebnisse der Textanalyse darstellen. Das Webinterface erfordert keine Programmierkenntnisse – durch einfaches Auswählen der NLP-

[4] uima.apache.org/doc-uimaas-what.html
[5] uima.apache.org/doc-uimaducc-whatitam.html

Komponenten und der Eingabe des Textes wird die Analyse gestartet und anschließend visualisiert, so dass auch Nicht-Informatiker mit diesen Tools arbeiten können.

In Kapitel 3 demonstrierten wir die Integration des statistischen Frameworks `R` in die Funktionalität und Architektur von TextImager. Hier haben wir die OpenCPU-API verwendet, um R-Pakete auf unserem eigenen R-Server bereitzustellen. Dies ermöglichte die Kombination von R-Paketen mit den modernsten NLP-Komponenten des TextImager. So erhielten die Funktionen der R-Pakete extrahierte Informationen aus dem TextImager, was zu verbesserten Analysen führte. Darüber hinaus haben wir interaktive Visualisierungen integriert, um die von R abgeleiteten Informationen zu visualisieren.

Einige der im TextImager entwickelten Visualisierungen sind besonders herausragend und haben in vielen Bereichen Anwendung gefunden. Ein Beispiel dafür ist `PolyViz`, ein interaktives Visualisierungssystem, das die Darstellung eines multipartiten Graphen ermöglicht. Im Kapitel 4 haben wir PolyViz anhand von zwei verschiedenen Anwendungsfällen veranschaulicht. Darüber hinaus wurde die Visualisierung weiterentwickelt, so dass die einzelnen Gruppen geclustert werden können (siehe Abbildung 13.6).

Abbildung 13.7 zeigt die visuellen Effekte bei der Interaktion mit einem bestimmten Cluster. Alle zugehörigen Knoten und Kanten innerhalb und außerhalb der Gruppen werden entsprechend hervorgehoben.

`SemioGraph`, eine Visualisierungstechnik zur Darstellung multikodaler Graphen, wurde in Kapitel 5 vorgestellt. Die visuellen und interaktiven Funktionen von SemioGraph wurden mit einer Anwendung zur Visualisierung von Worteinbettungen vorgestellt. Wir haben gezeigt, dass verschiedene Modelle zu völlig unterschiedlichen Grafiken führen können. So kann Semiograph bei der Suche nach Worteinbettungen für bestimmte NLP-Aufgaben helfen.

Inspiriert von all den Textvisualisierungen im TextImager ist die Idee für `text2voronoi` entstanden (siehe Kapitel 6). Hier stellten wir einen neuartigen Ansatz zur bildgetriebenen Textklassifizierung vor, der auf einem Voronoi-Diagram linguistischer Merkmale basiert. Dieser Klassifikationsansatz wurde auf die automatische Patientendiagnose angewendet und wir haben gezeigt, dass wir das traditionelle Bag-Of-Words-Modell sogar übertreffen. Dieser

