Natalie Förster
6814258
Bachelor of Science
Informatik
7. Fachsemester
s7783707@stud.uni-frankfurt.de

Bachelor Thesis

Twitter Author Topic Modeling - Comparative and Classifactory Topic Analysis Using Latent Dirichlet Allocation

Natalie Förster

July 22, 2021

Lehrgebiet Texttechnology
Goethe-Universität Frankfurt am Main
Prof. Dr. Alexander Mehler

Abstract

The aim of this bachelor thesis is to compare and empirically test the use of classification to improve the topic models Latent Dirichlet Allocation (LDA) and Author Topic Modeling (ATM) in the context of the social media platform Twitter. For this purpose, a corpus was classified with the Dewey Decimal Classification (DDC) and then used to train the topic models. A second dataset, the unclassified corpus, was used for comparison. The assumption that the use of classification could improve the topic models did not prove true for the LDA topic model. Here, a sufficiently good improvement of the models could not be achieved. The ATM model, on the other hand, could be improved by using the classification. In general, the ATM model performed significantly better than the LDA model. In the context of the social media platform Twitter, it can thus be seen that the ATM model is superior to the LDA model and can additionally be improved by classifying the data.

Acknowledgement

First, I would like to express my special thanks to my supervisors at the Text Technology Lab at Goethe Universität, Frankfurt am Main. I would like to thank Prof. Dr. Alexander Mehler for being my mentor and enriching my work with his ideas and suggestions. I would like to direct my special thanks to Guiseppe Abrami, who helped me with every problem I had regarding the code and for all the other tips I have received from him. A special thanks also goes to my family and friends who have supported me in every situation during my studies.

Contents

List of Abbreviations

API Application Programming Interface

LDA Latent Dirichlet Allocation

ATM Author Topic Modeling

DDC Dewey Decimal Classification

JSON JavaScript Object Notation

XMI Extensible Markup Language Metadata Interchange

DNB Deutsche Nationalbibliothek

OCLC Online Computer Library Center

ELBO Evidence Lower Bound

NLP Natural Language Processing

BOW Bag-of-Words

PLDA Parallel Latent Dirichlet Allocation

LLDA Labeled Latent Dirichlet Allocation

CTM Correlated Topic Model

List of Figures

List of Tables

1 Introduction

1.1 Motivation

Natural Language Processing (NLP) describes the use of computers for the linguistic analysis of natural language text or speech to achieve human-like language processing for a range of tasks or applications (cf. Chowdhury 2003, p. 1; Liddy 2001, p. 2; Allen 2003, p. 1218).

Topic models, which originate from the field of NLP, analyse the distribution of semantic word groups in a text collection (also called documents). The text collection is processed by an algorithm and all topics contained in the document corpus are discovered. The usage of topic models is diverse. For example, reviews can be evaluated to generate opinions. Schmiedel et al. (2018) evaluated Glassdoor reviews, which provide insight into how and under what circumstances Fortune 500 companies are evaluated. Many other use cases dealing with reviews are conceivable (e.g., analyzing restaurant and hotel reviews). Another possible use case is the analysis of social media data. One could analyze the change in topics of a social media account over a certain period of time and also generate general opinions of certain groups from social media data (e.g., attitudes toward German climate policy among users younger than 20). In addition, popularity scores, represented, for example, in the form of likes, mentions, or retweets, offer the possibility of collecting data that encompasses a large number of people. In the context of social media, examining posts can be promising for many other applications. Breaking news can be detected, friend or advertising recommendations can be created, and sentiment analysis can be performed, to name a few. The use of social media data is beneficial as the data is freely available and abundant. However, there are dangers that social media texts bring, such as the limited length of posts, comments, and messages, and the noisiness in them due to orthographic errors and unstructured accounts. Since standard topic models are designed for longer and semantically correct texts, the performance of the models is severely limited by the use of social media short texts. This can lead to barely interpretable topics that are not informative. Therefore, there have been many attempts to improve the performance of topic models in social media, especially regarding Twitter. The improvements have been attempted by incorporating various pooling techniques. For example, Hong et al. (2010) developed the method to collect all texts of an author into one document (referred to as user pooling). Mehrotra et al. (2013) developed the aggregation method of collecting all tweets containing a particular hashtag into one document (referred to as hashtag pooling). Alvarez-Melis et al. (2016) have presented another method where conversations (an original tweet with all its comments) are combined into one document (referred to as conversation pooling). Although these techniques improved the models results to some extent, other problems arose, such as the merged documents often being thematically inconsistent.

Therefore, a new scheme is presented that modifies the preprocessing of tweets and potentially makes the use of special pooling techniques obsolete. The tweets are to be preselected by a classifier before being used to train a topic model. The aim is to increase the topic consistency of a corpus among the social media texts by this. The motivation behind the application of it is that even though a user has an account with a specific topic, they may publish posts that have nothing to do with the topic of their account. Accordingly, the classification of tweets should ensure that a corpus actually consists only of topic-related documents, which prevents the possible inconsistency of topics. The models trained on the classified corpus are then compared to those resulting from the models trained on the unclassified corpus, with a comprehensive empirical comparison. Both Latent Dirichlet Allocation (LDA) and Author Topic Modeling (ATM) topic models are trained. The models are then evaluated in terms of their topic quality and performance on a document retrieval task.

1.2 Structure

This work can be divided into four structural parts, starting with an introduction to the methodological handling of the tools, the software, a subsequent thematic explanation of topic modeling, its evaluation techniques, as well as the Dewey Decimal Classification (DDC) and an introduction to related work of this thesis. The second part shows the collection and preparation of the data and will continue with the processing phase, which will lead to the evaluation of the results in the third part. Finally, in the last part, the obtained results are discussed, challenges and following future research is proposed.

1.3 Preliminaries

The tools and software used are described below. A brief introduction to topic modeling and the used topic models, the DDC and possible evaluation techniques are included as well.

1.3.1 Tools and Software

A variety of programming tools have been used in the analysis as stated in the following:

- The Twitter Application Programming Interface (API) [1] was used to collect the tweets from each user automatically.
- Microsoft's PowerShell framework served as the runtime environment for the Twitter API and performed additional tasks in the data preparation steps, such as removing all non-numeric values from the tweets.
- In the Java programming language, the raw JavaScript Object Notation (JSON) data, retrieved from the Twitter API, was converted to the Extensible Markup Language Metadata Interchange (XMI) format and later converted back to JSON after being used for the classification with DDC.

[1] https://developer.twitter.com/en/docs/twitter-api

- For the automatic classification of tweets based on DDC, the *text2ddc* (Uslu et al. 2019) tool was used, which allows documents to be placed into the classification system libraries and later filtered using the predefined category mapping, which can be found in the Category Mapping section in the Appendix. To execute the classification, using *text2ddc*, the *TextImager*[2] (Hemati et al. 2016) tool was utilized. Both instruments are provided by the *Text Technology Lab*[3] of Goethe Universität, Frankfurt am Main.

- The Python programming language was used for the final execution of LDA, ATM and the posterior evaluation calculations. The library Gensim [4] trained the two topic models and the library pyLDAvis [5] created the plots.

For a more detailed documentation of each entity, see the Software Documentation section of the Appendix.

1.3.2 Topic Modeling

Topic models make it possible to find latent semantic structures in an unstructured corpus of text documents. The technique of topic modeling is based on the assumption that documents consist of a variety of topics. By finding similarity patterns between documents, topics are generated. A topic is then represented as a probability distribution over words (cf. Blei et al. 2003, p. 2; Griffiths et al. 2002, p. 2; Griffiths et al. 2004, p. 5230; Hofman 2001, p. 180). Documents can be of different types: Blei et al. (2003) and Griffiths et al. (2004) for example used scientific abstracts as documents, Wei et al. (2006) applied topic modeling to newspaper archives and Alvarez-Melis et al. (2016) applied it to tweets. The underlying idea of topic models stems from the distributional assumption of linguistics (cf. Firth 1957, p. 18; Harris 1954, pp. 16–18). It hypothesises that words that occur in similar contexts have similar meanings (cf. Turney et al. 2010, p. 142).

The two topic modeling methods used in this thesis are LDA, introduced by Blei et al. (2003), and ATM, introduced by Rosen-Zvi et al. (2004).

LDA is based on a hierarchical Bayesian probabilistic model that assumes a collection of K topics in a document corpus, where each topic represents a multinomial distribution over the vocabulary drawn from the Dirichlet β. The process of LDA for each document d works as follows (cf. Blei et al. 2003, p. 4):

1. Draw a distribution over topics Θ ~ Dirichlet(α)

2. For each word w_n in N:
    a) Draw a topic index $z_n \in \{1,...,K\}$ ~ Multinomial(Θ)
    b) Draw the observed word w_n from the selected topic $w_n \sim \beta_{z_n}$

[2] https://textimager.hucompute.org/rest/doku/

[3] https://www.texttechnologylab.org/

[4] https://radimrehurek.com/gensim/

[5] https://pyldavis.readthedocs.io/en/latest/index.html

As shown in the graphical representation, Figure 1.1, LDA consists of three levels. Level one is corpus-based, with α and β as parameters of the Dirichlet prior of topic distribution per document and word distribution per topic. Level two is document-based, with the variable Θ as the topic distribution for each document. The last level is word-based, and the parameters w_n and z_n are therefore sampled for each word in each document.

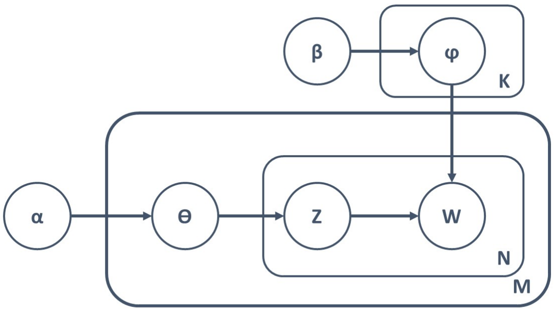

Figure 1.1: Plate model for LDA, reflecting the work of Blei et al. (2003, p. 14). The outer plate represents repetitions, where M is the number of documents in the corpus. The inner plate N represents the number of words in a document and is therefore the repeated selection of topics and words for each document. The outer plate K represents the number of topics set at initialisation.

ATM extends LDA by including information about authorship, whereas each author represents a multinomial distribution over topics and each topic is a multinomial distribution over words. This model does not only demonstrate the documents content, but also the interests of each author. The process of ATM for each document d works as follows (cf. Rosen-Zvi et al. 2004, p. 2):

1. Draw a distribution for each author Θ ~ Dirichlet(α)
2. Draw a distribution over topics Φ ~ Dirichlet(β)
3. For each word w_n in N:
   a) Draw an author x_i
   b) Draw a topic index $z_n \in \{1,...,K\}$ ~ Multinomial(Θ)
   c) Draw the observed word w_n from the selected topic w_n ~ Φ_{z_n}

In Figure 1.2 the plate model for ATM is shown. The model does not only present the topics of an author, but also the general content of each document represented by the topics.

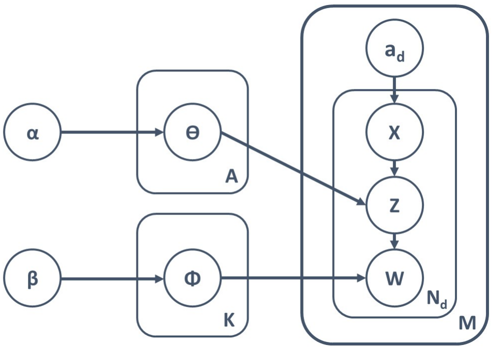

Figure 1.2: Plate model for ATM , reflecting the work of Rosen-Zvi et al. (2004, p. 3). The outer plate represents repetitions, where *M* is the number of documents in the corpus. The inner plate *N* represents the number of words in a document and is therefore the repeated selection of topics and words for each document. The outer plate *K* represents the number of topics set with initialisation and plate *A* the number of unique authors.

Both LDA and ATM use a Bag-of-Words (BOW) model for training. The BOW is a stream of document vectors. The document vectors contain mixture distributions of the individual words based on a vocabulary of known words (cf. Goldberg 2017, p. 69). The vocabulary is retrieved from the entire corpus.

1.3.3 Dewey Decimal Classification (DDC)

The DDC is an internationally used system for subject classification in libraries (cf. Alex 2018, p. 1; OCLC 2019, p. 1). It is used by several projects as a target classification system, for example for the *text2ddc* tool of Uslu et al. (2019) or the *Deutsche Nationalbibliothek (DNB)*[6] to organise knowledge of different forms, such as documents and books, into predefined thematic classes.

The conceptual framework of the DDC contains ten main classes that represent the entire world of knowledge (cf. Alex 2018; OCLC 2019). These classes are:

000 Computer science, information & general works

100 Philosophy & psychology

200 Religion

300 Social science

400 Language

500 Science

600 Technology

[6] https://www.dnb.de/DE/Professionell/DDC-Deutsch/ddc-deutsch_node.html

700 Arts & recreation

800 Literature

900 History & geography

The second level of the DDC divides each class of the first division into ten subclasses, so that DDC in the second level contains one hundred divisions (e.g., the second level of the class ***300** Social Science* contains the classes *300 to 390*, with, for example ***320** Political Science*, ***330** Economy* and ***340** Law*). Each second level class is then subdivided into further ten subclasses, leading to a thousand subdivisions of the DDC in the third level. This allows a more detailed and specific classification of a text corpus when needed. The DDC is published and maintained by the Online Computer Library Center (OCLC). It is continuously developed and expanded to preserve knowledge (cf. OCLC 2019, p. 2).

1.3.4 Evaluation

The evaluation techniques in topic modeling can be either of intrinsic or extrinsic kind.

Intrinsic validation methods evaluate a model without using external information (such as test data) (cf. Palacio-Niño et al. 2019, p. 3). They measure the interpretability of a model. In topic modeling, two intrinsic evaluation methods are topic coherence (Lau et al. 2014; Palacio-Niño et al. 2019) and the calculation of the Evidence Lower Bound (ELBO) of a model (cf. Blei et al. 2003, p. 11).

The coherence of a model assesses the cohesiveness of the topics, consequently how steady the model is when confronted with intrusive words, which reflects the semantic interpretability of the topics (cf. Lau et al. 2014, p. 1). The pipeline of coherence estimation follows four stages, as proposed by Röder et al. (2015):

1. **Segmentation**: Given a word set *t*: Segment it into a set of pairs of word subsets *S*.

2. **Probability Estimation**: Compute the word probabilities *P* based on a given reference corpus (which is the corpus of the model).

3. **Confirmation Measure**: Calculate the agreements ϕ of pairs of *S* by input of the set of words subset *S* and the computed probabilities *P*.

4. **Aggregation**: Aggregate the values to receive a single coherence value *c*

The aim of a topic model is to be as stable as possible and thus to show the highest possible coherence.

The ELBO is an approximation method used to detect the response of a model to unknown values and reflects its ability to represent the statistics of the original data. It is calculated by running the Jensen's inequality on log probability equation on the BOW model of the training corpus (referred to as *corpus*):

$$\log p(x) \geq E_q[\log p(corpus)] - E_q[\log q(corpus)]$$

Maximizing the ELBO leads to finding the parameters giving the tightest bound on the marginal possibility of *x*, as described in Blei (cf. 2011, p. 11) and Jordan et al. (1999, p. 31).

Extrinsic validation methods evaluate a model using additional information (e.g., test data) (cf. Palacio-Niño et al. 2019, p. 6). Standard evaluation measures for unsupervised machine learning algorithms, such as topic modeling, are precision, recall and the F1 score (cf. Sorower 2010, p. 13). However, these calculations require additional information for the evaluation process, as they are based on a classification task that evaluates to which set of different categories an observation belongs. The generated classification information is stored in a contingency matrix containing the following four variables generated from a test data set based on a binary decision of class X and $\neg X$ (cf. Fawcett 2006, p. 862; Palacio-Niño et al. 2019, p. 6; Ikonomakis et al. 2005, p. 973):

- **True Positive (TP)**: The number of values from the class X that have been correctly assigned to the class X.
- **False Negatives (FN)**: The number of values from the class X that have been incorrectly assigned to the class $\neg X$.
- **False Positives (FP)**: The number of values from the class $\neg X$ that have been incorrectly assigned to the class X.
- **True Negatives (TN)**: The number of values from the class $\neg X$ that have been correctly assigned to the class $\neg X$.

