Universität Hamburg
Fachbereich Informatik
Fakultät für Mathematik, Informatik und Naturwissenschaften

Classification of the Answers of the OMT

Bachelor-Thesis Mensch-Computer-Interaktion
Arbeitsbereich Language Technology

Eingereicht am: 25.09.2019

Eingereicht von:
Fabian Meyer
5meyer@informatik.uni-hamburg.de
Matrikelnummer: 6816480

Erstgutachter:
Prof. Dr. Chris Biemann
Zweitgutachter:
Dr. Gregor Wiedemann

Abstract

Implicit motives are unconscious intrinsic desires that cannot be measured with explicit methods such as self-report inventories. Usually, these motives are measured with projective methods such as the Operant Motive Test (OMT). During the OMT, participants are shown ambiguous stimuli in the form of simple drawn pictures of persons in different situations. The participants are asked to think up stories about what happens in the pictures and answer the provided questions. Trained psychologists label the answers to one of the three main motives, power, achievement, affiliation, or an additional zero level. The motives allow psychologists to predict behavior, educational and academic success, and long-term development. A restriction of the OMT is the long-lasting and expensive evaluation process. The objective of this thesis is to simplify this process by applying methods of machine learning to the classification problem. We use a dataset of labeled OMT answers to compare different text representation methods, including extensive pre-trained language models, and different classification methods. Especially the pre-trained language models proved their utility and showed to approach human-like performance in the classification of the answers of the OMT.

Contents

List of Figures

List of Tables

List of Abbreviations

BERT	Bidirectional Encoder Representation from Transformer
BOW	Bag-of-Words
CNN	Convolutional Neural Network
DMM	Dirichlet Multinomial Mixture
ELMo	Embeddings from Language Models
GB	Gradient Boosting
GRU	Gated Recurrent Unit
LDA	Latent Dirichlet Allocation
LFDMM	Latent Feature Dirichlet Mixture Model
LM	Language Model
LSI	Latent Semantic Analysis
LSTM	Long Short-term Memory
ML	Machine Learning
MNB	Multinomial Naïve Bayes
NB	Naïve Bayes
NLP	Natural Language Processing
NTM	Neural Topic Model
OMT	Operant Motive Test
OOV	out-of-vocabulary
pLSA	probabilistic Latent Semantic Analysis
RNN	Recurrent Neural Network
SOTA	state-of-the-art
SVM	Support Vector Machine
TAT	Thematic Apperception Test
TC	Text Classification
tf-idf	term frequency–inverse document frequency
TMN	Topic Memory Network
TM	Topic Model

1 Introduction

Motives select, direct and energize human behavior (McClelland 1988). Therefore, motives determine, in which sphere of life an individual spends effort, time, and hope in order to pursue a happy and fulfilling life.

The most obvious way to achieve this fulfilling life would therefore be to listen carefully to the inner self, detect the own, inner motives, and act accordingly. This is often unfeasible, as humans are often forced or distracted by situational influences and act contrary. Another problem of this approach is the composition of motives: Psychologists differentiate between to types of motives: explicit and implicit. As the names indicate, the explicit motives are present for introspective, while implicit motives are in general unconscious desires and needs. Humans can therefore not simply listen to their inner self and act accordingly and detect all inner needs. These motives can be detected by psychologists, which rely on written text or told stories of the subjects. As modern computers can learn from textual data and perform numerous text-related tasks, we aim to automatize the detection of the motives.

To measure, detect and explore human motives, researchers developed various tests intending to access them. The methods can be divided into two categories: introspective self-report inventories and projective tests. The introspective inventories utilize self-report questionnaires and reports from life records, obtained in rating scales such as the Likert scale. This is considered a highly subjective procedure and suffers from multiple limitations, such as a lack of adequate self-insight or downright dissimulation of subjects. The latter, projective tests, are designed to let a person respond to ambiguous stimuli to reveal hidden emotions and internal conflicts projected into the stimuli. These tests have origins in psychoanalysis and rely upon the fundamental assumption of humans having conscious and unconscious attitudes and motivations, already postulated by Sigmund Freud, known as the founder of psychoanalysis, already in the 19th and early 20th century.