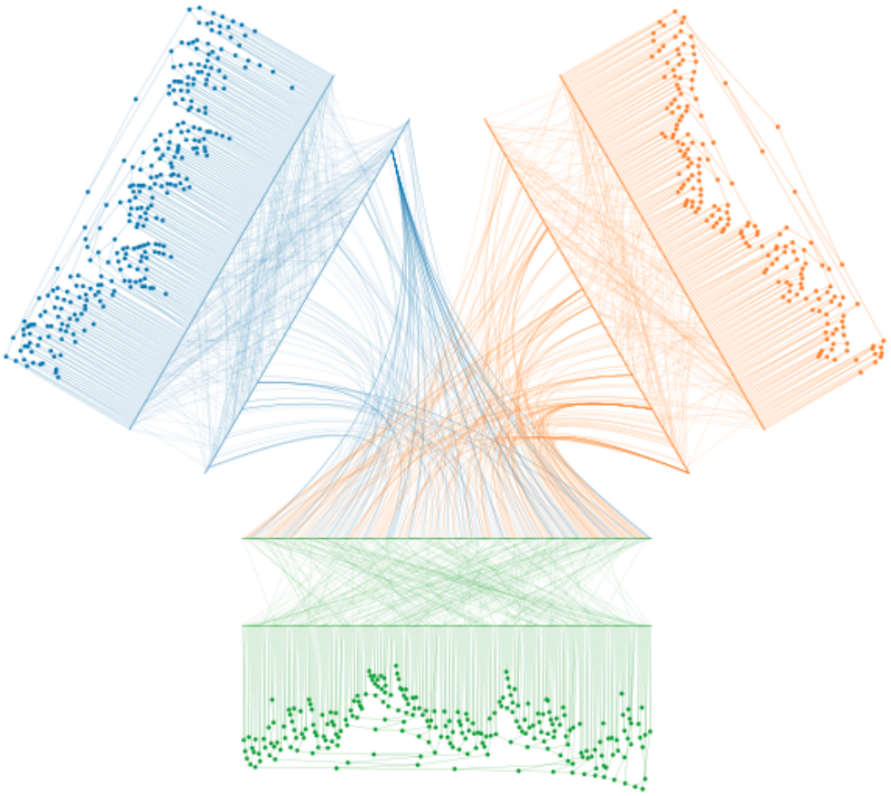
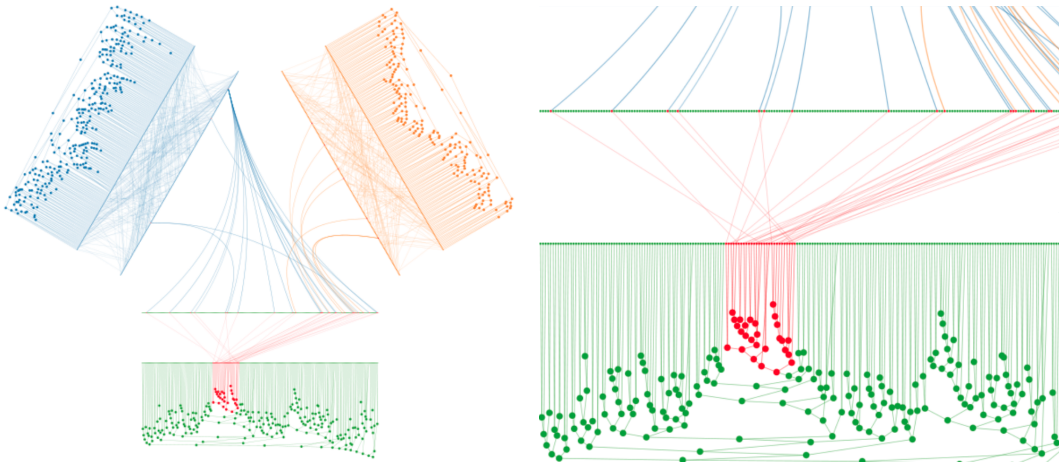


Figure 13.6: Weiterentwicklung von PolyViz inklusive Clusteranalyse.



(a) Overview (b) Zoomed in

Figure 13.7: Visuelle Hervorhebung bei der Interaktion mit PolyViz.

Ansatz ermöglicht es, die zugrunde liegenden Merkmale anschließend zu analysieren und damit einen ersten Schritt zur Lösung der Black Box zu machen. Im Kapitel 8 haben wir weitere Analysen zur automatischen Klassifizierung von Gedächtnisklinikpatienten und depressiven Patienten durchgeführt.

Wir haben text2voronoi auf literarische Werke angewendet (siehe Kapitel 7) und die entstandenen Visualisierungen auf einer webbasierten Oberfläche (`LitViz`) präsentiert. Hier ermöglichen wir den Vergleich von Voronoi-Diagrammen der verschiedenen Literaturen und damit den visuellen Vergleich der Sprachstile der zugrunde liegenden Autoren.