In terms of topic modeling, these classifications represent the assessment of a test tweet - it is evaluated whether the correct topic was assigned or not. Given *n* examples, the values of the evaluation measures precision, recall and F1 score for the example of i can then be determined with the help of the following calculations (Ikonomakis et al. 2005):

- **Precision** as the proportion of correctly classified examples out of the total number of examples, which represents the accuracy of the classification:

$$p = \sum_{i=1}^{n} \frac{TP_i}{TP_i + FP_i}$$

- **Recall** as the proportion of correctly classified examples out of the total number of all predicted examples:

$$r = \sum_{i=1}^{n} \frac{TP_i}{TP_i + FN_i}$$

- **F1** as the harmonic mean between the precision score and the recall score representing the performance of the classification:

$$F1 = 2\frac{pr}{p + r}$$

Depending on the desired outcome and the use of the topic model, the evaluation methods used may vary. As this paper undertakes a comparative analysis of different types of topic models, all of the methods presented above are used.

2 Related Work

Topic modeling has been applied in many research papers to improve its performance in micro-blogging short text environments. In particular, LDA, developed by Blei et al. (2003), and its extension ATM, developed by Rosen-Zvi et al. (2004), are widely used to compare the modeling of tweets. Yang et al. (2018), for example, computed the LDA topic model to explore the hidden topics of a large set of documents. They were able to identify the main topics of tweets, proving the explanatory power of LDA for Twitter data. However, there are challenges associated with using the standard topic modeling algorithms, such as computational time and the sole fact that the models were developed to work with sufficiently larger text documents than tweets to produce robust statistics. While working with short text documents works, the risk that the topics are hard to interpret and not as meaningful as with larger documents persists (cf. Alvarez-Melis et al. 2016, p. 1). Consequently, there have been several attempts to enhance the topic models. Chang et al. (2009), for example, have introduced *Nubbi*, a probabilistic topic model that extends LDA. By deriving descriptions of entities and the relationship between those entities, *Nubbi* can build more powerful predictive models by discovering richer descriptions of relationships. Another extended approach to LDA discovers groups between entities and the topics of the corresponding texts, all at once. This model was proposed by McCallum et al. (2006). Liu et al. (2009) and Nallapati et al. (2008) have both improved LDA by integrating it into community structures based on an author-community discovery (cf. Liu et al. 2009, p. 1) and a mixed membership block stochastic model (cf. Nallapati et al. 2008, p. 1). By extending ATM with all the additional auxiliary information of a tweet, Lim et al. (2016) could also achieve better results.

As Rosen-Zvi et al. (2004) found that ATM outperforms LDA the comparison between the two in an environment like Twitter is interesting. Based on that, several papers have tried comparing the two techniques. In the research paper of Hong et al. (2010) LDA and ATM both were trained with Twitter data pulled from 274 accounts that were selected based on 16 categories. The LDA model was trained in three different ways of aggregating the users and then compared to ATM based on precision, recall and the F1 measure. They found out that ATM does not guarantee better results when LDA is trained on user aggregated profiles.

Steinskog et al. (2017) created corpora based on hashtag and author aggregation, which results in a better coherence score. However, they do not compare the direct results between LDA and ATM and try to only enhance the coherence score for both, while evaluating their results based on human judgement.

Another approach of comparison has been made by Alvarez-Melis et al. (2016) that compares LDA and ATM based on four different pooling techniques: tweet pooling, user pooling, hashtag pooling and conversation pooling. In evaluation, they experienced ATM outperforming LDA on all schemes.

Even though other papers (Alvarez-Melis et al. 2016; Hong et al. 2010) have classified their input data by choosing accounts based on categories, it has never been verified if the corpus solely contains documents of these categories. It is questionable whether one can rely on an account always tweeting according to its category only. By incorporating the classification tool *text2ddc* it will be checked if the performance and interpretability of the results of LDA and ATM can be improved without using other pooling or aggregation techniques besides the standard tweet pooling, which captures tweets without any form of aggregation.

An additional challenge that comes with Twitter is the uncertainty of linguistic correctness of tweets. Eisenstein (2013) highlighted the lexemic problems, that might come with social media texts, trying to find a way to improve the bad use of language, popular in social media, using NLP tools. Baldwin et al. (2013) executed a similar study and proved that NLP techniques, like language identification, lexical normalization and part-of-speech tagging can be helpful in reducing the noise that comes with social media texts.

By including the NLP features lemmatizing and stemming and part-of-speech filtering it will be observed if the result can be upgraded. It is not evident if these NLP techniques have been incorporated in the past studies, introduced in this section, using Twitter data. They have, however, been used in other research papers using topic modeling (e.g., (Schmiedel et al. 2018)). Therefore, it will be examined whether this can additionally improve the results.

3 Data Phase

Topic modeling, as a subtype of text mining, analyses text content automatically (cf. Schmiedel et al. 2018, p. 3). Due to this feature, traditional approaches of content analysis characterised by four basic phases can be applied. For further understanding of the applied approach, these four phases will be described.

During the *Data Collection*, the sources of the information are specified. The first step is to define the document types that will be used later. This is done in order to obtain the type of texts that fit the previously defined framework and correspond to the purpose of the research (cf. Duriau et al. 2007, p. 13; Weber 1990, pp. 15–22). Topic modeling requires a large corpus of documents. Therefore, it is advisable to collect the data by machine (cf. Schmiedel et al. 2018, p. 3).

In the *Coding Phase*, the code is created, tested and implemented for later use (cf. Duriau et al. 2007, p. 15; Weber 1990, pp. 22–25). The standard codes of topic modeling are unsupervised machine learning algorithms. They follow an exploratory approach in which the models attempt to identify the association of words to topics as well as the topical category for each document (cf. Quinn et al. 2010, p. 5).

Next, in the *Analysis of Content* phase, different methods are used to examine the data. These approaches can be, for example, frequency counts or cross tabulations in combination with qualitative descriptions of data trends and topics (cf. Duriau et al. 2007, p. 16). The need to adapt the methods to the requirements of the research must be taken into account by selecting the right techniques (cf. Weber 1990, p. 41). Topic modeling content analysis also follows these quantitative assessments by producing summary statistics based on document metadata and interpretation based on associated documents and words (cf. Quinn et al. 2010, p. 4).

The last phase includes the *Interpretation of Results*. The interpretation framework can include measurements, meta-descriptions or inferences and depends on the requirements of the research (cf. Duriau et al. 2007, p. 17). In topic modeling, interpretation can be done through various statistical approaches such as component analysis, clustering and the like (cf. Debortoli et al. 2016, pp. 9–11).

In what follows, the *Data Collection* phase is represented by the Data Collection section, which is divided into two parts. The General Framework characterises the basis of the project. This includes both the purpose of the data and the exact specifications. The Tweet Retrieval describes the process of data collection.

The *Coding* and *Analysis of Content* phases are combined into one section and divided into smaller subsections. At the beginning, the possible classification of the data by the *text2ddc* will be evaluated. For this purpose, a subset of the data will be classified in order to subsequently evaluate whether the approach is useful and to what extent the amount of data changes. The different data preparation processes are described and the techniques used for data cleaning are considered. Finally, the concluding phase, Coding, which explains the different inputs created, how they are processed and includes an overview of the number of tweets before and after classification and cleaning, as well as the final calculation of the models, follows.

Due to its complexity, the *Interpretation of Results* phase is presented as a separate chapter where the models are evaluated based on coherence, ELBO, precision, recall and F1 score.

3.1 Data Collection

This section provides an overview of the data collection framework and an explanation of the approach.

3.1.1 General Framework

The data for the models is extracted from Twitter. The functionality of LDA and ATM is to be tested in short text environments, therefore tweets are used. A tweet is a post by a user and has a character limit of 280 [1].

The data set used is compiled from tweets of a set of users that are assigned to thematic categories. The selection of categories is based on the 16 categories of Hong et al. (2010, p. 4), but has been reduced and to only 9 categories: *Books, Business, Charity, Health, Politics, Science, Shows, Sport* and *Technology*. These categories were adjusted accordingly so that a clear allocation to the DDC classes could be made. In addition, categories were removed due to the lack of availability of accounts. The amount of data should be evenly distributed, so the number of accounts for each category was set to 25.

The selection of accounts was based on predefined rules that had to be fulfilled mandatorily. They concerned the number of followers, the number of tweets and retweets per day and the time window in which the tweets were collected. Based on the approach of Hong et al. (2010) and its adaptation by Alvarez-Melis et al. (cf. 2016, p. 2) the search was designed to target users with an influence in specific categorical areas. Thus, the accounts must correspond to one of the nine predefined categories.

In order to be able to define the criteria for the accounts, the term influential user (hereafter referred to as influencer) in social media should be explained. The term influencer can be replaced by opinion leader. In general, opinion leaders can be classified as members of a group with a strong personal influence on other members of the group. With their influence, they shape the opinions of other members (cf. Meffert et al. 2018, p. 120) and thus also influence their decisions (cf. Flynn et al. 1996, p. 137). People who are influential on social media platforms are called social media influencers. They create and distribute content of a certain category on a certain platform and influence the opinions, purchases and decisions of their followers with their posts (cf. Campbell et al. 2020, p. 469; Lou et al. 2019, p. 58). This requires not only followers, but also constant exchange through posts and interactions (referred to as engagement).

[1] `https://help.twitter.com/en/using-twitter/how-to-tweet`

Based on this information, the lower limit was set to 5.000 followers per account, with no upper limit. This number is the limit between normal users and the smallest form of influencers, the micro-influencers (cf. Andrae et al. 2019). A further specification is the number of posts per day, which has been set at a minimum of two per day. This includes a users' own tweets and the users' retweets of other users' posts. Taking these rules into account, it can be assumed that a certain level of engagement is generated.

The data collection period was set to seven days. No major events (e.g., the outbreak of a pandemic or a plane crash) were to take place during this period to prevent the majority of accounts from only tweeting about this issue rather than their usual topics. With this in mind, the week of 22.02.2021 to 28.02.2021 was chosen.

In order to find accounts belonging to one of the nine predefined categories, different approaches were taken. In the best case, a list with a ranking of German influencers from this category existed. If there was no such list, people from this category in Germany were searched for on Google or the Twitter algorithm was used to get similar accounts suggested. For the categories for which there was no clear list, more than 25 accounts were selected if possible and later the final accounts were drawn at random. The exact methodology and an overview for each category is provided in the Category Mapping section in the Appendix. By categorising the accounts, an attempt was made to create a coherent corpus, which will be further improved in the follow-up processing with the *text2ddc*.

3.1.2 Tweet Retrieval

To retrieve the tweets, the Twitter API Academic Research product track[2] was used. This allowed the collection of all tweets posted by users and all tweets in which they were mentioned in the selected time period.

It would have also been possible to use the Standard Twitter DEV account[3], but it is limited in the number of tweets and mentions that can be collected. This is problematic as potentially not all data can be retrieved in this way. Also, the time period of the query cannot be chosen by the user. It is limited to the last 3200 tweets and 800 mentions. If the time period from which the data is needed is further in the past, so that more than 3200 tweets and 800 mentions have been posted since then, the data cannot be collected using the Standard account.

The data retrieval was automated by using a PowerShell script. A total of 549.853 tweets were collected using this method, with most tweets belonging to the *Politics* category and the fewest to the *Books* category. The overview of the retrieved tweets per category, sorted by amount, can be found in Figure 3.1.

[2] https://developer.twitter.com/en/products/twitter-api/academic-research

[3] https://developer.twitter.com/en/products/twitter-api/standard

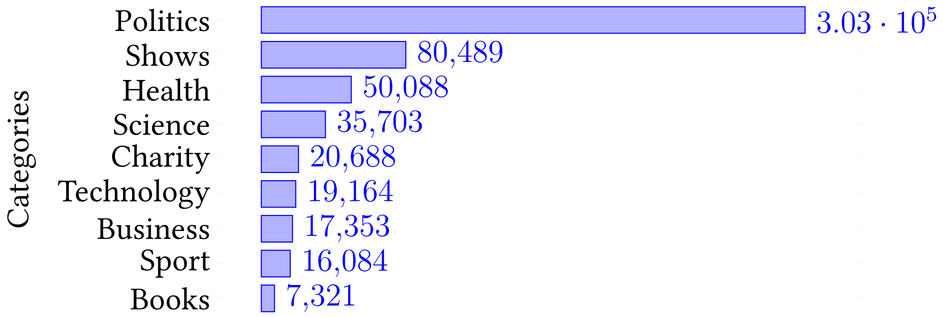

Figure 3.1: Overview of the individual categories with the respective number of tweets retrieved, sorted by descending size.

3.2 Data Preparation

In this section, the previously collected data are processed. First, the classification by the *text2ddc* is discussed, and then the additional steps needed to create a good corpus for topic modeling is presented.

3.2.1 DDC Classification

In order to assess whether the topics of a corpus are consistent and only include tweets that match the categories, a classification of the raw data is performed by using the *text2ddc* classification system. This should improve the performance and interpretability of the results obtained by all the models. However, data loss must also be expected as a result of the classification, as tweets are discarded because they do not belong to the predefined DDC classes. To estimate the extent to which this data loss occurs and whether the loss of this amount of data can be accepted, a classification is simulated with randomly selected test tweets and evaluated by a distribution measurement. Depending on the result, the procedure must be changed, for example by increasing the time period in which the tweets are collected.

As explained in the Dewey Decimal Classification (DDC) section, the DDC uses a class-based system that sorts the input texts into classes of a defined level of the DDC. The level used for this work is the second level, except for the categories *Sport* and *Shows*. These are placed into the third level of the DDC because they would both have the same target class in the second level. The individual DDC classes have been evaluated and, where possible, mapped to each of the nine categories. The results of this can be observed in the Category Mapping part of the Appendix.

The tool used to perform the DDC classification was the *text2ddc* (Uslu et al. 2019). The classification has been started using the *TextImager* [4] (Hemati et al. 2016) as a process. For this a XMI file is read in and then output, tagged with the DDC class labels. This file contains the respective mapping with the percentage distributions for each recognised DDC class for each classified text document. The classes are listed and sorted by the size of the distribution.

[4] https://textimager.hucompute.org/rest/doku/

Each input contains not only one DDC class, but all possible ones that could be the topic of the text. How many classes are listed depends on the text analysed and its content.

For each of the nine categories (defined above), 1000 test tweets were separately and randomly selected from the entire tweet pool for each category. These tweets were classified using the *text2ddc.* The results were then used to create a distribution map of the DDC classes for each category, to then determine the proportion of tweets that thematically belong to each category.

The distribution of tweets that thematically belong to a DDC class was calculated as follows:

$$\frac{Number\ of\ all\ assignments\ of\ DDC\ class\ X_i}{Total\ number\ of\ sampled\ tweets}$$

The value was calculated for the total of 1000 tweets of each category to show the thematic distribution of the respective corpus. In Table 3.1, the percentage of the correctly mapped DDC classes, denoted as *X*, to the nine target categories can be examined.

	Books	Business	Charity	Health	Politics	Science	Shows	Sport	Technology
X	20.85%	22.45%	20.65%	14.23%	17.3%	5.2%	17.08%	33.9%	13.12%

Table 3.1: Overview of the distributions of the DDC classes that were assigned to the target categories as thematically correct, based on the multi-label classification.

The table shows that the proportions differ strongly.

The category *Science* is the category with the lowest representation. Looking at the detailed distribution for this category (Figure 3.2), it is obvious that only the class *520 Astronomy* is represented and the podium places are split between classes such as *330 Economics* or *620 Technology.* This could be due to a poor corpus of the 1000 test tweets, but it could also be possible that the accounts belonging to the *Science* category only tweeted fewer science related tweets in the selected time period.

One of the stronger distributed categories is the *Business* category. Within this category the subclass *330 Business* is the strongest and thus corresponds to the topic that the tweet corpus should fulfil, as can be seen in Figure 3.3. Nevertheless, there are also classes in this category that have a strong thematic representation, even though they aren't thematically equivalent to the business topic. Additionally, there are classes for which it makes sense that they are represented in the *Business* corpus. These include, for example *300 Social Sciences, Sociology and Anthropology* and *150 Psychology.* It can be assumed that these topics are also covered by business influencers, as social management is an important part of a business. However, these DDC classes were intentionally not included in the predefined target classes, as the *Business* category is not supposed to be about the social but about the economic enterprise. This reduction of the corpus was therefore to be expected.