The Operant Motive Test (OMT, Kuhl et al. 1999) is a motive measure that utilizes projective methods and is the objective of this thesis. The test consists of at least 15 ambiguous stimuli that are presented to subjects. These subjects are asked to answer the following questions for each stimulus:

- Who is the main person and what is important for the main person?
- How does this person feel?
- Why does the person feel this way?
- How does the story end?

The OMT and other projective tests are based on the assumption that subjects develop spontaneous emotions and needs from an *inner necessity*. From these spontaneous reactions, written down in short answers to the four questions, psychologists conclude different manifestations of the three motives power, affiliation, and achievement.
Up to this day, the OMT is manually analyzed and evaluated by trained professional psychologists, which results in a time- and cost-intensive evaluation, which furthermore results in a decline of on research in the area.

Meanwhile, recent advances in the area of Natural Language Processing (NLP) and especially text classification (TC) allow machine learning algorithms and deep neural networks to be utilized in tasks like language translation, grammatical analysis, sentiment analysis, language understanding tasks and many more with improving performance, approaching and even surpassing human-level performance in a variety of these tasks(Young et al. 2017). NLP is also utilized in psychology-related fields. Classifiers are trained to detect signs of dementia, can detect emotions, and are utilized to predict the degree of suicide risk in social network posts. However, human motives and motivation are today not in focus of the NLP community(Johannßen et al. 2018).

This thesis is intended to remedy the resource-intensive evaluation process of the OMT and contribute to the motive research with NLP. The objective is to automate the evaluation of the OMT with automated TC using methods of machine learning.

Due to the design of the OMT, answers often turn out short. The TC quality often suffers from short and spontaneous written texts, due to a lack of semantic information, and noise. Hence, in this thesis, we apply multiple TC methods that proved their utility in various other tasks
Beside models that are considered a standard in TC (S. Wang et al. 2012), topic models (Gildea et al. 1999) as well as language models are utilized as an attempt to gain additional knowledge from the data in the classification process. Topic models, in this context, refers to "probabilistic models of a corpus that not only assigns high probability to members of the corpus but also assigns high probability to other 'similar documents' (Blei et al. 2003), language modeling is the process of learning the joint probability function of sequences in a language (Bengio et al. 2003).

Based on this, the following questions arise, which determine the methodology of this work:

1. Can methods of machine learning learn to attribute motives to the answers of the OMT?

2. Do sentence representations, derived from Language Models, yield the potential to improve the classification of the answers of the OMT?

3. Do Topic Models yield the potential to improve the classification of the answers of the OMT?

To address these questions, we utilize a dataset of OMT answers, created at the Universität Trier. As our baseline, we consider the so-called zeroR classifier; a classifier with minimal domain knowledge assigning each instance to the most commonly occurring class. We train models based on machine learning, including different data transforming techniques and classification algorithms. To address the second research question, we will apply two different contextual embedding methods, including a State-of-the-art (SOTA) deep neural network architecture, to the given problem. To address Question 3, we use Latent Dirichlet Allocation (LDA) (Blei et al. 2003) to enrich features with TMs. The performance and improvements are measured in F_1 score, the harmonic mean of precision and recall (see also Chapter 3.2).

The structure of this thesis is as follows: First, we introduce related work in the area of TC, short text classification, and NLPsych (Johannßen et al. 2018), which describes the usage of machine learning to predict psychological traits in Section 2. Thereafter in Chapter 3, the theoretical background of the OMT and applied techniques is presented. Section 4, details the methodology, followed by an evaluation in Section 5 and discussion of results in Section 6. Finally, we present our conclusion in Section 7.