Mit unserer Kompetenz in der Vorverarbeitung und der Analyse von Texten sind wir unserem Ziel der semantischen Dokumentenanalyse einen Schritt näher gekommen. Als nächstes haben wir die Auflösung der Sinne auf der Wortebene im Kapitel 9 untersucht. Hier stellten wir `fastSense` vor, ein Disambigierungsframework, das mit großen Datenmengen zurecht kommt. Um dies zu erreichen, haben wir einen Disambiguierungskorpus erstellt, der auf Wikipedias 221,965 Disambiguierungsseiten basiert, wobei die sich auf 825,179 Sinne beziehen. Daraus resultierten mehr als 50 Millionen Datensätze, die fast 50 GB Speicherplatz benötigten. Wir haben nicht nur gezeigt, dass fastSense eine so große Datenmenge problemlos verarbeiten kann, sondern auch, dass wir mit unseren Wettbewerbern mithalten und sie bei einigen NLP-Aufgaben sogar übertreffen können.

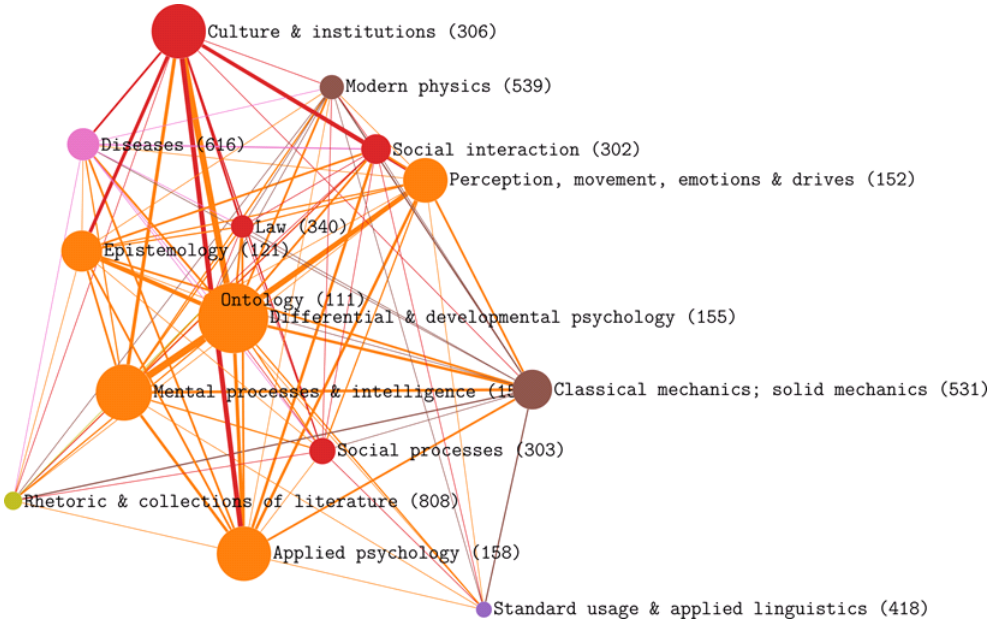
Jetzt, da wir den Wörtern Sinne zuordnen können, sind wir der semantischen Dokumentenanalyse einen weiteren Schritt näher gekommen. Je mehr Informationen wir aus einem Text und seinen Wörtern gewinnen können, desto genauer können wir seinen Inhalt analysieren, wie in den Kapiteln 10 bis 12 bestätigt wurde. Ausgehend von Kapitel 10 stellten wir einen netzwerktheoretischen Ansatz zur Modellierung der Semantik großer Textnetzwerke am Beispiel der deutschen Wikipedia vor. Zu diesem Zweck haben wir einen Algorithmus namens `text2ddc` entwickelt, um die thematische Struktur eines Textes zu modellieren. Dabei basiert das Modell auf einem etablierten Klassifikationsschema, nämlich der `Dewey Decimal Classification`. Mit diesem Modell haben wir gezeigt, wie man aus der Vogelperspektive die Hervorhebung und Verknüpfung von Themen, die sich in Millionen von Dokumenten manifestiert, darstellt. So haben wir eine Möglichkeit geschaffen, die thematische Dynamik von Dokumentnetzwerken automatisch zu visualisieren. Die Trainings- und Testdaten, die wir in diesem Kapitel hatten, bestanden jedoch hauptsächlich aus kurzen Textausschnitten. In Kapitel 11 haben wir DDC Korpora erstellt, indem wir Informationen aus Wikidata, Wikipedia und der von der Deutschen Nationalbibliothek verwalteten Gemeinsamen Normdatei