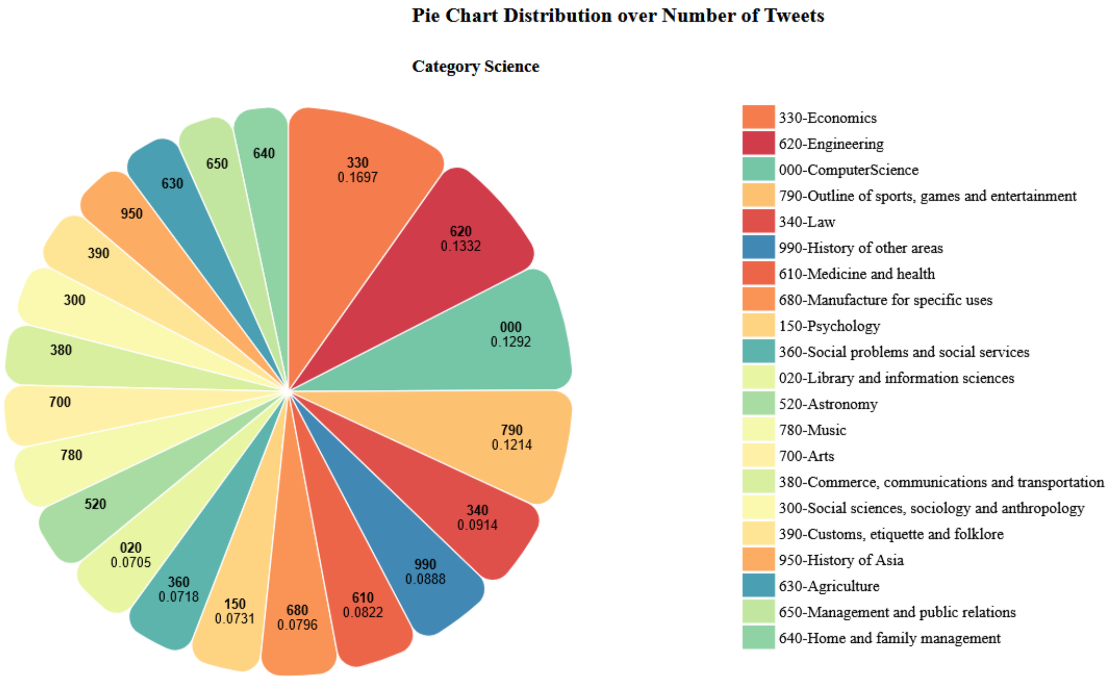

Figure 3.2: Representation of the DDC class distribution for the 1000 test tweets of category *Science*.

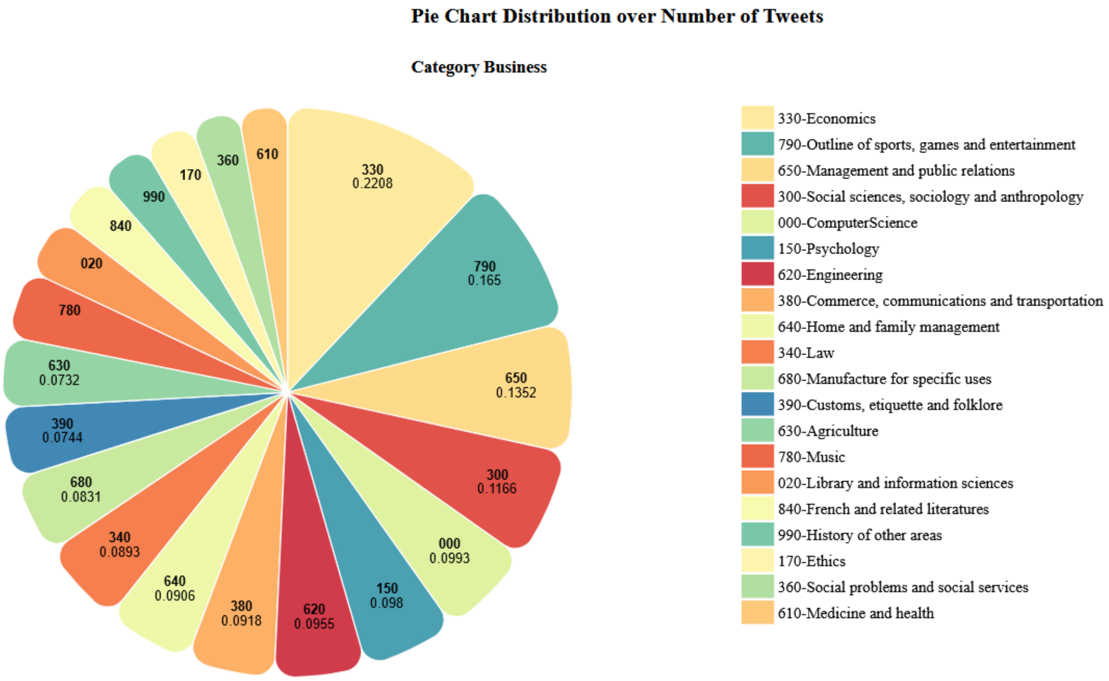

Figure 3.3: Representation of the DDC class distribution for the 1000 test tweets of category *Business*.

Performing this preprocessing step had the following advantage. The amount of tweets identified by the *text2ddc* classification and the associated loss of data can be analysed in detail, so that the procedure can be adjusted in advance. When the results of the distributions are combined, an average of 18.3% of the tweets are expected to remain from the original corpus. Without adjustment, this would be only 100.623 of the original 549.853 tweets. The analysis confirms how crucial it can be to classify a corpus thematically before using it.

The reduction found is significant, so the procedure for processing the tweets changes. Instead of using the categorisation of the accounts and basing the classification of the tweets on this preselection, the corpus containing the entire set of tweets is classified as one unit. The preselection of accounts therefore becomes irrelevant for the part of the *text2ddc* classification. This still reduces the number of tweets, but to a lesser extent, as tweets that do not belong to the account's original category can be made use of additionally.

3.2.2 Data Cleaning

The process of data cleaning attempts to find and remove errors and inconsistencies in data. The aim thereby is to improve the quality of the data by removing errors that occur in data collections. These errors can be of various origins, such as spelling mistakes or grammatical errors. The main problem that data cleaning aims to solve is the removal and correction of invalid data of any kind. The process should identify the most important errors while reducing the need for manual observance (cf. Rahm et al. 2000, 3f.).

The following provides an overview of the standard data cleaning and preparation steps that transform unstructured text data into qualitative text data, as described in Schmiedel et al. (2018, p. 9) and Miner et al. (2012, p. 4).

- **Transforming Document Formats**: Converting of the format of the texts into the format needed for the later processing. For example, data that is in JSON format must be converted if a different data format is needed for further processing. The application of this step in topic modeling depends on the topic modeling toolbox used and which input format it accepts. Depending on the format in which the data has been obtained, this step must be applied and is therefore used if necessary.

- **Constructing Metadata Attributes**: If the research needs metadata variables from the corpus, these should be extracted. This can be the case in different environments, for example, when obtaining data through conversation pooling in Twitter, where tweets that have the same conversation ID are merged. This step is also performed, only when necessary.

- **Removing Duplicates**: Removal of duplicate documents to avoid biases. Must be carried out to avoid possible bias in the results of the topic models.

- **Tokenization**: Tokenization describes the process of splitting documents into individual sentences and these sentences into individual words. This process is especially important for the later training of the topic model, since it is based on a BOW model. This step must therefore be performed.

- **Stop Word Removal**: Removing non-informative words that do not contribute to the meaning of the sentence. There are lists that contain the default stop words for different languages. This clean-up prevents topics from being generated that contain non-informative words (such as *the, and, ...*) and thus contributes significantly to the subsequent quality of the model. The implementation is strongly recommended.
- **Normalization**: Describes the process of converting upper case letters into lower case letters. This step contributes significantly to the quality of the model, as it prevents a word from occurring in two different spellings in one topic, thus preventing bias. The implementation is recommended.
- **Part-of-Speech Filtering**: Describes the process of filtering the text so that it contains only those parts of speech that have a significant impact on sentence meaning. For example, pronouns could be removed as part-of-speech. Thus, from the corpus, person pronouns (such as *I, you, he, she, it, ...*), among others, would be eliminated. However, it is important to note that stylistic information may be lost by removing certain part-of-speeches. The implementation is optional.
- **Lemmatizing and Stemming**: Describes the process of reducing individual words to their root forms. For example, the stemmed form of the word *reading* is *read*. The suffix *-ing* has been removed, generating the root of the word. However, the aggressive use of lemmatizing and stemming risks a loss of information in the meaning of the word. The implementation is optional.

There are other cleaning steps that should be applied specifically to social media data, as the short texts are noisy and may contain various attributes that make processing difficult. One of these steps is to remove non-numeric values, such as emoticons, and to remove short words of less than 3 characters to avoid pure affirmations (such as *Yes*, *Ok*). In addition, all URLs should be removed as they cannot be processed and increase noise in social media texts. All references to other users, so-called mentions, should be removed as they are not carriers of meaning but direct interactions. The language of texts should be limited to one and all texts that do not comply should be sorted out. All documents with less than three tokens should be discarded, as they cannot provide qualitative added value to the topic models.

To decide which of these steps to apply to a project, it is necessary to understand which parts of the text corpus are relevant and useful. Based on the understanding of the data, it can then be decided which parts of the data will be used for the topic model and which parts of the data will be filtered out through the data cleaning process. In general, it is recommended to clean the data to improve the quality of various models. The use of the cleaning steps for the corpora involved is defined and explained in the following.

3.3 Coding

3.3.1 Notations

For later graphical evaluation use, notations are to be defined in the following:

- *DDC* will refer to the training corpus classified by the *text2ddc*.
- *NonDDC* will refer to the training corpus not classified by the *text2ddc*.
- *Words* will refer to the unpreprocessed training corpus.
- *POS* will refer to the training corpus preprocessed by a part-of-speech filter.
- *Stem* will refer to the training corpus preprocessed by lemmatizing and stemming.
- *POSStem* will refer to the training corpus preprocessed by lemmatizing and stemming and a part-of-speech filter.

The notations are combined with the labels of the two topic models used, LDA and ATM. For example, the notation *NonDDCLDAPOS* denotes the LDA model trained on the unclassified corpus with preprocessing of the data by part-of-speech filtering.

3.3.2 Processing of the Data

The following section describes the processing of the data collected during data collection. This includes the implementation of the DDC classification and data cleaning steps and how they are applied, the overview of the different input variables for training the topic models, and how the training of the models is carried out. The coding section contains the following processing steps of the data, listed in the order of work: Data cleaning, the *text2ddc* classification, the division of the data into training and test sets and the training of the models.

The first step that needs to be performed is to clean the data. The tweets have already been pulled in Part 3.1.2 via the Twitter API. By default, the data is retrieved in JSON format. Subsequent training of the models is done with a JSON data input, which means that the document formats do not need to be transformed.

The construction of metadata attributes is also not performed, as there is no specific research question for which this would be necessary.

The tweets come from predefined users. It is unlikely that they will create tweet duplicates. Duplicates could, however, appear in the form of retweets. Multiple accounts could retweet a particular tweet from another user to show their agreement with the topic addressed in the tweet. Since not only the retweet was pulled via the Twitter API, but also the original tweet that was retweeted, it is essential to check whether it is included more than once and remove it if necessary.

In order to remove the stop words, it is first necessary to determine exactly which words are to be removed. There are several ways to identify these words. For some languages, libraries of individual programming languages exist that contain functions for removing stop words. The functions maintain lists containing the words for each of the languages covered. An example of this is the Python package *nltk.corpus*[5]. Another approach is to search online for lists of stop words. These lists can be used independently so that words can be filtered from the list or specific words can be added to the list. For this work, a stop word list was used that was compiled from various online lists[6][7][8]. The words were then removed by a PowerShell script.

To normalise the data, a PowerShell script was used as well, which replaces all upper case letters with lower case letters.

Part-of-speech filtering and the lemmatizing and stemming step can be considered special cases, as they are optional preprocessing steps. In similar papers (Alvarez-Melis et al. 2016; Hong et al. 2010; Steinskog et al. 2017) both steps have not been used, or it is not visibly documented. The aim is therefore to determine whether their use has an impact on the results and significantly improves or worsens them. For this purpose, the corpora are preprocessed in four different ways: Neither by part-of-speech filtering nor by lemmatizing and stemming, by part-of-speech filtering only, by lemmatizing and stemming only, and by part-of-speech filtering and lemmatizing and stemming. For both part-of-speech filtering and lemmatizing and stemming, the tagging function of the *text2ddc* (Uslu et al. 2019) could have been used. However, this would mean that the tagged corpus that is not to be classified would have to be processed by *text2ddc* unnecessarily. As this is associated with a high programming effort due to the further processing of the data and the transformation of the data formats, the decision was made that the processing should take place directly in the code and be carried out during the preparation of the data for training. Again, there are libraries of various programming languages that already contain functions for processing texts with part-of-speech filtering, as well as lemmatizing and stemming. To be able to filter certain part-of-speech types, the Python package *nltk.tag*[9] was used. The function *pos_tag()* implements the possibility to tag an arbitrary string. It generates an encoded tuple of the form *(tag, token)* as a return value. The *tag* is the information to which part-of-speech the *token* (i.e. word) belongs. The filtered part-of-speech words are the WH words. They include WH determiners (like *that, what,...*), WH pronouns (like *what, which, who, whom,...*) and the WH adverbs (like *how, but, whence, when,...*). Since they are not direct carriers of information in a text, they can be removed. Other part-of-speeches could have been removed as well, but the risk of losing information would have been high. In further work, however, it would be very interesting to weed out riskier parts and look at the changes in the models.

Lemmatizing and stemming are done using a Python library called *nltk.stem*[10]. The *Snowball Stemmer*, originally created by Porter (2001), is used. The *Snowball Stemmer* assumes that words follow a uniform structure. This structure implies that words are written from left to right and the root is always on the left. Suffixes are added on the right, which can be removed to obtain the root of a word.

[5] https://www.nltk.org/api/nltk.corpus.html

[6] https://countwordsfree.com/stopwords/german

[7] https://github.com/stopwords-iso/stopwords-de

[8] https://groups.google.com/g/resourcespace/c/g4U5Xia-bX4

[9] http://www.nltk.org/api/nltk.tag.html

[10] https://https://www.nltk.org/api/nltk.stem.html

As social media data requires additional preparation, additional data cleaning steps were applied. All steps described below were implemented using a PowerShell script. Tweets can contain special variables such as hashtags, mentions and links. Both links and mentions are not information carriers, as links cannot be processed and mentions by users also do not contain information about the content of a tweet. Therefore, both have been removed. The Twitter API allows tweets to be pulled with extended object information. This allows information about URLs and mentions to be stored directly in the JSON file of each tweet. This information includes, among other things, the position of URLs and mentions, making it easy to remove them. Hashtags were not removed, but the hashtag character before the actual word was deleted as it is a non-numeric value. All other non-numeric values have also been removed. These include punctuation marks such as hyphens and commas. In the next step, also using Twitter's extended object information, all tweets that did not belong to the German language were removed. In addition, all tokens consisting of less than three characters were eliminated. If a tweet had fewer than three tokens after these steps, it was discarded as well.

After the cleaning of the data was completed, the preprocessing was done through the *text2ddc* classification. Similar to the preparation part, where the randomly selected data was classified to get an overview of the distribution, the data was first converted into XMI format and then processed and tagged by the *text2ddc* classification. The entire corpus was processed as a unit to check for each tweet whether it fits into one of the DDC classes in order to classify it as category-based, regardless of the previously determined categorical account assignment. All tweets that could be assigned to a predefined category were then added to the DDC classified corpus for subsequent topic modeling.

The inputs for the LDA and ATM algorithms are processed once each for the classified corpus and the unclassified corpus. Four different sets of inputs are processed. As previously described, these are:

- All tweets preprocessed with neither part-of-speech filtering nor lemmatizing and stemming.
- All tweets preprocessed with only part-of-speech filtering.
- All tweets preprocessed with only lemmatizing and stemming.
- All tweets preprocessed with both part-of-speech filtering and lemmatizing and stemming.

This results in a total of 8 different corpora, which are further multiplied by the different input variables for the number of topics.

Since three of the later evaluations of the models are based on a document retrieval test with training and test data, it is necessary to divide the corpora into training and test sets. The proportion of data in the training set was set to 80 percent. The proportion of data in the test set was set to 20 percent. The division of the corpora was to be random and uninfluenced. For this purpose, the Python library *sklearn.model_selection* [11] with its function *train_test_split()* was used. This function can be used to randomly split an array as an input value into a training set and a test set. Below is an overview of the final data sets, which includes not only an overview of the number of tweets, but also the number of authors and unique tokens.