2 Literature Review

In this chapter, we present models and research in relevant NLP-tasks. A focus is set on language models and topic models for short text classification, as these are the foundations of this thesis. Additionally, we give a brief overview of NLPsych, the domain where this thesis is associated with. This chapter claims not to be a complete representation of research, but will present milestones and SOTA in relevant NLP-tasks.

2.1 Text Classification

Text classification is of booming interest in the last decades, due to the increased availability of documents and texts in digital form and numerous needs to organize them (Sebastiani 2002). The dominant approach to organizing documents today is text classification with machine learning. The automated classification of text became a major subfield of information systems in the 1990's This is mainly due to the more powerful hardware and the increased interest (Sebastiani 2002). TC is utilized in many different contexts, ranging from document filtering, metadata generation, word sense disambiguation, sentiment analysis or NLPsych[1].

Mainly two different approaches have shown to be useful. In the 1980's, the most popular approach for TC has been *knowledge engineering* (Sebastiani 2002). In knowledge engineering, experts in the particular topic built a set of manually defined logical rules, a *disjunctive normal form*. This disjunction of conjunctive formulas are the rules that determine the membership to a category. This requires well-defined and distinguishable categories. Also, the knowledge engineering approach to TC is often confronted with the *knowledge acquisition bottleneck*. The knowledge engineer, who manually defines the rules in cooperation with a domain expert, has to redefine rules, what, if the set of categories changes, might be connected to a time-consuming work. Furthermore, these classifiers can not be ported to a different domain, so for every use case at least two experts, the knowledge engineer and the domain experts (in some cases they might be the same person) have to create new rules and train a new classifier. So this system does not scale well and might even break down with new categories. Nevertheless, this approach showed some good results in TC like Hayes et al. (1990) with their CONSTRUE system showed, a commercial TC system build for the Reuters news agency, which was able to classify news into 674 categories and detect the presence of over 17,000 companies.

This work focuses on the second approach to TC, the machine learning (ML) approach, which became the most important already in the 1990's in the research community (Mitchell 1997). This approach focuses on building the classifier itself that learns automatically the characteristics of the categories derived from labeled

[1] A neologism created from the conference CLPsych and NLP, see also Chapter 2.4

instances of them (Sebastiani 2002). So the classifier, also called *learner*, learns iteratively from labeled documents, which characteristics belong to which class. With the learned characteristics, unlabeled documents can be classified afterward.

For classic machine learning, we present four different popular approaches: Naïve Bayes (NB, Ikonomakis et al. 2005), SVM (Cortes et al. 1995), Nearest Neighbor Classifier and Decision Trees.
NB approaches are widely applied in text classification tasks due to their simplicity, but results are often inferior to other classifiers. Nevertheless, they can be a useful baseline, especially in combination with shorter text instances (S. Wang et al. 2012) and there are different strategies to improve the performance (S.-B. Kim et al. 2002; Schneider 2010).
The SVM is a so-called *large margin classifier*. The purpose of SVM is to linearly separate n-dimensional space into two classes. In the context of text classification, this is often a very high dimensional feature space, where the SVM shows robustness against (Allahyari et al. 2017). SVM often yield best results in last decades in TC tasks (Sebastiani 2002), even though it might lack good recall scores (Ikonomakis et al. 2005).
Another family of classifiers are the Nearest Neighbor Classifiers. These are proximity-based classifiers based on distance measures like cosine similarity. The idea behind this strategy is, the closer two instances are, the more likely they are similar to each other. The test instance's labels are inferred from the class labels of similar documents in training set (Allahyari et al. 2017).
The last group of machine learning classifiers presented here are *Decision Trees*. These are hierarchical trees of the training instances. The data is divided hierarchically by attribute values of its dates. Each node of the tree is a test of some attribute of the training instance, so each branch, descending from this node corresponds to this value. The classification is done by (i) beginning at the root node, (ii) testing the attribute and (iii) moving down the branch corresponding to the value of the attribute. In TC, the conditions are usually terms in the text documents (Allahyari et al. 2017).
Ensemble methods (Zhou 2012) use a multitude of classifiers. Bagging classifiers train a subset of the training data in parallel on multiple classifiers. Afterwards, all individual predictions are aggregated, either voted or averaged, to form a final predictor. An often applied implementation of this principle is the Random Forest by Breiman (2001), where decision trees and random feature selections are utilized. In *Boosting*, a strong classifier is built from several weaker classifiers. This classifiers are trained sequentially to decrease error rates. A well known implementation is AdaBoost (Freund et al. 1996).