(GND) vereinigt haben. Auf diese Weise konnten wir nicht nur die Datenmenge erhöhen, sondern auch Datensätze für viele bisher unzugängliche Sprachen erstellen. Wir haben text2ddc so weit optimiert, dass wir einen F-score von 87.4% erzielen für die 98 Klassen der zweiten DDC-Stufe. Die Vorverarbeitung von TextImager und die Disambiguierung durch fastSense hatten einen großen Einfluss darauf. Für jedes Textstück berechnet text2ddc eine Wahrscheinlichkeitsverteilung über die DDC-Klassen berechnen.

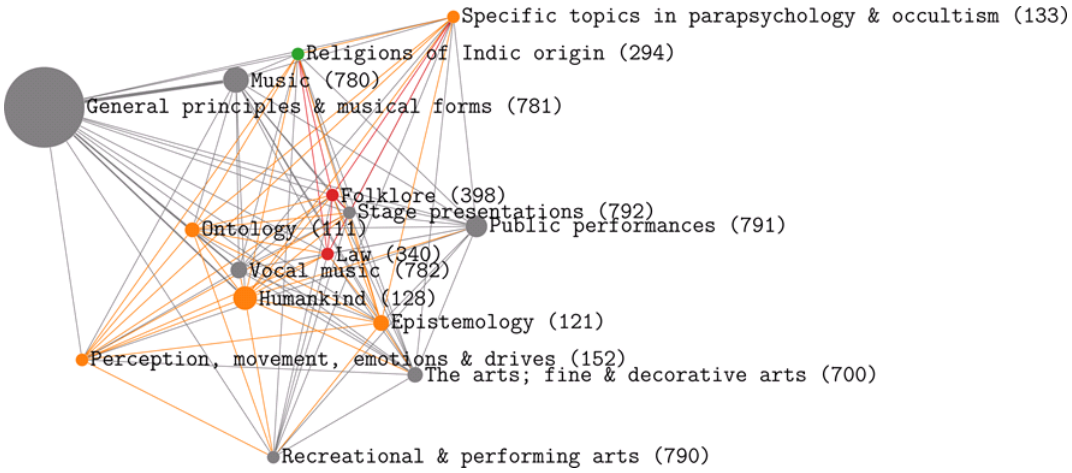
text2ddc wurde bereits für weitere Forschungsfragen verwendet. Mit text2ddc analysiert Mehler, Uslu, Rüdiger Gleim, et al. (2019) beispielsweise literarische Werke bekannter Autoren nach ihrem thematischen Inhalt. Abbildung 13.8 zeigt einen thematischen Vergleich zwischen den Autoren Sigmund Freud und Friedrich Nietzsche.

Mehler, Rüdiger Gleim, Gaitsch, et al. (2019) verwenden text2ddc, um die Stadtwikis zu erkunden. Hier analysierten sie, welche Themen in verschiedenen Städten diskutiert werden und wie sie sich zueinander verhalten. Abbildung 13.9 zeigt die thematische Relation von München und Dresden, wobei im Fall von München PolyViz zur Veranschaulichung der Verteilung verwendet wurde.

Der klassifikatorinduzierte semantische Raum von text2ddc wurde auch zur Verbesserung weiterer NLP-Methoden genutzt. Dazu gehört auch `text2wiki`, ein Framework für automatisches Tagging nach dem Wikipedia-Kategoriensystem. Auch hier haben wir einen klassifikatorinduzierten semantischen Raum, aber diesmal basiert er auf dem Wikipedia-Kategoriensystem. Ein großer Vorteil dieses Modells ist die Präzision und Tiefe der behandelten Themen und das sich ständig weiterentwickelnde Kategoriesystem. Damit sind auch die Kriterien eines offenen Themenmodells erfüllt. Um die Vorteile von text2wiki zu demonstrieren, haben wir anschließend die von text2wiki bereitgestellten Themenvektoren verwendet, um text2ddc zu verbessern, so dass sich beide Systeme gegenseitig verbessern können. Die Synergie zwischen den erstellten Methoden in dieser Dissertation war entscheidend für den Erfolg jeder einzelnen Methode. Abbildung 13.10 zeigt eine detailliertere Ansicht des Interaktionsmodells aus Abschnitt 1.3. Aus Übersichtlichkeitsgründen haben wir die Schwerpunkte dieser Dissertation hervorgehoben. Da der Schwerpunkt der Dissertation die semantische Analyse mehrerer Dokumente ist, haben wir