As can be seen in the Table 3.2, the number of tweets was reduced to 475.236 after data cleaning. After classification with the *text2ddc*, 145.819 tweets remained. This shows that only 30.6 percent of the data was classified as category-related by the *text2ddc*. This percentage is almost double the 18.3 percent that would have remained if the tweets had only been processed by their account-based category. This proves how important it can be to include classification, as the unassigned 69.4 per cent of tweets have a large impact on the results of the models. To confirm what influence classification can have, the modeling of the classified and unclassified corpus is reviewed.

Dataset	Corpus	Documents	Authors	Tokens
Train	NonDDC	380.186	88.274	210.141
	DDC	116.655	40.028	138.794
		475.236	128.302	348.935
Test	NonDDC	95.050	40.749	95.343
	DDC	29.164	16.857	60.630
		145.819	57.606	155.973

Table 3.2: Data set statistics, including the number of documents, authors and unique tokens in each data set.

The final step of modeling contains training the models. There are several ways to do this. It is possible to create ones' own algorithm based on the framework presented in the papers by Blei et al. (2003) for LDA and Rosen-Zvi et al. (2004) for ATM. However, there are also various online toolboxes or libraries of individual programming languages that can be used. The *Stanford Topic Modeling Toolbox*[12], for example, trains LDA topic models, as well as its extensions Labeled Latent Dirichlet Allocation (LLDA) and Parallel Latent Dirichlet Allocation (PLDA), and was implemented by creating *Scala* scripts. It uses Excel spreadsheets as input data. An example of a topic modeling implementation in a programming language is the R package *topicmodels*[13]. The LDA topic model is implemented as well as the Correlated Topic Model (CTM). For this work, the topic modeling library *Gensim*[14] was used, which was developed in the Python programming language. The advantage of using it is that it implements not only LDA but also ATM. In many other toolboxes or libraries ATM is not included.

Using the library *Gensim* has other advantages as well. Many processing steps can be solved faster because predefined functions are already available. For example, there are func-

[11] https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.train_test_split.html

[12] https://downloads.cs.stanford.edu/nlp/software/tmt/tmt-0.4/

[13] https://cran.r-project.org/web/packages/topicmodels/index.html

[14] https://radimrehurek.com/gensim/index.html

tions to turn a tokenised list into a BOW model or to create a token dictionary. In addition, the library contains several possibilities to apply preprocessing steps for the later evaluation metrics. Thus, it ensures a minimal amount of work in the training phase and simplifies the later evaluation part.

The training of the models proceeds as follows: The algorithm accepts an input file. This file contains the data (tweets) in a format such as JSON. The files are preprocessed by creating a list containing the set of all tweets in the form of a tokenised list. Linguistic preprocessing is performed on these lists, if desired, and the individual words are part-of-speech filtered and lemmatised and stemmed. Next, a dictionary is created. It contains a list of all words with a corresponding ID for each word. The dictionary is used to weed out extreme values of word occurrences. In this way, words that occur in very few documents or words that occur in many documents can be sorted out. If a word occurs in very few documents, it does not contribute to the topics of the overall corpus. The limit has been set to 15, so words that occur in fewer than 15 documents will be discarded. If a word occurs in many documents, it strongly influences the topics of the whole corpus and could potentially be another stop word that went unrecognised. Therefore, words that occurred in 50 per cent of the documents were discarded. Another step in preparing the data for training the model is to process the words into a BOW model, which later represents the input corpus. The concept of BOW has already been explained in the Section 1.3.2. The preparation for the training of LDA models is completed after generating this additional data. For the ATM model, an additional step is required by creating an author map. This contains an assignment of the individual authors to all documents they have authored.

Training the models with *Gensim* requires different input values. At the beginning, the number of topics to be trained should be determined. In the case of this thesis this is done by iterating over a sequence of different topic counts. The sequence contains: 10, 50, 100, 200, 300. In total, 80 models are trained by the five different quantities. Each model was trained over 2000 iterations, with a training chunk of 2000 documents each and 10 runs of the corpus per iteration. These values could of course be changed, but were chosen following Alvarez-Melis et al. (2016, p. 3).

Training the models requires different amounts of time, as can be seen in Figure 3.4. Most of the time is spent training the ATM models on the corpus that has not been classified by the *text2ddc*. It takes significantly more time than its counterpart, training the LDA models on the corpus that has not been classified by the *text2ddc*. Moreover, training the ATM models on the corpus classified by the *text2ddc* takes more time than training the LDA models. It is particularly striking here that the models trained with LDA whose corpus was not classified with the *text2ddc* take only minimally more time to train than the models trained on the classified corpus. The time difference is less than 1 hour, while the difference for the ATM models trained on the unclassified corpus is several hours compared to the ATM models trained on the classified corpus. It is understandable that a larger corpus requires more time to train than a smaller corpus. Since the training time of the ATM models is higher than that of the LDA models when the classified corpus is used, it is self-explanatory that the training time of the ATM models also takes more time than that of the LDA models when the unclassified and thus larger corpus is used. Nevertheless, it is interesting that the difference between the models trained with LDA and ATM is so much greater when using the unclassified corpus than when using the classified corpus. One could assume that the training time of the ATM models increases faster than the training time of the LDA models when the number of documents contained in the corpus increases. Regarding the different preprocessing steps, it can be seen that the models trained with the unclassified corpus usually take longer if either part-of-speech filtering, lemmatizing and stemming or both were applied in combination. The models trained with LDA using the corpus classified with the *text2ddc* show similar behaviour. In contrast, the models trained with ATM do not show much difference in training time and require almost the same amount. The exact results, of the training time, can be viewed in the Evaluation Results part of the Appendix.

However, the validity of this data should be critically questioned. Times may vary when using other computer systems and depend on computing power. It does not necessarily mean that two different systems will achieve exactly the same training times. However, it can be assumed that models generated with the ATM algorithm take longer than models trained with the LDA algorithm.

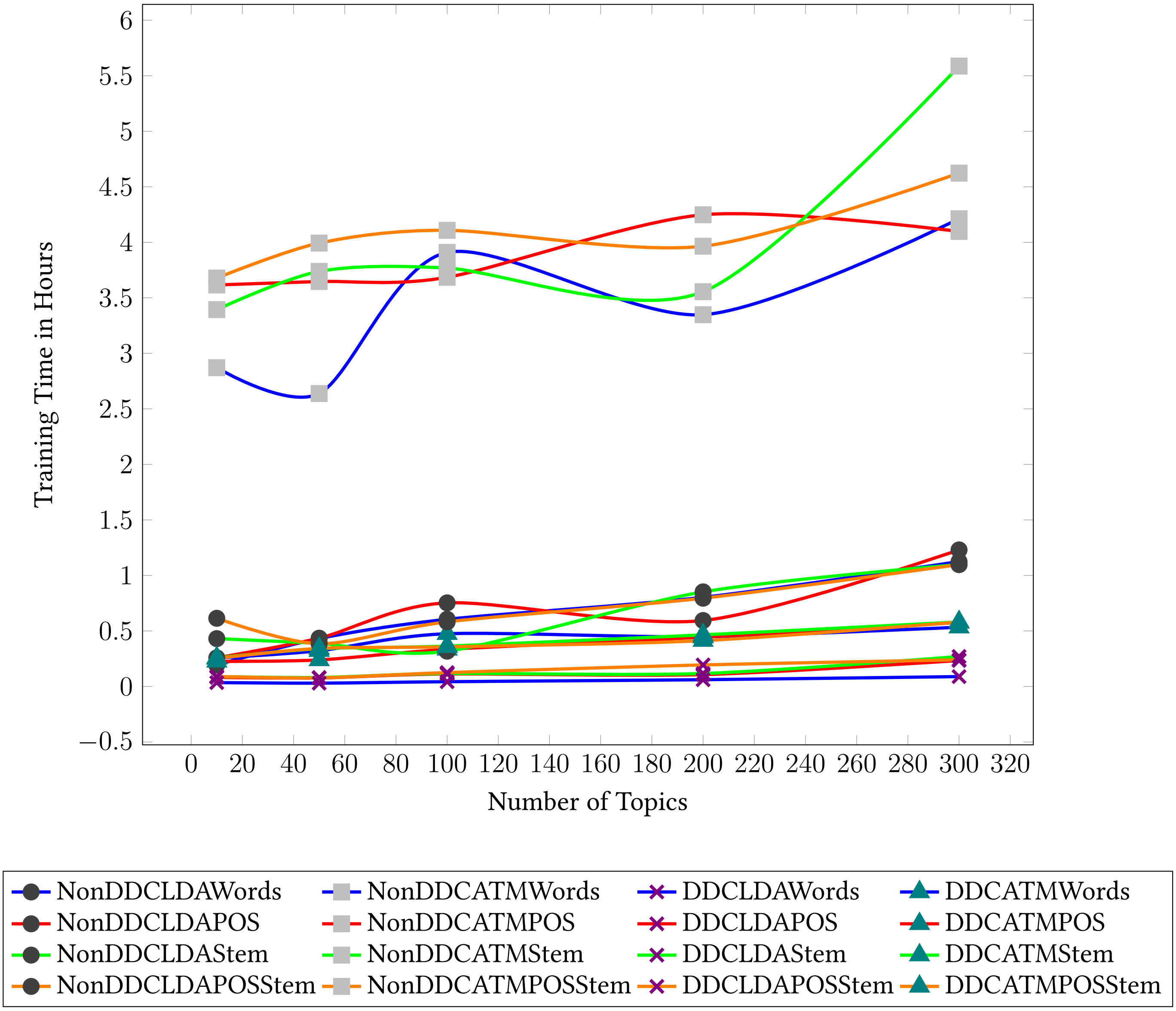

Figure 3.4: Training times in hours.

4 Interpretation of Results

In the following section, the trained models are evaluated. This is done on the basis of five different metrics: coherence, ELBO, precision, recall and F1 score. The evaluation analyses to what extent the topic models have improved by using the *text2ddc* classifier. In addition it tests whether there is a difference between the different models trained on the basis of LDA or ATM and to what extent the results have improved or deteriorated by the clean-up techniques part-of-speech filtering and lemmatizing and stemming.

4.1 Intrinsic Evaluation

In the following, the evaluations of the two intrinsic metrics coherence and ELBO are discussed and analysed.

4.1.1 Coherence

Topic coherence is used to test a simple concept: The co-occurrence in a reference corpus should be taken into account for the top words of the individual topics. The more coherent the topics are, the more stable the models are and the higher the quality of the learned topics. The topics should therefore be evaluated to see whether a model can generate a meaningful output and with which number of topics this output is most coherent. The model that is most coherent should be used for further use of the topic model, as it is the most stable. Coherence can be determined in two different ways. The topics of a model can be judged as accurate or inaccurate either by a human evaluation or by a machine approach, as presented by Röder et al. (2015).

Table 4.1 shows how a topic could be constructed. It represents one of the Topics from the ATM models, preprocessed with a part-of-speech filter and trained on the unclassified corpus. A topic consists of a cluster of words. The words are uniquely determined by distributions. The topic shown most likely covers the political election campaign in Germany. By looking at the individual topics, one can try to understand and interpret the overall topic of a corpus and evaluate whether it covers the expected outcome.

Distribution	Word
0.040	macht
0.026	deutschland
0.015	elternzeit
0.014	wahlkampfhilfe
0.013	wochenenden
0.012	cdu
0.011	heftig
0.010	wahlkampf
0.009	deutsche
0.008	schmackhaft

Table 4.1: Top 10 words of a topic with the respective distributions from 0 to 1.

However, the manual evaluation of all models and topics is labour-intensive and the execution rather unrealistic. In the following, the coherence is therefore calculated mechanically using a score, based on Röder et al. (2015). The larger the coherence score, the better the result.

As can be seen in Table 4.2, the values of the coherence score range in an interval of [-4,-19]. Considering the different corpora with which the models were trained, the ATM and LDA models trained with the classified corpus seem to produce a better coherence score than the ATM and LDA models trained with the unclassified corpus.

Looking at the models of the unclassified corpus individually, it is noticeable that the models trained with LDA can produce minimally better results than the models trained with ATM. For both types of trained models, the models preprocessed by lemmatizing and stemming or by lemmatizing and stemming and part-of-speech filtering achieve better scores than the models not preprocessed or preprocessed only by part-of-speech filtering. This is different for the models of the classified corpus. The achieved scores of the models trained with ATM are significantly better than the scores of their counterparts, the LDA models. However, it can be seen that the models preprocessed by lemmatizing and stemming or by lemmatizing and stemming in combination with the part-of-speech filter can also achieve better scores here than the models without or only with part-of-speech filter preprocessing.

The maximum values of the individual topic numbers are marked in bold italics in the table, so that it is comprehensible which preprocessing step of which model can achieve the best score. As a result, it can be deduced that the classification of the corpus can produce more coherent models. To improve this further, it is recommended to use the ATM model with preprocessing by lemmatizing and stemming or by lemmatizing and stemming in combination with part-of-speech filtering.

	Model	T = 10	T = 50	T = 100	T = 200	T = 300
NonDDC	LDAWords	-9.340	-11.372	-14.169	-16.423	-18.195
	LDAPOS	-9.480	-11.314	-14.325	-16.349	-18.083
	LDAStem	***-6.954***	-10.098	***-13.385***	-16.095	-18.253
	LDAPOSStem	-7.878	***-10.006***	-13.394	-15.985	-18.039
	ATMWords	-10.126	-13.475	-15.248	-16.185	-14.617
	ATMPOS	-10.486	-13.330	-15.372	-16.350	-14.701
	ATMStem	-9.187	-12.228	-14.143	-15.742	***-13.902***
	ATMPOSStem	-8.399	-11.834	-14.267	***-15.570***	-14.009
DDC	LDAWords	-4.658	-10.956	-13.412	-14.808	-18.169
	LDAPOS	-4.939	-10.951	-13.637	-15.264	-18.145
	LDAStem	-4.368	-9.619	-12.471	-14.942	-18.465
	LDAPOSStem	-4.364	-9.938	-12.454	-14.803	-18.749
	ATMWords	-5.006	-9.698	-13.225	-14.823	-13.562
	ATMPOS	-4.510	-9.519	-13.025	-14.912	-13.625
	ATMStem	-4.445	***-8.258***	-11.612	***13.819***	12.482
	ATMPOSStem	***-4.334***	-8.394	***-11.589***	-13.827	***-12.113***

Table 4.2: Coherence score for all models with T as number of topics.

It should be noted, however, that a general interpretation of the models and the topics they contain remains difficult, even though it is possible to assess the coherence of the models via various techniques. In addition, one should consider whether the output of the model matches what is expected and draw samples when identifying coherence as a key basis for evaluating the models.

One way to do this is to look at the relationships between the different topics. They contribute significantly to the quality of the models. The possibility to visualise topic models is provided by the Python library *pyLDAvis*[1]. In the following, a model with a better coherence value and a model with a worse value are presented. The models were both trained on 10 topics based on preprocessing with lemmatizing and stemming and the part-of-speech filter. One of the models was trained on the classified corpus and the other on the unclassified corpus.

The *Intertopic Distance Maps* shown in Figure 4.2 and 4.1 represent the word overlaps within individual topics (i.e., they show how similar topics are). The more similar the topics in a model are, the more difficult it is for the model to make clear decisions and the less stable it is. It is therefore important that a model contains as few overlapping topics as possible. The diagrams additionally show the proportion of topics in the entire corpus. The larger the *Marginal Topic Distribution* and thus the larger the graphical representation of the circle, the more frequently this topic is contained in the corpus. As can be seen, the topics from Figure 4.2 are more widely distributed and have less overlap than the topics from Figure 4.1. This suggests that the model from Figure 4.2 was able to identify more unique topics and is therefore more stable than the model from Figure 4.1. This assumption can be confirmed by the

[1] `https://pyldavis.readthedocs.io/en/latest/index.html`

coherence score of the models, as Figure 4.2 can achieve a better score than Figure 4.1.

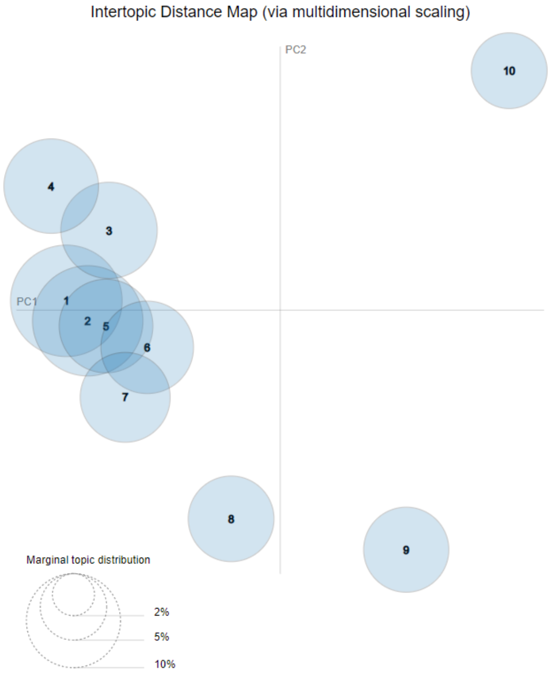

Figure 4.1: Distance Map for model trained on the unclassified corpus.