Since the accessibility of data and the computational power in general are still increasing, deep learning methods (Bengio et al. 2017) received increased interest also in TC. Especially Convolutional Neural Networks (CNN, LeCun et al. 1998), and Re-

current Neural Networks (RNN, Elman 1990) are applied TC-tasks, although these architectures are often not introduced and developed for TC or NLP in general.
W. Yin et al. (2017) compared CNN and recurrent architectures, such as Long Short-term Memory(LSTM, Hochreiter et al. 1997) and Gated Recurrent Unit (GRU, (Cho et al. 2014a)) in context of NLP. Their study detected that CNN yield great results in short texts while RNN excel in tasks where word dependencies are important, e.g. sentiment analysis. These *deep learning methods* (Bengio et al. 2017) achieved SOTA results in many TC areas (Conneau et al. 2016).

X. Zhang et al. (2015) demonstrated better results from different neural architectures compared to a traditional Bag-of-Words (BOW) model on various datasets and tasks, even though they did not adjust their machine learning model whilst adjusting the neural networks multiple times. In a comparison of classic machine learning methods and level convolutional networks, a combination of term frequency–inverse document frequency (tf-idf) document representation and a linear classifier is still competitive on different datasets (X. Zhang et al. 2015). Tf-idf is an extension of the BOW approach, where each word is weighted by the term frequency times the inverse of the document frequency (see also Chapter 3.2). So even if deep neural networks are powerful architectures that achieve high performance, not only in TC tasks, classic methods often obtain comparable results, if the right features are utilized (Joulin et al. 2016), while maintaining interpretability.

2.2 Short Text Classification

As online communication and e-commerce become more important as well as more frequent, a new genre of text growth: short texts. These texts consist of "a dozen words to few sentences" (Song et al. 2014). Characteristics are sparseness, large-scale, non-standardization and immediacy, so it is often difficult for traditional approaches in TC to deal with short texts. The previously described methods often fail to achieve the desired accuracy, because they rely on shared context, enough word co-occurrences, and the word frequency (Song et al. 2014). Most of the sources of short text are not reviewed (posts in social networks and forums like *Twitter* and *Reddit*, short messages, valuations, recommendations, and the answers of the OMT) and written with less care than e.g. newspaper article or books.
This results in more misspellings, more usage of slang, more abbreviations and, particular in German, the disregard of capitalization. In the context of text mining spoken, the language in short texts is very *noisy* (R. Zhang et al. 2013). As a consequence, the size of the vocabulary of text corpora increases, there is higher chance of out-of-vocabulary (OOV) or characters in new documents and sparseness of the vector representations. Methods to handle noisy texts are e.g. shown by Subramaniam et al. (2009). To increase the classification results, it is important to choose feature items and reduce the spatial dimension and noises. When we speak of better

results in this section, we refer to an improved accuracy and/or F_1 score on relevant datasets, such as Twitter Corpus[2] with 6 to 20 words per document on average.