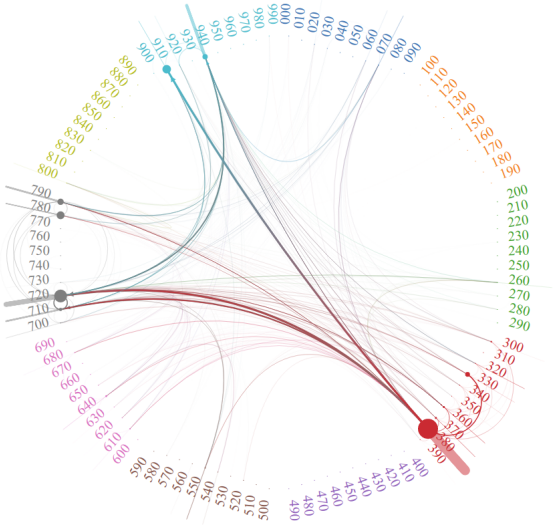
(a) Sigmund Freud



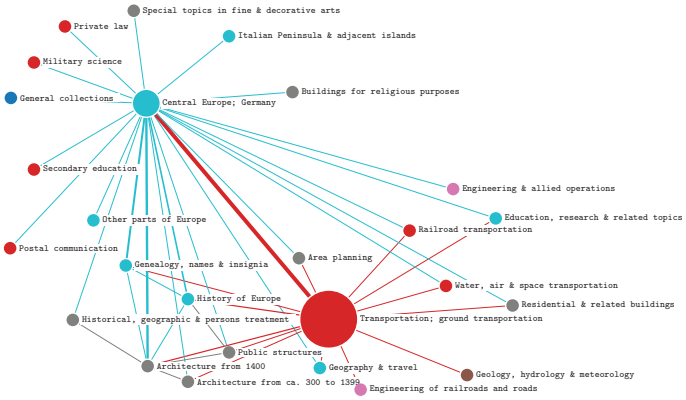
(b) Friedrich Nietzsche

Figure 13.8: Vergleich der thematischen Zusammensetzung zweier Autoren (Mehler, Uslu, Rüdiger Gleim, et al. 2019).

die 3 Methoden fastSense, text2ddc und text2wiki hervorgehoben. Darüber hinaus haben wir die Kanten hervorgehoben, bei denen diese Werkzeuge Verwendung gefunden haben. Dabei steht eine gerichtete Kante für die Verwendung des Startknotens im Zielknoten. Wir haben die Methoden zur semantischen Textanalyse erfolgreich entwickelt und auf weitere Methoden übertragen. Auch die Methoden, die in Abbildung 13.10 nicht hervorgehoben sind, haben dazu einen wichtigen Beitrag geleistet. TextImager bot immer eine starke zu-



(a) München



(b) Dresden

Figure 13.9: Thematischer Aufbau in Stadtwikis (Mehler, Rüdiger Gleim, Gaitsch, et al. 2019).

grunde liegende Architektur und stellte extrahierte Informationen durch seine Vorverarbeitung zur Verfügung. Durch die Visualisierungen konnten die Daten jederzeit analysiert und Fehler frühzeitig erkannt werden. Mit SemioGraph konnten spezifische Worteinbettungen gefunden werden, die für die jeweilige Aufgabenbereiche geeignet sind. Letztendlich unterstützten sich auch die semantischen Modelle gegenseitig und führten zu Verbesserungen.