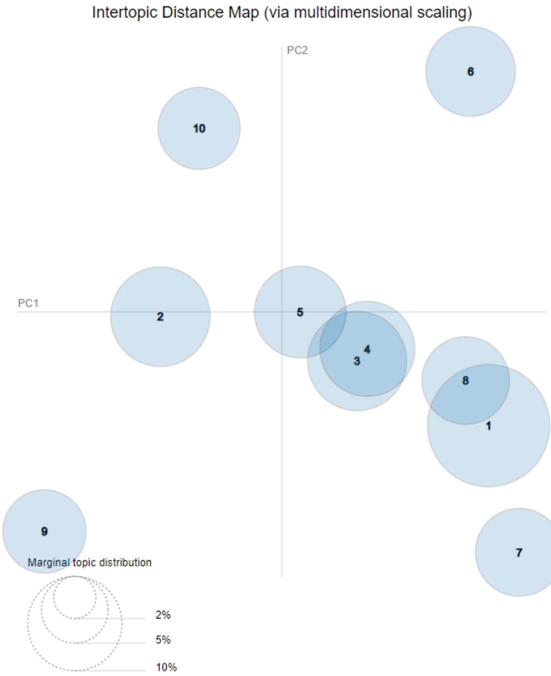

Figure 4.2: Distance Map for model trained on the classified corpus.

The depth of the analysis of the coherence score depends strongly on the goal one wants to achieve with the topic model and should therefore always be made dependent on the research question.

4.1.2 Evidence Lower Bound

As described in Section 1.3.4, the ELBO measures the ability of a model to reproduce the statistics of the original data and thus the consistency of the model in the face of unknown data. Since this score is a maximisation, the larger the calculated score, the better the result.

Regarding the calculated scores of ELBO, which can be seen in Table 4.3, a clear trend can be observed for all trained models. The models trained with the corpus classified by the *text2ddc* all perform better than the models trained with the unclassified corpus. Furthermore, when looking at the two corpora (DDC-classified and non-DDC-classified) individually, it is clear that the models trained with ATM almost consistently perform better than the models trained with LDA.

When evaluating the models trained with the corpus not classified by the *text2ddc*, there are big differences between the models trained with LDA and those trained with ATM. While there are no strong differences for a few topics, the ELBO value deteriorates with increasing number of topics. For example, the scores of the LDA models are more than four times higher than the scores of the ATM models when the number of them is 200.

In general, the ELBO values of the LDA models trained by the classified corpus increase sharply as the number of topics increases. Although the values of the LDA models trained with the classified corpus are also higher than the values of the ATM models in some cases, the difference in the values is not as significant as the difference of the models trained with the unclassified corpus. This could lead to the conclusion that models trained on the basis of a classified corpus achieve better results, regardless of which topic model they were trained on.

Since the models differ not only in the training on the basis of LDA and ATM, but also in the processing steps, an additional look is taken at these differences. It is noticeable that the models perform better when either only lemmatizing and stemming is applied as a preprocessing step or when lemmatizing and stemming are applied in combination with the part-of-speech filter. Pure processing without any of the preprocessing steps performs worse. Using the part-of-speech filter alone gives the worst results, regardless of the corpus and training used.

In terms of the data, this is reflected as well. The maximum value (marked in bold italics in Table 4.3) obtained for a set of topics is in the columns lemmatized and stemmed or lemmatized and stemmed with additional part-of-speech filtering. The individual maximum values indicate a distribution that is not based on ATM models. In particular, when evaluating with a small number of topics, the models trained with LDA using the classified corpus have a minimal advantage. This advantage is not visible with the models of the unclassified corpus, here the models trained with ATM predominate.

In general, these results allow to judge that although individual LDA models have a minimal advantage over some ATM models, it is the ATM models that perform better on average. Moreover, especially the models that have been preprocessed by lemmatizing and stemming as well as part-of-speech filtering can achieve better scores. With regard to the preprocessing of the corpora, the corpus classified with *text2ddc* delivers better results than the unclassified corpus in all respects. Accordingly, it can be assumed that the ELBO score benefits from the use of classification.

	Model	T = 10	T = 50	T = 100	T = 200	T = 300
NonDDC	LDAWords	-9.303	-21.403	-39.508	-159.827	-599.050
	LDAPOS	-9.312	-21.447	-39.722	-162.941	-599.049
	LDAStem	***-8.833***	-18.147	-31.157	-109.669	-539.957
	LDAPOSStem	-8.836	-18.107	-31.155	-109.629	-539.930
	ATMWords	-9.000	-11.997	-15.984	-25.483	-145.972
	ATMPOS	-9.482	-11.991	-16.015	-25.531	-146.179
	ATMStem	-8.900	***-11.397***	***-15.214***	***-24.335***	***-126.213***
	ATMPOSStem	-8.986	-11.398	-15.221	-24.342	-126.169
DDC	LDAWords	-8.530	-10.089	-14.466	-69.156	-459.408
	LDAPOS	-8.531	-10.079	-14.550	-71.490	-432.255
	LDAStem	-8.257	***-9.501***	***-12.516***	-46.879	-310.502
	LDAPOSStem	***-8.256***	-9.528	-12.532	-48.031	-341.517
	ATMWords	-8.673	-10.901	-14.841	-24.750	-100.037
	ATMPOS	-8.663	-10.906	-14.855	-24.807	-100.363
	ATMStem	-8.380	-10.502	-14.201	***-23.526***	***-90.312***
	ATMPOSStem	-8.380	-10.504	-14.213	-23.544	-90.710

Table 4.3: ELBO score for all models with T as number of topics.

4.2 Extrinsic Evaluation

In the following, the evaluations of the extrinsic metrics precision, recall and F1 score are discussed and analysed. Since the metrics can be linked, a correlation between the three scores is to be established.

To calculate the precision and recall scores, the models were evaluated using a document retrieval test with a test data set. Each tweet in the test data set is evaluated as a single query, which is scored based on its similarity to the training tweets. The scores are calculated as follows. For each test tweet, the cosine similarity of its topics and the topics of all training tweets is calculated. Then, for each test tweet, the ten most similar training tweets are determined. Afterwards, it is determined whether the top category of the ten training tweets matches the top category of the test tweet. The evaluation procedure is based on the approach of Alvarez-Melis et al. (2016) and Hong et al. (2010).

4.2.1 Precision

The precision value, which determines the proportion of correctly classified examples in the total number of examples and thus represents the accuracy of the classification, is a stochastic value ranging from 0 to 1. The larger the proportion (i.e., the closer the value is to 1), the better the classification. The graphically represented precision values of all models can be viewed in Figure 4.3.

Looking at the results of both corpora, there is not much difference in the precision scores of the models trained with the classified corpus and the models trained with the unclassified corpus. The results differ only minimally and therefore do not allow for a hard separation in terms of classification. Looking at the topic models trained with the two corpora separately, it is noticeable that the models trained with ATM consistently show a higher precision value. The ATM models trained with the classified corpus show significantly higher values than the trained LDA models. The ATM- models trained with the unclassified corpus are also more precise, but the LDA models can compensate for this with increasing number of topics, as their precision values rise to a similar level as the values of the ATM models. In general, the precision values of the LDA models decrease with increasing number of topics. However, this changes with large number of topics (300 in the case of the trained models), above which the values increase again. The values of the ATM models decrease steadily with increasing number of topics. Thus, the highest values can be reached with the smallest number of topics. Nevertheless, it is worth noting that even the worse scores of the ATM models outperform the good scores of the LDA models.

The performance differences for the different models trained by the various preprocessing steps are distinct. For the LDA models trained with the unclassified corpus, the models preprocessed by lemmatizing and stemming or by lemmatizing and stemming combined with part-of-speech filtering achieve the best results. The ATM models of the unclassified corpus, on the other hand, achieve the best results when the corpus has not been preprocessed or has only been preprocessed by part-of-speech filtering. When using the classified corpus, one of the LDA models shows a large lead over the other models. The model trained with the unprocessed corpus performs significantly better than the preprocessed models. The ATM models, on the other hand, as with the unclassified corpus, perform best when the data has not been further preprocessed or when the corpus has only been modified by the part-of-speech filter.

Ultimately, however, the following conclusion can be drawn. Even if the results partly differ, it is evident from the results that the ATM models can achieve significantly better results than the LDA models. The classification of the corpus by the *text2ddc* does not change the results significantly. The preprocessing of the data also has no significant influence on the results and could therefore be neglected. The exact results, of the precision score can be looked up in the Evaluation Results part of the Appendix.

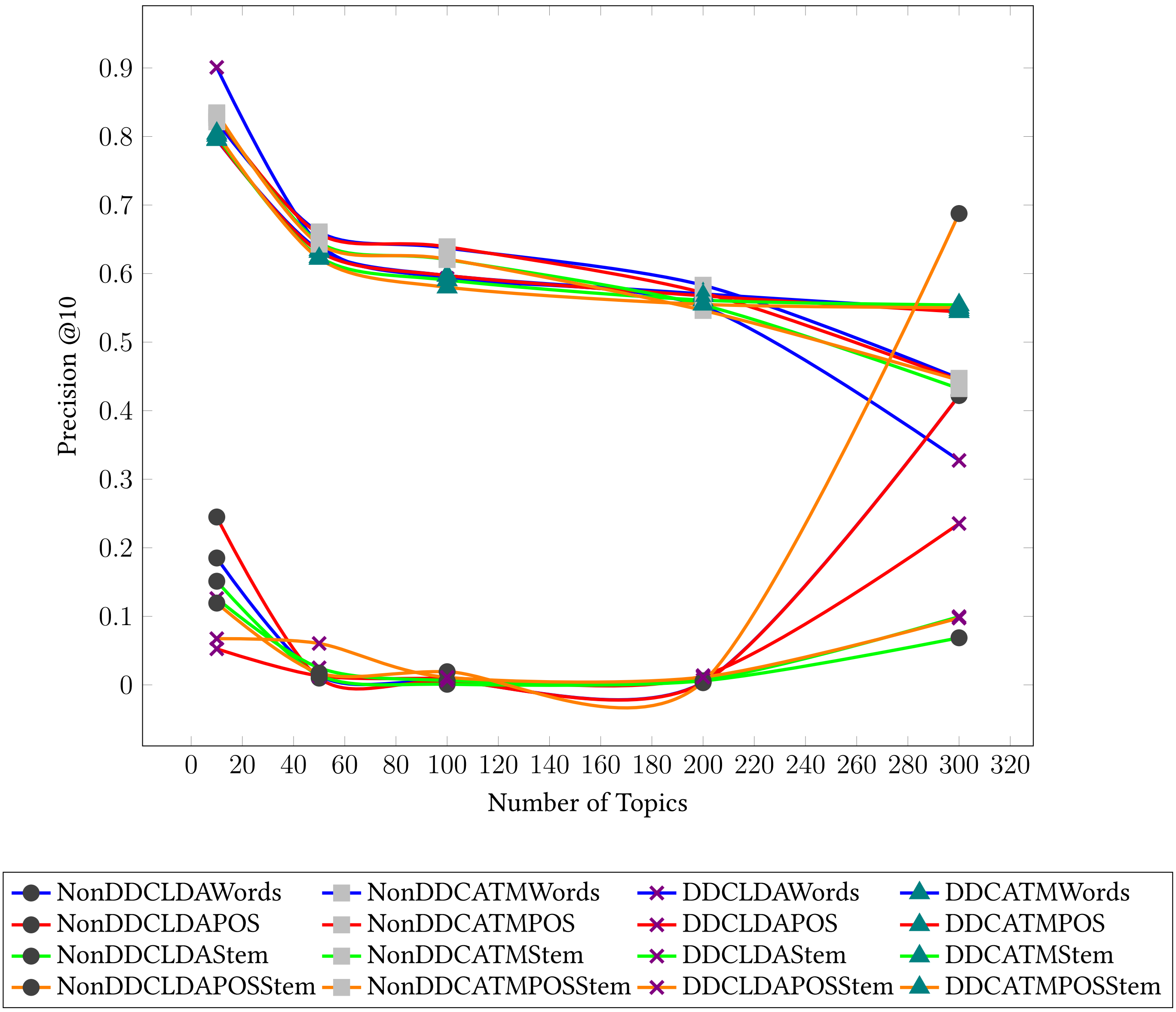

Figure 4.3: Measured precision at 10.

4.2.2 Recall

The recall value, which determines the proportion of correctly classified examples out of the total number of all predicted examples, is a stochastic value ranging from 0 to 1. The larger the proportion (i.e., the closer the value is to 1), the better the classification. The graphically represented recall values of all models can be viewed in Figure 4.4.

With respect to the models trained with the two corpora, there is a clear advantage of the ATM models trained with the classified corpus over the ATM models trained with the unclassified corpus. The recall values of the ATM models trained with the classified corpus are almost twice as high as the values of the models trained with the unclassified corpus. The LDA models trained with the classified corpus appear to have a minimal advantage over the LDA models trained with the unclassified corpus. However, there is one exception. The LDA model trained with the classified corpus and containing no further preprocessing (referred to as *DDCLDAWords*) performs significantly better than the LDA models of the unclassified corpus and also than the other LDA models trained with the classified corpus. In general, there is a trend that the ATM models, regardless of the corpus, lose value with increasing number of topics. However, these losses are minimal. The LDA models show high values when the number of topics is minimal, which then decrease, but increase again when the number of topics is large.

When looking at the preprocessing steps, slight differences between the versions can be observed. When analysing the models generated with the unclassified corpus, it is noticeable that for the models LDA and ATM the best results can be achieved with part-of-speech filtering or without preprocessing. The models that perform worst are those that use a combination of part-of-speech filtering, lemmatizing and stemming. On the other hand, the models trained with the classified corpus can achieve the best recall scores when no further preprocessing step has been applied. They also give the worst values when the combination of part-of-speech filter and lemmatizing and stemming is used. However, it is worth noting that these are minimal variations between evaluations.

When looking at the individual topic levels, it can also be seen that the ATM models consistently perform better than their counterparts, the LDA models, regardless of the number of topics. Overall, this suggests that the models trained with ATM generally achieve better recall and can thus correctly classify a higher number of test inputs. Furthermore, there is a clear advantage of the ATM models trained with the corpus classified by *text2ddc*. This allows the claim that the DDC classification can have a positive influence on the results of an ATM model. The classification does not seem to have a sufficient influence on the results of the LDA models to positively change the recall values. The exact results, of the recall score can be looked up in the Evaluation Results part of the Appendix.