TC is processed generally in vector space model under the assumption that the relationship between words is independent and texts are not correlated (Song et al. 2014). These semantic expressions are usually weaker in short texts, so research often focuses on additional semantic analysis. For leveraging semantic information multiple approaches exist, well known representatives are Latent Semantic Analysis (LSA, also referred to as Latent Semantic Indexing, LSI, Dumais et al. 1988), Probabilistic Latent Semantic Analysis (pLSA, Hofmann 1999), and Latent Dirichlet Allocation (LDA, Blei et al. 2003). These so-called *probabilistic TM* are used in different ways to improve the results of TC. Topic models, especially LDA assume that each document consists of multiple topic distributions (Blei et al. 2003). Here, the length limitations in short text prevent a good topic representation as it is hard to express multiple topics on few words (Li et al. 2018).
Researchers developed different strategies to still use topic models to increase the classification performance of short texts. Zelikovitz et al. (2000) introduced "Transductive LSI", where transductive describes a singular value decomposition that is applied not only on training data but, since it is an unsupervised algorithm, also on any available test data to improve the classification results. In the context of classification, "LSI returns the nearest neighbors of a test example in the new space, even if the test example does not share any of the raw terms with those nearest neighbors" (Zelikovitz et al. 2000). The accuracy of the transductive learner outperformed the basic nearest neighbor algorithm. In real-world applications, a limitation of this approach might occur that is already mentioned by the authors: This model is limited to tasks where you have sufficient time for the classification process to train a new, improved model and to tasks that provide many data at once. However, in many contexts, it might be possible to have insufficient labeled, but many unlabeled data where this model showed promising results.
Another approach to reduce sparsity of short texts is a framework proposed by Phan et al. (2008). The main idea is to collect a large-scale external data collection, a so-called *universal dataset* to each classification task and then performing a topic analysis for this dataset with the above-mentioned topic analysis models. One advantage of this approach compared to the first is that a universal dataset works for every classification task in the related domain, so the dependence on huge datasets is reduced. Thus, this framework is more flexible. As the authors mentioned, the accuracy improvement is not impressive, but reduced dependency from large labeled data collections, as they could achieve nearly the same accuracy with the universal dataset approach using five times less labeled training data (4,500 compared to 22,500). A related approach is presented by B.-k. Wang et al. (2012), who uses an LDA model to build a so-called "strong feature thesaurus" to achieve better results in short text classification. All these approaches use an external data source to enrich

[2] `https://trec.nist.gov/data/tweets/`

the sparse features in different ways and so highlight their semantic features that are otherwise, due to their shortness, very sparse. A limitation that might influence the transfer to our scenario is the nature of the answers of the OMT, as described in Section 3.1: Most of the research tries to classify texts to topics the documents are directly talking about:: military, politics, and sport (i.e. Q. Chen et al. 2017, B.-k. Wang et al. 2012), while in the OMT classification, we classify latent motives, which are not explicitly mentioned (see Chapter 3.1). So, collecting data on the topic *power* might work well when we assume the participants talk about power explicitly. As they are not discussing the topic *power*, it is not reasonable to expect an increased performance in numbers.

Other works focus on improving the quality of topic representations without external data. In their work, Li et al. (2018) compared different approaches to improve topic representation over short texts like the Dirichlet Multinomial Mixture (DMM, Nigam et al. 2000). In a cluster task, J. Yin et al. (2014) showed promising results with this model. The approach of the *Bigram Topic Model* (Griffiths et al. 2005), is to try keeping more semantical information than conventional TMs. As most of them are based on a BOW-representation, they lose sequences, and therefore a lot of semantics during processing. The Bigram Topic Model is based on the assumption that words, which occur often together (the bigrams) tend to belong more likely to the same topic. Thus, their conditional distribution is based on latent topics and the previous word. This approach is extended by Wallach (2006). She explored a hierarchical generative probabilistic model that incorporates n-gram statistics and bigram topic model. Cheng et al. (2014) went even further to overcome the sparsity and the rarity of word co-occurrence. As they propose, the frequency of words in individual short texts plays a less discriminative role than in lengthy text, making it hard to infer which words are more correlated in each document (Hong et al. 2010). Thus, they trained their *Biterm Topic Model* over the corpus, they avoid the problem of sparse pattern based on bi-terms and could improve previous related benchmarks. As TMs, no matter what architecture they use, focus on global information, some tried to combine them with embedding methods (e.g. Mikolov et al. 2013b; Pennington et al. 2014), which focus on local context (Li et al. 2018) to get the best of both methods (Moody 2016; Niu et al. 2016; Liu et al. 2015). The Latent Feature Dirichlet Mixture Model (LF-DMM) (Nguyen et al. 2015) uses a mixture of two distributions, the Dirichlet polynomial and word embedding instead of the Dirichlet distribution for topic-word like the DMM. In their research, Li et al. (2018) compared the DMM, biterm topic model, and LF-DMM with their normal settings as well as with added word embeddings. Results showed that the LF-DMM does not perform well with short texts in comparison. The DMM however, is highly dependent on the length of texts, as it is not reasonable to assume only one topic in longer short texts. The biterm topic model, especially with additional word2vec features (Mikolov et al. 2013b) yields even better results in given tasks.