Die Beiträge dieser Dissertation bieten noch Raum für Verbesserungen, die in der zukünftigen Arbeit angegangen werden können. Zum einen können die einzelnen Methoden weiter optimiert werden, zum anderen bieten diese

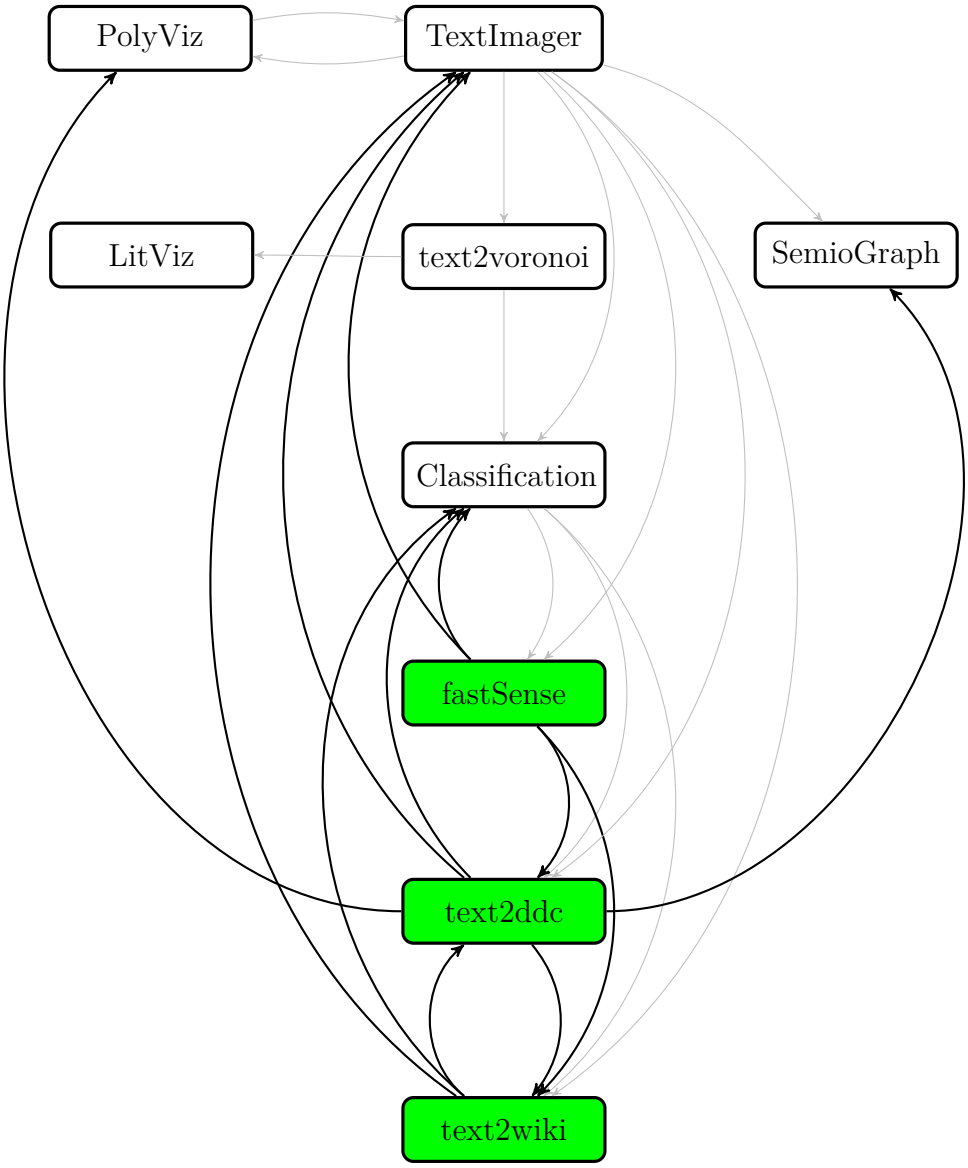


Figure 13.10: Eine detailliertere Ansicht des Interaktionsmodells aus Sektion 1.3, die den Schwerpunkt dieser Dissertation hervorhebt.

Methoden zahlreiche Möglichkeiten für weitere Forschungsfragen.