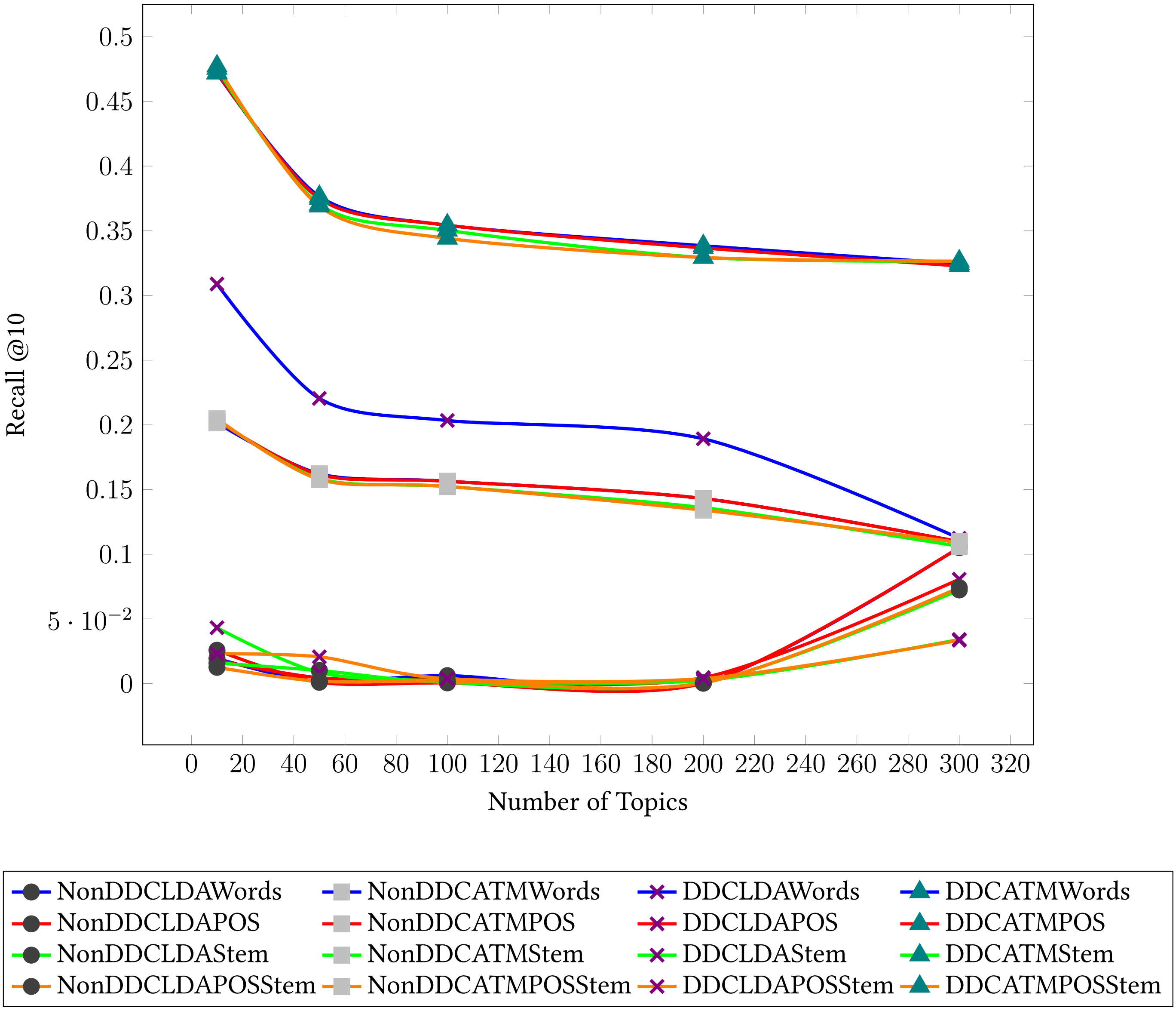

Figure 4.4: Measured recall at 10 based on a scale of $\times 10^{-3}$.

4.2.3 F1 score

The F1 score, which represents the performance of a classification, is a stochastic value that ranges from 0 to 1 and represents the balance between precision and recall for a model. The larger the proportion (i.e. the closer the value is to 1), the better the classification. The plotted F1 scores of all models can be viewed in Figure 4.5.

The models present the evaluation of the results of the F1 scores in a similar way as the evaluations of the recall scores presented in 4.2.2. There is a clear dominance of the corpus models trained with ATM and classified with *text2ddc* over all other models. The ATM models trained with the unclassified corpus also perform well. Here they perform better than the LDA models trained independently of the corpus. It is logical to conclude that the ATM models outperform the LDA models in all respects.

As mentioned above, the ATM models trained with the classified corpus can improve the ATM models trained with the unclassified corpus. This is different when looking at the trained LDA models. In particular, for a small number of topics, the LDA models do not differ in their performance scores. Only a minimal superiority of the models trained with the classified corpus can be seen. However, this is vanishingly small, so that they tend to meet at the same level. This tendency turns when the number of topics reaches 200. At this point, the result set splits. Some LDA models are able to catch up with the lead of the unclassified ATM models. Again, the models preprocessed by the non-preprocessed corpus or by the corpus preprocessed only by part-of-speech filtering are able to achieve better scores. This trend can be observed not only for the ATM models but also for the LDA models. As already shown for Precision and Recall, the LDA trained with the classified corpus can achieve better results. Thus, the model itself can even outperform the ATM models trained with the classified corpus. The exact results, of the recall score can be looked up in the Evaluation Results part of the Appendix.

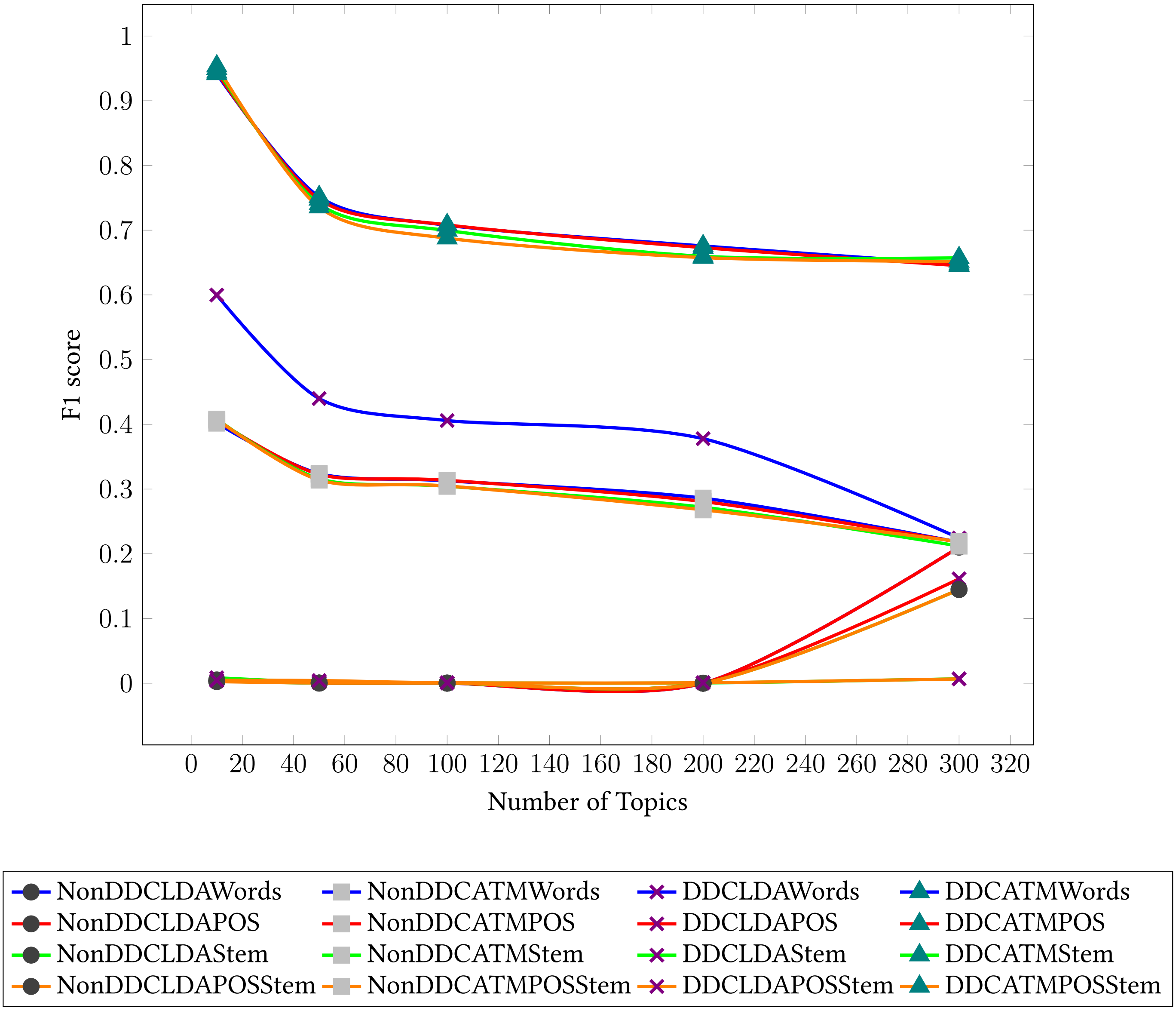

Figure 4.5: Measured F1 score based on a scale of $\times 10^{-2}$.

4.2.4 Context

In the following, the data analysis presented in the last three parts is put into context. In summary, the ATM models can achieve better values than the LDA models regardless of the corpus on which they were trained. This tendency is evident in the precision and recall values, regardless of the preprocessing of the data. For the ATM models, it is shown that pre-classification by the *text2ddc* provides a significant improvement in the results and thus enhances the models in the context of individual tweets. For the LDA models, on the other hand, classification by the *text2ddc* cannot achieve a sufficiently good improvement in the context of social media tweets. As described, the models are on the same point scale regardless of which corpus they were trained on.

Since most models perform best without preprocessing or only with part-of-speech filtering, it is reasonable to assume that processing the data with lemmatizing and stemming degrades the models. Part-of-speech filtering probably gets good scores because it only removes a small amount of data. In retrospect, only WH words were discarded, which did not result in a large change in the data set. Thus, the corpora most closely resemble the non-preprocessed corpora and therefore performed similarly well.

5 Discussion

5.1 Results

The evaluation of the models showed on an intrinsic and extrinsic level that the ATM models outperformed the LDA models in every way. There was also a clear dominance of the ATM models trained with the corpus classified by the *text2ddc*. They were consistently able to perform better than the ATM models trained with the unclassified corpus. However, this improvement could not be shown for the LDA models. Even though in individual cases the models trained with the classified corpus achieved better scores than the LDA models trained with the unclassified corpus, the results remained similarly poor. The use of classification to increase data quality did not lead to an improvement of the LDA models. The LDA models produce poor results that cannot be sufficiently improved by the use of classification. Therefore, it is questionable whether their use is advisable in the context of a social media analysis. Alternatively, other topic models should be used, such as ATM, which can be improved by classifying the input data in addition to their generally good results.

In terms of preprocessing of the data, a difference was found between the intrinsic and extrinsic ratings. The values of the intrinsic ratings were higher when the data was preprocessed with lemmatizing and stemming or lemmatizing and stemming in combination with the part-of-speech filter. Extrinsic ratings were higher when the data was not preprocessed at all or when only the part-of-speech filter was used. Since the intrinsic scores mainly examine the semantic interpretability of the topics and the extrinsic scores are more concerned with the response of the models to unknown test data, it can be assumed that lemmatizing and stemming have a positive impact on the interpretability of the topics, while no preprocessing improves the response to unknown data. Based on these results, the preprocessing of the data should be determined according to what is to be achieved with the topic model.

5.2 Challenges

There were several challenges that made it difficult to accomplish this work. The selection of German accounts as sources of the tweets proved to be problematic. The first attempt to create the account lists was based on finding lists that contained well-known personalities from the respective category. However, this not only proved difficult, but also presented the additional challenge that only a fraction of the list entries had Twitter accounts or met the requirements to be considered usable. This makes sense, as Twitter is not one of the most popular social media platforms in Germany. According to statistics from the state broadcasters ARD and ZDF, Twitter is used by only 2 per cent of the German population, while Instagram is used by 15 per cent and Facebook by 14 per cent (Beisch et al. 2020). It can also be assumed that not all accounts based in Germany actually tweet in German, as scientific accounts, for example, often use English to communicate with other scientists around the world, as English is the global language of science (cf. Ammon 1989, p. 205). It was not possible to rely purely on list selection. Therefore, the path chosen was to have accounts suggested by the Twitter algorithm. This inevitably leads to the list of accounts being biased in some way. The bias continues to drag through the Corona pandemic, as even accounts that have nothing to do with medicine and health post about this issue.

Other problems were encountered in the preparation of the data. The original categories that were set for the selection of accounts had to be reduced later. Originally there were 14 categories from which tweets were needed. However, during the later processing with the *text2ddc* tool, it was then found that no clear delimitation of the DDC classes to all 14 categories could be made. There were overlaps between the categories, so the decision was made to reduce the number of categories from 14 to 9. This ensured that all DDC classes could be clearly assigned. An unavoidable problem with the data is the limitation on the number of characters in a single tweet. The data sets contain tweets that stop in the middle of a sentence and are likely to continue in another tweet. Unfortunately, it was not possible to merge these tweets, which raises the concern that the topics of the individual tweets may not be fully captured and the topic models may be compromised as a result.

5.3 Future Work

The training of the models is based on Twitter data collected within one week. In order to check and validate the results obtained, the models should also be re-checked with Twitter data from a different time period. Increasing the size of the corpus, by using a longer time interval, could additionally change the results and should be verified. The review could also be extended by using other accounts from the same categories. It would also be interesting to analyse whether the language of the tweets has an influence and thus similar results could be obtained when using Twitter data from other languages.

The classifier *text2ddc* used in this work could also be replaced. It could be investigated whether similar results are obtained by using another classifier, or whether they differ from the results obtained with the *text2ddc*.

Another interesting extension attempt would be to use other texts from social media instead of Twitter data. These could be Reddit or Instagram posts, comments under YouTube videos or even Google reviews. In this way, one could check whether texts from other social media platforms offer more added value as selected documents than tweets.

Finally, one could try to improve the results by applying other pooling techniques to the Twitter data. These include user pooling, hashtag pooling and conversation pooling.

Bibliography

Heidrun Alex (2018). *Die Dewey-Dezimalklassifikation (DDC).* Berlin, Boston: De Gruyter, pp. 65–110. DOI: 10.1515/9783110299250-003.

James Allen (2003). "Natural language processing". In: *Encyclopedia of computer science*, pp. 1218–1222.

David Alvarez-Melis and Martin Saveski (2016). "Topic Modeling in Twitter: Aggregating Tweets by Conversations". In: pp. 1–4. URL: https://lsm.media.mit.edu/papers/topic-modeling-twitter.pdf (visited on 03/25/2021).

Ulrich Ammon (1989). "Schwierigkeiten der deutschen Sprachgemeinschaft aufgrund der Dominanz der englischen Sprache". In: *Zeitschrift für Sprachwissenschaften 8*, pp. 1–27.

Eduard Andrae and Philipp Rodewald (2019). *Micro-Influencer: Wenn weniger Reichweite die bessere Wahl ist.* URL: https://upload-magazin.de/19798-micro-influencer/ (visited on 03/25/2021).

Timothy Baldwin, Paul Cook, Marco Lui, Andrew MacKinlay, and Li Wang (Oct. 2013). "How Noisy Social Media Text, How Diffrnt Social Media Sources?" In: *Proceedings of the Sixth International Joint Conference on Natural Language Processing*. Nagoya, Japan: Asian Federation of Natural Language Processing, pp. 356–364. URL: https://www.aclweb.org/anthology/I13-1041.

Natalie Beisch and Carmen Schäfer (2020). "Ergebnisse der ARD/ZDF-Onlinestudie 2020 Internetnutzung mit großer Dynamik: Medien, Kommunikation, Social Media". In: pp. 1–20. URL: https://www.ard-zdf-onlinestudie.de/files/2020/0920_Beisch_Schaefer.pdf (visited on 03/25/2021).

David Blei (2011). "Variational inference". In: *Lecture from Princeton, variational inference.* URL: https://www.cs.princeton.edu/courses/archive/fall11/cos597C/lectures/variational-inference-i.pdf.

David Blei, Andrew Ng, and Michael Jordan (Jan. 2003). "Latent Dirichlet Allocation". In: *Journal of Machine Learning Research* 3, pp. 993–1022. URL: https://www.jmlr.org/papers/volume3/blei03a/blei03a.pdf.

Colin Campbell and Justine Rapp Farrell (2020). "More than meets the eye: The functional components underlying influencer marketing". In: *Business Horizons* 63.4, pp. 469–479. ISSN: 0007-6813. DOI: https://doi.org/10.1016/j.bushor.2020.03.003. URL: https://www.sciencedirect.com/science/article/pii/S000768132030032X.

Jonathan Chang, Jordan Boyd-Graber, Chong Wang, Sean Gerrish, and David M. Blei (2009). "Reading Tea Leaves: How Humans Interpret Topic Models". In: *Neural Information Processing Systems.* Vancouver, BC. URL: http://umiacs.umd.edu/~jbg//docs/nips2009-rtl.pdf.

Gobinda Chowdhury (2003). "Natural language processing". In: *Annual review of information science and technology* 37.1, pp. 51–89.

Stefan Debortoli, Oliver Müller, Iris Junglas, and Jan vom Brocke (2016). "Text Mining for Information Systems Researchers: An Annotated Topic Modeling Tutorial". English. In: *Communications of the Association for Information Systems* 39.1. ISSN: 1529-3181. DOI: 10.17705/1CAIS.03907.

Vincent Duriau, Rhonda Reger, and Michael Pfarrer (2007). "A Content Analysis of the Content Analysis Literature in Organization Studies: Research Themes, Data Sources, and Methodological Refinements". In: *Organizational Research Methods* 10.1, pp. 5–34. DOI: 10.1177/1094428106289252. eprint: https://doi.org/10.1177/1094428106289252. URL: https://doi.org/10.1177/1094428106289252.