One of the current best approaches is short text classification is achieved by the Topic Memory Network (TMN) (Zeng et al. 2018), a model composed of three components,

a neural topic model that infers topics in a given corpus, a topic memory mechanism that maps the inferred topics to classification features and a classifier, which is an added neural network, e.g a CNN or RNN, for classifying the derived features.

2.3 Language Models

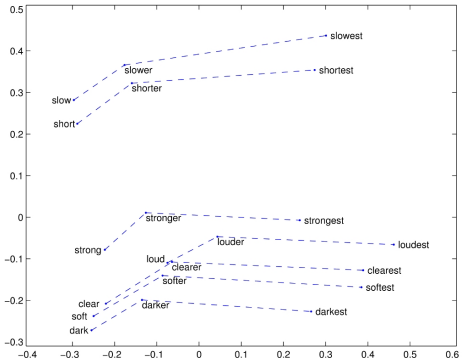
Language models can be defined as "function that puts a probability measure over strings drawn from some vocabulary" (Manning et al. 2008). The models can be utilized for speech recognition, machine translation, part-of-speech-tagging and many more applications. Word embeddings are trained like a language model and utilized in many TC tasks.

In general, the objective of LM is to approximate the possibility of a word w_n given its context, formally $P(w_n|w_1, \ldots, w_{n-1})$. Thus, LMs are trained to learn the sequences of words occurring in the corpus and project the sequences into numerical representation that yield information about its contexts, the word embeddings, many NLP-tasks, especially TC benefit from the obtained representations.

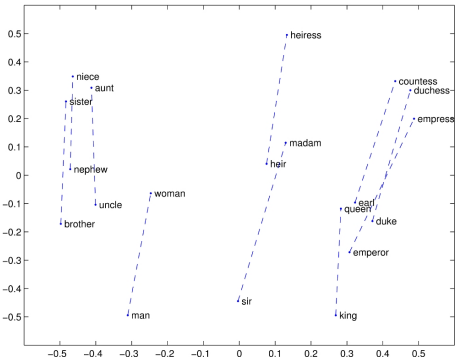
A fundamental language model is a count-based model or *n-gram language model*. The output of such a model is the most likely word based on the frequency of the previous n words. This is a very simple approach, which yields different limitations, primarily the handling of unseen words and the sparsity of the model. Different smoothing techniques (see e.g. S. F. Chen et al. (1999)) can improve the robustness of these models. The n-gram LMs are in general inferior to neural language models, thus this thesis focuses on the latter. The models, also called continuous space language model, help to remedy the data sparsity problem of n-gram LM and the *course of dimensionality*, meaning that the number of possible sequences grows exponentially with increasing vocabulary. Words are represented in a distributed way, more precisely as a non-linear combination of weights in a neural network.