Man könnte beispielsweise das Themenmodell auf Basis des DDC verbessern, indem man mehr Trainingsdaten generiert. In text2ddc haben wir Wikipedia, Wikidata und GND kombiniert, um Wikipedia-Artikel mit DDC-Informationen zu versehen. Allerdings waren wir hier auf GND beschränkt, so dass viele Wikipedia-Artikel ohne DDC-Tag geblieben sind. Eine Idee wäre es, die DDC-Klassen auf Wikipedia-Kategorien abzubilden und so neue Trainingsdaten zu finden, da jeder Wikipedia-Artikel mit Wikipedia-Kategorien versehen ist. Da

Wikipedia-Kategorien auch in anderen Sprachen verfügbar sind, hat dies den Vorteil, dass mehr Daten für ressourcenarme Sprachen erstellt werden können. Ein weiterer Vorteil des obigen Ansatzes wäre die Verbesserung der dritten Stufe des DDC, weil dadurch alle Klassen dieser Stufe abgedeckt wären und auch viel mehr Daten generiert werden könnten.

Ein weitere Methode, die in Zukunft verbessert werden könnte, wäre fastSense. In fastSense haben wir auf Basis der Wikipedia-Disambiguierungsseiten mehrdeutige Wörter ermittelt und einen Datensatz aus den entsprechenden Artikeln erstellt. Wir haben Wikipedia als Grundlage verwendet, da es eine große Datenquelle und eine sehr detaillierte Aufteilung der Sinne bietet. In einigen Fällen ist diese Aufteilung jedoch entweder zu genau, sodass mehr als 40 Bedeutungen für ein Wort entstehen können, oder zu grob, sodass bestimmte Bedeutungen fehlen können. Dies ist der Fall, da Wikipedia eine kollaborative Enzyklopädie ist, die es den Nutzern ermöglicht, Wörter mit beliebiger Genauigkeit in ihre Sinne zu unterteilen. Hier könnte es hilfreich sein, Ressourcen zu integrieren, die von Experten erstellt und gepflegt werden, wie z.B. der Duden[6]. Ein weiterer Vorteil zusätzlicher Ressourcen wäre auch, dass sie sich nicht auf Substantive beschränken, wie im Fall von Wikipedia. Dies würde es jedoch ermöglichen, das zugrunde liegende Modell von fastSense zu verbessern und zu erweitern, ohne die Vorteile von Wikipedia zu verlieren. Eine weitere Funktion, von der fastSense profitieren könnte, ist die automatische Erkennung von Mehrwortausdrücken. In den durchgeführten Experimenten (Wikipedia und Evaluationsaufgaben) sind die zu disambiguierenden Wörter bereits festgelegt. In der Praxis müssen diese Worte jedoch zuerst gefunden werden. Hier arbeiten wir derzeit mit einer einfachen Suche mit regulären Ausdrücken. Neuere Methoden, die mit neuronalen Netzen arbeiten, könnten zur Optimierung dieses Prozesses eingesetzt werden.

In der jüngsten Vergangenheit sind Transformer-Architekturen (Vaswani et al. 2017) in die Welt des maschinellen Lernens eingetreten. Es gibt bereits zahlreiche Forschungsartikel, die signifikante Verbesserungen durch die Verwendung von Transformer-Architekturen erzielen (Radford et al. 2018; Tripathi et al. 2019; Prato, Charlaix, and Rezagholizadeh 2019; Zhao et al. 2019). Der größte Nachteil dieser Architektur ist jedoch ihre langsame Performanz, wobei es bereits erste Ansätze zur Beschleunigung dieser Architektur

[6] `https://www.duden.de/`

gibt (Winata et al. 2019; Murray et al. 2019; Gong et al. 2019). Wenn die Transformer-Architekturen eines Tages in der Lage sein werden, die Millionen von Dokumenten, mit denen wir es in dieser Dissertation zu tun haben, auf einem Standardcomputer zu analysieren, könnten wir sie auch auf unsere Datensätze anwenden und die Modelle verbessern.