Jacob Eisenstein (2013). "What to do about bad language on the internet." In: *HLT-NAACL*. Ed. by Lucy Vanderwende, Hal Daumé III, and Katrin Kirchhoff. The Association for Computational Linguistics, pp. 359–369. ISBN: 978-1-937284-47-3. URL: http://dblp.uni-trier.de/db/conf/naacl/naacl2013.html#Eisenstein13.

Tom Fawcett (June 2006). "Introduction to ROC analysis". In: *Pattern Recognition Letters* 27, pp. 861–874. DOI: 10.1016/j.patrec.2005.10.010.

John Firth (1957). "A Synopsis of Linguistic Theory 1930-1955". In: *Studies in Linguistic Analysis.* reprinted in Palmer, F. (ed. 1968) Selected Papers of J. R. Firth, Longman, Harlow. Philological Society, Oxford.

Leisa Reinecke Flynn, Ronald Goldsmith, and Jacqueline Eastman (1996). "Opinion leaders and opinion seekers: Two new measurement scales". In: *Journal of the academy of marketing science* 24.2, pp. 137–147.

Yoav Goldberg (2017). *Neural Network Methods for Natural Language Processing*. Vol. 37. Synthesis Lectures on Human Language Technologies. San Rafael, CA: Morgan & Claypool. ISBN: 978-1-62705-298-6. DOI: 10.2200/S00762ED1V01Y201703HLT037.

Thomas Griffiths and Mark Steyvers (2002). "A probabilistic approach to semantic representation". In: *Proceedings of the 24th Annual Conference of the Cognitive Science Society*, pp. 1–6. URL: https://cocosci.princeton.edu/tom/papers/semrep.pdf.

– (2004). "Finding scientific topics". In: *Proceedings of the National Academy of Science* 101, pp. 5228–5235. URL: https://www.pnas.org/content/101/suppl_1/5228.

Zellig Harris (1954). "Distributional Structure". In: *WORD* 10.2-3, pp. 146–162. DOI: 10.1080/00437956.1954.11659520. eprint: https://doi.org/10.1080/00437956.1954.11659520. URL: https://doi.org/10.1080/00437956.1954.11659520.

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

Thomas Hofman (2001). "Unsupervised Learning by Probabilistic Latent Semantic Analysis". In: *Machine Learning Journal* 42, pp. 177–196. URL: https://doi.org/10.1023/A:1007617005950.

Liangjie Hong and Brian Davison (2010). "Empirical Study of Topic Modeling in Twitter". In: *Proceedings of the First Workshop on Social Media Analytics.* SOMA '10. Washington D.C., District of Columbia: ACM, pp. 80–88. ISBN: 978-1-4503-0217-3. DOI: 10.1145/1964858.1964870. URL: http://doi.acm.org/10.1145/1964858.1964870.

M. Ikonomakis, S. Kotsiantis, and V. Tampakas (Aug. 2005). "Text Classification Using Machine Learning Techniques". In: *WSEAS TRANSACTIONS on COMPUTERS* 4, pp. 966–974.

Michael Jordan, Zoubin Ghahramani, Tommi Jaakkola, and Lawrence Saul (Jan. 1999). “An Introduction to Variational Methods for Graphical Models”. In: *Machine Learning* 37, pp. 183–233. DOI: 10.1023/A:1007665907178.

Jey Han Lau, David Newman, and Timothy Baldwin (Apr. 2014). “Machine Reading Tea Leaves: Automatically Evaluating Topic Coherence and Topic Model Quality”. In: *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics.* Gothenburg, Sweden: Association for Computational Linguistics, pp. 530–539. DOI: 10.3115/v1/E14-1056. URL: https://www.aclweb.org/anthology/E14-1056.

Elizabeth Liddy (2001). “Natural language processing”. In: *Encyclopedia of Library and Information Science.*

Kar Wai Lim, Changyou Chen, and Wray Buntine (2016). “Twitter-Network Topic Model: A Full Bayesian Treatment for Social Network and Text Modeling”. In: arXiv: 1609.06791.

Yan Liu, Alexandru Niculescu-Mizil, and Wojciech Gryc (2009). “Topic-Link LDA: Joint Models of Topic and Author Community”. In:

Chen Lou and Shupei Yuan (2019). “Influencer Marketing: How Message Value and Credibility Affect Consumer Trust of Branded Content on Social Media”. In: *Journal of Interactive Advertising* 19.1, pp. 58–73. DOI: 10.1080/15252019.2018.1533501. eprint: https://doi.org/10.1080/15252019.2018.1533501. URL: https://doi.org/10.1080/15252019.2018.1533501.

Andrew McCallum, Xuerui Wang, and Natasha Mohanty (2006). “Joint Group and Topic Discovery from Relations and Text.” In: *SNA@ICML*. Ed. by Edoardo M. Airoldi, David M. Blei, Stephen E. Fienberg, Anna Goldenberg, Eric P. Xing, and Alice X. Zheng. Vol. 4503. Lecture Notes in Computer Science. Springer, pp. 28–44. ISBN: 978-3-540-73132-0. URL: http://dblp.uni-trier.de/db/conf/icml/sna2006.html#McCallumWM06.

Heribert Meffert, Christoph Burmann, Manfred Kirchgeorg, and Maik Eisenbeiß (2018). *Marketing – Grundlagen marktorientierter Unternehmensführung.* 13th ed. Wiesbaden: Springer Gabler.

Rishabh Mehrotra, Scott Sanner, Wray Buntine, and Lexing Xie (2013). “Improving LDA topic models for microblogs via tweet pooling and automatic labeling”. In: *Proceedings of the 36th international ACM SIGIR conference on Research and development in information retrieval.* ACM, pp. 889–892.

Gary Miner, Dursun Delen, John Elder, Andrew Fast, Thomas Hill, and Robert Nisbet (2012). “Chapter 3 - Conceptual Foundations of Text Mining and Preprocessing Steps”. In: *Practical Text Mining and Statistical Analysis for Non-structured Text Data Applications.* Boston: Academic Press, pp. 43–51. ISBN: 978-0-12-386979-1. DOI: https://doi.org/10.1016/B978-0-12-386979-1.00003-7. URL: https://www.sciencedirect.com/science/article/pii/B9780123869791000037.

Ramesh Nallapati, Amr Ahmed, Eric Xing, and William Cohen (2008). “Joint Latent Topic Models for Text and Citations”. In: *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining.* KDD ’08. Las Vegas, Nevada, USA: ACM, pp. 542–550. ISBN: 978-1-60558-193-4. DOI: 10.1145/1401890.1401957. URL: http://doi.acm.org/10.1145/1401890.1401957.

OCLC (2019). “Introduction to the Dewey Decimal Classification”. In: *DDC 23 Introduction.* URL: https://www.oclc.org/content/dam/oclc/dewey/versions/print/intro.pdf.

Julio-Omar Palacio-Niño and Fernando Berzal (2019). "Evaluation Metrics for Unsupervised Learning Algorithms". In: arXiv: 1905.05667 [cs.LG].

Martin Porter (2001). "Snowball: A language for stemming algorithms". In: URL: http://snowball.tartarus.org/texts/introduction.html.

Kevin Quinn, Burt Monroe, Michael Colaresi, Michael Crespin, and Dragomir Radev (Jan. 2010). "How to Analyze Political Text With Minimal Assumptions and Costs". In: *American Journal of Political Science* 54, pp. 209–228. DOI: 10.1111/j.1540-5907.2009.00427.x.

Erhard Rahm and Hong Hai Do (2000). "Data cleaning: Problems and current approaches". In: *IEEE Data Eng. Bull.* 23.4, pp. 3–13.

Michael Röder, Andreas Both, and Alexander Hinneburg (Feb. 2015). "Exploring the Space of Topic Coherence Measures". In: *WSDM 2015 - Proceedings of the 8th ACM International Conference on Web Search and Data Mining*, pp. 399–408. DOI: 10.1145/2684822.2685324.

Michal Rosen-Zvi, Thomas Griffiths, Mark Steyvers, and Padhraic Smyth (2004). "The Author-Topic Model for Authors and Documents". In: arXiv: 1207.4169 [cs.IR].

Theresa Schmiedel, Oliver Müller, and Jan vom Brocke (May 2018). "Topic Modeling as a Strategy of Inquiry in Organizational Research: A Tutorial With an Application Example on Organizational Culture". English. In: *Organizational Research Methods.* ISSN: 1094-4281. DOI: 10.1177/1094428118773858.

Mohammad Sorower (June 2010). "A Literature Survey on Algorithms for Multi-label Learning". In: *Oregon State University*, pp. 1–25. DOI: 10.1016/j.patrec.2005.10.010.

Asbjorn Ottesen Steinskog, Jonas Foyn Therkelsen, and Björn Gambäck (2017). "Twitter Topic Modeling by Tweet Aggregation". In: *Proceedings of the 21st Nordic Conference of Computational Linguistics*, pp. 77–86.

Peter Turney and Patrick Pantel (2010). "From Frequency to Meaning: Vector Space Models of Semantics". In: *CoRR* abs/1003.1141. arXiv: 1003.1141. URL: http://arxiv.org/abs/1003.1141.

Tolga Uslu, 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. accepted. La Rochelle, France.

Robert Weber (1990). "Basic Content Analysis". In: vol. 2. SAGE Publications, pp. 1–99. ISBN: 9780803938632. URL: https://dx.doi.org/10.4135/9781412983488.

Xing Wei and Bruce Croft (2006). "LDA-Based Document Models for Ad-hoc Retrieval". In: *SIGIR*, pp. 1–8. URL: http://ciir.cs.umass.edu/pubfiles/ir-464.pdf.

Sidi Yang and Haiyi Zhang (2018). "Text Mining of Twitter Data Using a Latent Dirichlet Allocation Topic Model and Sentiment Analysis". In: *International Journal of Computer and Information Engineering* 12, pp. 525–529.

Appendix

Category Mapping

In the following the mapping of chosen accounts and the mapping of the DDC categories [1] to the predefined categories can be found. For a better overview, the sources used to determine the accounts have been added as footnotes where possible. For the categories for which no lists existed, a note *Twitter algorithm* was added.

[1] https://en.wikipedia.org/wiki/List_of_Dewey_Decimal_classes

Books

List of accounts[2][3]

Number	Name	Follower	Comment
1	@buchreport	12,4K	Magazine
2	@leanderwattig	13,7K	Publisher
3	@wasmitbuechern	14K	Connection Network Publisher
4	@buecherfrauen	5K	Bookseller
5	@marga_owski	134,7K	Author
6	@SharonDoduaOtoo	8,3K	Author
7	@Buchkolumne	16,5K	Author
8	@cafehaussitzer	7,8K	Book Blogger
9	@buechereiwien	16,6K	Library
10	@brodnig	49,7K	Author
11	@buchkultur	5K	Magazine
12	@literaturcafe	10,6K	Bookseller
13	@TanjaHanika	5K	Book Blogger
14	@CatStreff	2K	Book Blogger
15	@magretkind	3K	Author
16	@Xerubian	2K	Author
17	@VictoriaLinnea1	5,4K	Author
18	@sfischerverlage	14K	Publisher
19	@dtv_verlag	19,1K	Publisher
20	@piperverlag	19,2K	Publisher
21	@ReclamVerlag	8,7K	Publisher
22	@Aufbau_Verlag	13,8K	Publisher
23	@lovelybooks	15,7K	Book Blogger
24	@diogenesverlag	14K	Publisher
25	@Ullstein	12,5K	Publisher

[2] https://www.petra-schier.de/tipps-fuer-autoren/verlags-und-agentursuche/deutschsprachige-belletristik-verlage/

[3] Twitter algorithm

List of DDC categories

DDC Categories
010 Bibliographies
030 Encyclopedias and books of facts
050 Magazines, journals and serials
070 News media, journalism and publishing
080 Quotations
090 Manuscripts and rare books
400 Language
410 Linguistics
800 Literature, rhetoric and criticism

Business

List of accounts[4][5]

Number	Name	Follower	Comment
1	@klauseck	44,4K	Communication Advisor
2	@gsohn	19,2K	Economist
3	@MarkusWendt	8,4K	Management Consulting
4	@HolgerSchmidt	62K	Digital Economist
5	@karrierebibel	21,5K	Founder
6	@LenaRogl	21,3K	Head of Digital Challenges Siemens
7	@CWRoehl	15,4K	Capital
8	@AlleAktien	7,4K	Shares
9	@Tiefseher	16,7K	Finances
10	@marcusreif	5,9K	HR Director
11	@XING_de	49,1K	Network
12	@Michael_Kissig	6,7K	Value Investor
13	@marcfriedrich7	21,1K	Economist
14	@iw_koeln	14,3K	Institution of Economy
15	@schnellenbachj	9,5K	Economy magazine
16	@handelsblatt	335,2K	Economy magazine
17	@wiwo	250,3K	Economy magazine
18	@FAZ_Finanzen	56,3K	Economy magazine
19	@brandeins	163,1K	Economy magazine
20	@TijenOnaran	21,1K	Entrepreneur
21	@pm2null	6,2K	HR Blogger
22	@saatkorn	6,1K	Employer Branding
23	@SZ_Wirtschaft	91K	Economy magazine
24	@ZP_Universe	14,3K	Economy Blogger
25	@zeitonline_wir	77,2K	Economy magazine

List of DDC categories

DDC Categories
340 Law 650 Management and public relations

[4] `https://de.wikipedia.org/wiki/Kategorie:Wirtschaftsmagazin_(Deutschland)`

[5] Twitter algorithm

Charity

List of accounts[6]

Number	Name	Follower	Comment
1	@msf_de	34,3K	Ärzte ohne Grenzen
2	@UNICEFgermany	26,4K	UNICEF
3	@greenpeace_de	448,6K	Greenpeace
4	@BROT_furdiewelt	38,1K	Brot für die Welt
5	@WorldVisionDe	11,6K	WorldVision
6	@Caritas_web	24,2K	Caritas
7	@kindernothilfe	13,7K	Kindernothilfe
8	@Welthungerhilfe	149,3K	Welthungerhilfe
9	@WWF_Deutschland	492,7K	WWF
10	@SEENOTRETTUNG	40,2K	Seenotrettung
11	@NABU_de	145,4K	Naturschutzbund
12	@roteskreuz_de	106,2K	Rotes Kreuz
13	@VIERPFOTEN	17,8K	Vierpfoten
14	@amnesty_de	216,6K	Amnesty International
15	@soskinderdorf	7,6K	SOS Kinderdorf
16	@tierschutz_bund	114,5K	Tierschutzbund
17	@DKMS_de	19,1K	DKMS
18	@campact	149,2K	Campact
19	@PETADeutschland	409,7K	PETA
20	@bund_net	149,8K	Bund für Umwelt und Naturschutz Deutschland e.V.
21	@Umwelthilfe	63,1K	Umwelthilfe
22	@germandoctors	10,5K	German Doctors
23	@Oxfam_DE	43,5K	Oxfam
24	@nothilfe	11,7K	Medico International
25	@FridayForFuture	168,8K	Fridays for Future

List of DDC categories

DDC Categories
300 Social sciences, sociology and anthropology 360 Social problems and social services

[6] https://sozialmarketing.de/die-groessten-nonprofits-deutschlands/

Health

List of accounts[7][8]

Number	Name	Follower	Comment
1	@BMG_Bund	260,5K	Professor
2	@Flying__Doc	25,1K	Doctor
3	@ApothekerDer	41,4K	Pharmacist
4	@SchwesterFD	43,2K	Doctor
5	@demutsch	55,8K	Doctor
6	@Anaesthet1	14,7K	Anesthetist
7	@JoStowasser	44K	Assistant Doctor
8	@19Insomnia82	11,6K	Anesthetist
9	@Doktor_FreakOut	24,5K	Doctor
10	@drluebbers	47,8K	Doctor
11	@Arthrosezentrum	5,6K	Doctor
12	@hustendoktor	6,8K	Doctor
13	@FrankBettina	5K	Pro eHealth
14	@Dt_Aerzteblatt	24,6K	Magazine
15	@BARMER_Presse	8,3K	Health Insurance
16	@AOK_Politik	10,3K	Health Insurance
17	@focusgesundheit	18,5K	Magazine
18	@Dr_Emergencydoc	17,6K	Doctor
19	@DrCWerner	6,4K	Doctor
20	@TheBinderLab	7,9K	Research
21	@twankenhaus	11,6K	Health Thinktank
22	@Nell781	32,2K	Emergency Room
23	@Lam3th	5,9K	Doctor
24	@Susannchen01	7,3K	Information network
35	@DrInfluenza	13,6K	Doctor

List of DDC categories

DDC Categories
610 Medicine and Health

[7] https://de.wikipedia.org/wiki/Liste_deutscher_Krankenkassen

[8] Twitter algorithm

Politics

List of accounts[9][10][11]

Number	Name	Follower	Comment
1	@Markus_Soeder	250,1K	Politician
2	@jensspahn	232,2K	Politician
3	@JuliaKloeckner	73,1K	Politician
4	@peter_simone	34,5K	Politician
5	@Karl_Lauterbach	374,6K	Politician
6	@SWagenknecht	458K	Politician
7	@cem_oezdemir	219,6K	Politician
8	@NiemaMovassat	22,2K	Politician
9	@MalteKaufmann	29,6K	Politician
10	@c_lindner	435,8K	Politician
11	@MarcoBuschmann	17,7K	Politician
12	@Wissing	13,8K	Politician
13	@OlliLuksic	10,4K	Politician
14	@johannesvogel	19,8K	Politician
15	@Lambsdorff	19,8K	Politician
16	@RKiesewetter	11,1K	Politician
17	@HESommer	5,1K	Politician
18	@RegSprecher	999K	Politician
19	@CDU	354,3K	Party
20	@Die_Gruenen	549,1K	Party
21	@AfD	167,3K	Party
22	@OlafScholz	122,3K	Politician
23	@dieLinke	316,1K	Party
24	@spdde	391,1K	Party
25	@fdp	376,3K	Party

List of DDC categories

DDC Categories
320 Political science 350 Public administration and military science

[9] https://de.wikipedia.org/wiki/Portal:Politiker/Regierungsmitglieder

[10] https://de.statista.com/infografik/8656/die-aktivsten-deutschen-politiker-auf-twitter/

[11] Twitter algorithm

Science

List of accounts[12][13][14]

Number	Name	Follower	Comment
1	@pavel23	16,8K	Mathematician
2	@Umweltbundesamt	81,4K	Research Institute
3	@KITKarlsruhe	29,4K	Research Institute
4	@helmholtz_de	86,3K	Research Institute
5	@fz_juelich	8,1K	Research Institute
6	@PIK_Klima	18,3K	Research Institute
7	@weltderphysik	58,9K	Science magazine
8	@GundG	42,4K	Science magazine
9	@NatalieGrams	38,8K	Doctor
10	@unibern	13,9K	Universität
11	@dfg_public	23K	Science magazine
12	@dw_wissenschaft	78,9K	Science magazine
13	@spektrum	88,2K	Science magazine
14	@tudresden_de	14,4K	University
15	@DIW_Berlin	18,8K	Research Institute
16	@angew_chem	50,8K	Science magazine
17	@techreview_De	49,7K	Science magazine
18	@Sterne_Weltraum	51,4K	Science magazine
19	@NatGeoDE	8K	Science magazine
20	@BMBF_Bund	58,7K	Research Institute
21	@DLR_de	104,3K	Research Institute
22	@ESA_de	50,6K	Research Institute
23	@DLR_next	166,3K	Research Institute
24	@rki_de	451,9K	Research Institute
25	@Martin_Moder	12K	Molecular biology

[12] https://de.wikipedia.org/wiki/Kategorie:Wissenschaftliche_Zeitschrift_(Deutschland)

[13] https://www.dmg-ev.de/ausseruniversitaere-forschungseinrichtungen-deutschland-mit-meteorologischemklimatologischem-schwerpunkt/

[14] https://www.forschungsportal.net/forschungseinrichtungen-in-deutschland/

List of DDC categories

DDC Categories
500 Science
510 Mathematics
520 Astronomy
530 Physics
540 Chemistry
550 Earth sciences and geology
560 Fossils and prehistoric life
570 Biology
580 Plants
590 Animals (Zoology)

Shows

List of accounts[15]

Number	Name	Follower	Comment
1	@heuteshow	1,1M	Satire
2	@extra3	970,4K	Satire
3	@zdfmagazin	671,9K	Satire
4	@koesterfabian	101,4K	Heute Show
5	@janboehm	2,2M	Moderator
6	@FestFlauschig	55,2K	Podcast
7	@ArminRohde	114,2K	Actor
8	@latenightberlin	1,3M	TV Show
9	@SophiePassmann	160,4K	Moderator
10	@AnnaDushime	34,2K	Podcast
11	@gemischtes	37,3K	Podcast
12	@Christian_Huber	58,4K	Podcast Moderator
13	@ProSieben	1,8M	TV Channel
14	@sat1	553,1K	TV Channel
15	@RTLde	567,4K	TV Channel
16	@ZDF	1,2M	TV Channel
17	@DasErste	628,8K	TV Channel
18	@VOXde	251,7K	TV Channel
19	@BBielendorfer	20,4K	Moderator
20	@MickyBeisenherz	253,8K	Moderator
21	@swr3	147,5K	Radioshow
22	@Mann_mit_el	16,5K	TV Producer
23	@twitkalk	100,1K	Actor
24	@NetflixDE	1,2M	TV Channel
25	@sixxTV	210,7K	TV Channel

List of DDC categories

DDC Categories
791 Public performances 792 Stage presentations

[15] Twitter algorithm

Sport

List of accounts[16][17]

Number	Name	Follower	Comment
1	@11Freunde_de	266,4K	Soccer
2	@CollinasErben	36,9K	Soccer
3	@Bundesliga_DE	2,2M	Soccer
4	@TheBorisBecker	673,8K	Tennis
5	@SGFleHa	36,8K	Handball
6	@RNLoewen	55,7K	Handball
7	@liquimoly_hbl	56,2K	Handball
8	@MST_Formel1	16,1K	Motorsport
9	@gtmasters	12,6K	Motorsport
10	@Eurosport_DE	39,7K	Commentator
11	@TeamD	21,9K	Alpine Sport
12	@skiverband	44,3K	Alpine Sport
13	@ransport	167,9K	Commentator
14	@ZDFsport	619,3K	Commentator
15	@rannfl_suechtig	32,7K	NFL
16	@NFLDeutschland	66,5K	NFL
17	@NFLdeutsch	5K	NFL
18	@ZDFsportstudio	351,9K	Commentator
19	@basketball_de	6,7K	Basketball
20	@radentscheid	11,7K	Bike
21	@FrauKrone	5,3K	Bike
22	@sportschau	638,1K	Commentator
23	@SkySportDE	427,8K	Commentator
24	@Eisbaeren_B	21,7	Hockey
25	@RickGoldmann	11,6K	Hockey

List of DDC categories

DDC Categories
796 Athletic and outdoor sports and games 797 Aquatic and air sports 798 Equestrian sports and animal racing 799 Fishing, hunting, shooting

[16] https://cdn.dosb.de/user_upload/www.dosb.de/uber_uns/Bestandserhebung/BE-Heft_2017_aktualisierte_Version_25.01.18.pdf

[17] Google search

Technology

List of accounts[18][19]

Number	Name	Follower	Comment
1	@azrael74	11,5K	Founder 'Deutsche Startups'
2	@Kroker	26,6K	Wirtschaftswoche
3	@DomSchiener	42,8K	IOTA
4	@frank_thelen	51,1K	Entrepreneur
5	@mediadonis	27,1K	Ryte
6	@ChrisStoecker	31,2K	Spiegel Online
7	@teresabuecker	83,3K	SZ Magazine
8	@Doener	28,6K	t3n Magazine
9	@VerenaDE	13,9K	Fox&Sheep
10	@stratorob	32,4K	Digital Scouting
11	@electrouncle	5,7K	Entrepreneur
12	@jochensiegert	9K	Traxpay
13	@Herbert_Diess	23,7K	VWGroup
14	@klotzbrocken	8,9K	Payment and Banking
15	@CKemfert	28,2K	Leuphana University Lüneburg
16	@enormgruen	5,8K	Columnist
17	@DoroBaer	101K	Digitalisation advisor
18	@sascha_p	41,7K	Daimler
19	@heiseonline	240,3K	Digital magazine
20	@MkKueper	7,2K	Digital Coach
21	@HolgerSchmidt	62K	Entrepreneur
22	@martinfehrensen	12,3K	Entrepreneur
23	@ambajorat	11,4K	Entrepreneur
24	@aboutfintech	6,1K	Digital business
25	@Stefan_Hajek	5K	Publisher business magazine

List of DDC categories

DDC Categories
600 Technology 620 Engineering

[18] https://tytopr.com/german-tyto-tech-500-power-list-2019/

[19] https://tytopr.com/german-tyto-tech-500-power-list-2019/

Software Documentation

This section briefly introduces the software components developed and used. The section is divided according to the different programming languages used. The code can be found on GitLab[20].

PowerShell

RESTAPICall.ps1
API call script that retrieves all tweets from Twitter. Not only the tweets of a user are retrieved, but also the mentions and retweets. The API call can be defined according to the given values.

Random_Generator.ps1
Generates a random JSON file with an appropriately defined number of JSON objects from an original JSON file.

CountScript.ps1
Counts the number of JSON objects contained in a file.

CleaningScript.ps1
Includes all preprocessing steps for the cleaning of the data. This includes removing stop words, mentions, URLs, non-numeric values, tokens with less than 3 characters, tweets in a language other than German and tweets with less than 3 tokens. In addition, the script sets all letters to lower case.

Java

TwitterToXMI.java
Converts a JSON file into XMI format.

TwitterDDCAnalyse.java
Gets the distribution of the classes generated by DDC and puts them into a CSV file.

PieChartCreator.html
Implementation to create a pie chart of the DDC class distribution in a file.

DDCTaggedOutfilesOneFile.java
Puts every tweet, that was classified as belonging to one of the predefined categories, into a file in JSON format. Represents the back transformation of the data from the XMI form into the JSON format.

[20] https://gitlab.texttechnologylab.org/natalie_foerster/twitter-author-topic-modeling---comparative-and-classifactory-topic-analysis-using-latent-dirichlet-allocation

Python

MergeFiled.py
A script to merge several JSON files into one file.

RemoveDuplicates.py
Removes all duplicate JSON objects from a file in input format JSON.

RandomTrainTestGenerator.py
Splits a JSON file randomly into a train set and a test set. The proportions of the data sets can be adjusted dynamically.

LDA.py
This script trains the LDA models. Any processing step that prepares the data (as well as transforming it into a BOW model) can be performed by this script. The preprocessing steps that contain the additional data cleaning steps are also included. In addition, this script performs the calculations of the evaluation metrics.

ATM.py
This script trains the ATM models. Any processing step that prepares the data (as well as transforming it into a BOW model) can be performed by this script. The preprocessing steps that contain the additional data cleaning steps are also included. In addition, this script performs the calculations of the evaluation metrics.

Evaluation Results

The maximum values of the individual topics are marked as bold and italic.

Training Time

Training Time in Hours for all models with T as number of topics.

	Model	T = 10	T = 50	T = 100	T = 200	T = 300
NonDDC	LDAWords	***0.194***	0.432	0.605	0.803	1.125
	LDAPOS	0.258	0.435	0.753	***0.594***	1.230
	LDAStem	0.431	0.388	***0.318***	0.852	***1.098***
	LDAPOSStem	0.614	***0.385***	0.582	0.795	***1.098***
	ATMWords	2.871	2.638	3.909	3.348	4.212
	ATMPOS	3.616	3.647	3.685	4.248	4.098
	ATMStem	3.394	3.738	3.768	3.555	5.587
	ATMPOSStem	3.678	3.993	4.108	3.965	4.623
DDC	LDAWords	***0.035***	***0.030***	***0.043***	***0.061***	***0.089***
	LDAPOS	0.082	0.077	0.113	0.107	0.234
	LDAStem	0.089	0.081	0.117	0.117	0.27
	LDAPOSStem	0.09	0.078	0.125	0.194	0.24
	ATMWords	0.256	0.322	0.475	0.448	0.534
	ATMPOS	0.226	0.239	0.334	0.445	0.577
	ATMStem	0.258	0.34	0.364	0.465	0.581
	ATMPOSStem	0.253	0.343	0.357	0.413	0.58

Precision

Precision score for all models with T as number of topics.

	Model	T = 10	T = 50	T = 100	T = 200	T = 300
NonDDC	LDAWords	0.1851	0.0135	0.0058	0.0037	0.4222
	LDAPOS	0.2449	0.0098	0.0054	0.0031	0.4222
	LDAStem	0.1512	0.0141	0.0090	0.0058	0.6876
	LDAPOSStem	0.1193	0.0173	0.0192	0.0045	0.6876
	ATMWords	0.8214	***0.6608***	0.6373	***0.5830***	***0.4472***
	ATMPOS	0.8278	0.6579	***0.6391***	0.5721	0.4448
	ATMStem	0.8327	0.6457	0.6205	0.5542	0.4322
	ATMPOSStem	***0.8345***	0.6430	0.6212	0.5463	0.4445
DDC	LDAWords	***0.9008***	***0.6429***	0.5931	0.5518	0.3274
	LDAPOS	0.0526	0.0132	0.0096	0.0135	0.2354
	LDAStem	0.1260	0.0249	0.0065	0.0072	0.0995
	LDAPOSStem	0.0674	0.0604	0.0110	0.0114	0.0975
	ATMWords	0.7952	0.6347	***0.5972***	***0.5704***	0.5471
	ATMPOS	0.7962	0.6320	***0.5972***	0.5676	0.5442
	ATMStem	0.8014	0.6244	0.5905	0.5613	***0.5543***
	ATMPOSStem	0.8045	0.6219	0.5801	0.5551	0.5503

Recall

Recall score for all models with T as number of topics.

Scale: $\times 10^{-3}$	Model	T = 10	T = 50	T = 100	T = 200	T = 300
NonDDC	LDAWords	0.01947	0.00142	0.00616	0.00039	0.105207
	LDAPOS	0.0258	0.00104	0.000576	0.000336	0.1052
	LDAStem	0.0159	0.010148	0.000955	0.0006188	0.0724
	LDAPOSStem	0.0126	0.00182	0.00203	0.000482	0.0742
	ATMWords	0.20159	***0.16216***	***0.15641***	***0.14307***	***0.10976***
	ATMPOS	0.20314	0.16147	***0.15641***	***0.14307***	***0.10976***
	ATMStem	0.20434	0.15845	0.15227	0.13601	0.10608
	ATMPOSStem	***0.2047***	0.1578	0.15245	0.13407	0.10910
DDC	LDAWords	0.30887	0.22044	0.20337	0.18923	0.11228
	LDAPOS	0.0181	0.00453	0.00331	0.00463	0.0807
	LDAStem	0.0432	0.00855	0.00225	0.0025	0.0341
	LDAPOSStem	0.0231	0.0207	0.0038	0.00392	0.0335
	ATMWords	0.47173	***0.37655***	0.35428	***0.33843***	0.32461
	ATMPOS	0.47238	0.37492	***0.35431***	0.33675	0.32287
	ATMStem	0.47544	0.37043	0.3503	0.32934	***0.32646***
	ATMPOSStem	***0.47727***	0.36897	0.34413	0.32934	***0.32646***

F1 score

F1 score for all models with T as number of topics.

Scale: $\times 10^{-2}$	Model	T = 10	T = 50	T = 100	T = 200	T = 300
NonDDC	LDAWords	0.00389	0.00028	0.00012	0.00007	0.21034
	LDAPOS	0.00515	0.0002	0.00011	0.00006	0.21034
	LDAStem	0.00317	0.00029	0.00019	0.00012	0.14478
	LDAPOSStem	0.002519	0.00036	0.0004	0.00009	0.14478
	ATMWords	0.4019	***0.32392***	0.31272	***0.28592***	***0.21794***
	ATMPOS	0.4059	0.32272	***0.31352***	0.280731	***0.21794***
	ATMStem	***0.4079***	0.3159	0.304325	0.27193	0.21194
	ATMPOSStem	***0.4079***	0.31392	0.304725	0.26793	0.21877
DDC	LDAWords	0.5998	0.43984	0.40586	0.37787	0.22432
	LDAPOS	0.00361	0.0009	0.00066	0.00092	0.16134
	LDAStem	0.00845	0.0017	0.00044	0.00049	0.00681
	LDAPOSStem	0.00461	0.00413	0.00075	0.00078	0.00669
	ATMWords	0.94144	***0.75155***	0.70758	***0.67559***	0.64761
	ATMPOS	0.94403	0.74755	***0.70817***	0.673	0.64521
	ATMStem	0.94943	0.74036	0.6995	0.65961	***0.65721***
	ATMPOSStem	***0.95343***	0.73556	0.68759	0.65761	0.65159