Neben der Verbesserung der entwickelten Werkzeuge wurden viele neue Forschungsfragen für zukünftige Arbeiten geschaffen. Dafür sind insbesondere die beiden entwickelten Themenmodelle text2ddc und text2wiki verantwortlich. Durch den Einsatz unserer effizienten geschlossenen und offenen Themenmodelle ist es möglich, verschiedene Korpora nach ihrer Themenverteilung zu analysieren, wobei die Größe der Korpora kein Thema mehr ist. Das geschlossene Themenmodell eignet sich besonders gut, um den zeitlichen Verlauf von Themen zu messen und zu vergleichen. So könnten beispielsweise verschiedenen Zeitpunkte der Wikipedia thematisch untersucht werden. Dies könnte auch auf jede andere Plattform oder soziale Netzwerk angewendet werden, um bestimmte Trends zu identifizieren. Man könnte auch diese zeitliche thematische Verschiebung analysieren und mögliche zukünftige Trends prognostizieren.

Die Vergleichbarkeit ist bei dem offenen Themenmodell (text2wiki) nicht ganz so einfach, da sich das thematische Universum ständig weiterentwickelt. Es könnte jedoch eine bestimmte Zeit als Betrachtungspunkt für die Themen ausgewählt werden. Ein anderer Ansatz wäre es, einen bestimmten Textkorpus aus der Sicht verschiedener Zeiträume zu betrachten. Dies würde eine Art Zeitreise aus thematischer Sicht ermöglichen.

Lebenslauf

Tolga Uslu

Goethe-Universität Frankfurt am Main
Fachbereich Informatik und Mathematik, Abteilung Texttechnologie

Robert-Mayer-Straße 10, 60 325 Frankfurt am Main

Geburtsdatum: 16.01.1991
Geburtsort: Frankfurt am Main
Telefon: (0 69) 7 98 - 2 46 62
E-Mail: uslu@em.uni-frankfurt.de
www: https://www.texttechnologylab.org/team/tolga-uslu/

Privatanschrift:
Kurfürstenstraße 55
60486 Frankfurt am Main
0 15 90 42384280

Wissenschaftliche Qualifikationen

12/2014 – heute	**Promotionsstudium** an der Goethe-Universität Frankfurt am Main
10/2012 – 11/2014	**Master-Studium** Informatik an der Goethe-Universität Frankfurt am Main Abschluss: *Master of Science* (sehr gut)
10/2010 – 10/2012	**Bachelor-Studium** Informatik an der Goethe-Universität Frankfurt am Main Abschluss: *Bachelor of Science* (sehr gut)
10/2009 – 09/2010	**Schülerstudium** (Hochbegabten Programm) Informatik an der Goethe-Universität Frankfurt am Main
08/2001 – 06/2010	**Abitur** am Goethe-Gymnasium Frankfurt am Main

Berufliche Stationen

12/2014 – heute **Wissenschaftlicher Mitarbeiter** in der Arbeitsgruppe Texttechnologie am Fachbereich 12 - Informatik und Mathematik der Goethe-Universität Frankfurt, in verschiedenen Projekten.

04/2012 – 09/2014 **Wissenschaftliche Hilfskraft** in der Angewandten Informatik am Fachbereich 12 - Informatik und Mathematik der Goethe-Universität Frankfurt.

Lehre

Nachfolgend das Verzeichnis meiner bisher angebotenen Lehrveranstaltungen.

Titel[d]	**Art**	**Ort**	**Semester**[a b]	**SWS**[c]
Deep Learning for Text-Imaging	Praktikum	Goethe-Universität Frankfurt	WiSe 2019 / 2020	4
Deep Learning for Text-Imaging	Praktikum	Goethe-Universität Frankfurt	SoSe 2019	4
TextImaging	Praktikum	Goethe-Universität Frankfurt	WiSe 2018 / 2019	4
Deep Learning for Text-Imaging	Praktikum	Goethe-Universität Frankfurt	SoSe 2018	4
TextImaging	Praktikum	Goethe-Universität Frankfurt	WiSe 2017 / 2018	4
TextImaging	Praktikum	Goethe-Universität Frankfurt	SoSe 2017	4
TextImaging	Praktikum	Goethe-Universität Frankfurt	WiSe 2016 / 2017	4
TextImaging	Praktikum	Goethe-Universität Frankfurt	SoSe 2016	4

[a] WiSe = „Wintersemester“ [b] SoSe = „Sommersemester“ [c] SWS = „Semesterwochenstunden“
[d] gemeinsam mit Prof. Dr. Alexander Mehler und Wahed Hemati