Bengio et al. (2003) proposed a probabilistic model as one layer feed-forward neural network, still with the same objective, to approximate the probability of a word given n previous words, where n is a predefined parameter, called hyperparameter. Typically, this so-called *window size* is chosen within five to ten words, so distant context has no explicit influence on the language model (Mikolov et al. 2010). Recurrent Neural Language Models use their recurrent states to overcome this deficiency and cover longer contexts (Mikolov et al. 2010).

A major advance in the field was the release of pre-trained models. Well known and widely applied are *word2vec* (Mikolov et al. 2013b) and *GloVe* (Pennington et al. 2014). While word2vec utilizes local context for embeddings, GloVe is based on global corpus co-occurrence statistics. Both of these models produce representations for words with interesting properties, e.g. the cosine distance between to word vectors is a useful relatedness measurement in the vector space (Mikolov et al. 2013a). For instance, the nearest neighbor of a target word refers to the word vector with the highest similarity to the target word, and these neighbors tend to be related in



(a) Distances between comparatives and superlatives



(b) Distances and directions between gender specific terms

Figure 1: Roughly equal vector distances representing similar nuances necessary to distinguish related terms in the GloVe model (Pennington et al. 2014)

their meaning. In the pre-trained GloVe model[3], the nearest neighbors of the term "frog" are "frogs", "toad", "litoria", and "leptodactylidae", related animals. Furthermore, distances between gender-specific terms are very similar; taking the vector representation of *king* and adding the distance between the words *man* and *woman* results in a position near the word *queen*, which is visualized in Figure 1b. Also, comparatives and superlatives of adjectives tend to have similar distances (see Figure 1a).

A further improvement in the field of word embeddings was achieved by Facebook AI Research. Their embedding model, fastText[4], extended the word2vec approach by using *subword information* (Bojanowski et al. 2017). While word2vec and GloVe use word as smallest entities, fastText uses character n-grams (see Chapter 3.2.5). In this way, the model handles rare words and unknown words of an unseen corpus in general better. This is an advantage, especially in morphologically rich languages (Bojanowski et al. 2017), like German, because unknown words with a stem (e.g. changed by inflections) in the vocabulary can often be represented by the character n-grams. Additionally, fastText, as well as GloVe and word2Vec, can be used as pre-trained model, so e.g. be trained on a larger corpus to learn not only the sequences of the given corpus but of a vast amount of sequences in the given language. Thus, these pre-trained models are often utilized as input for downstream tasks, such as TC (Kowsari et al. 2019).

The limitations of such models are the sole vector representation of each word. For ambiguous words, such as *tank*, which refers to an armored military vehicle as well as to a container for liquids. For such words, all possible contexts are combined into one single vector representation. Furthermore, machine learning models often require a

[3] `https://nlp.stanford.edu/projects/glove/`
[4] `https://github.com/facebookresearch/fastText`

fixed length representation, thus, word vectors are often combined into a sentence or document vector by summing or averaging all word vectors (Le et al. 2014), which loses the positional information, the local context. Le et al. (2014) published the *paragraph vector* model, a fixed length representation of sentences, paragraphs and even documents, which takes word order into consideration. Natural language often yields long term connections to former or future words or sentences, thus different methods focus on embeddings, which represent longer context. These *contextualized representations* represent the context in which they appear, so the representation of a word is changing dynamically with changing context. Context2Vec (Melamud et al. 2016), using bidirectional LSTM (biLSTM), is an early proposal of contextual representation the foundation of other works.

Another representative is the Embeddings from Language Models (ELMo) representation (Peters et al. 2018) that provides character-based, deep contextual word representations, learned from internal states of deep biLSTMs. Additionally, the authors reported increasing performance on fine-tuning for specific NLP-tasks, even though the pre-trained model does perform well in these downstream task without the fine-tuning. This is referred to as "type of domain transfer for the biLM [bidirectional Language Model]" (Peters et al. 2018). In experiments, several benchmarks in downstream NLP-tasks, such as the SQuAD[5] by adding ELMo to the previous SOTA deep neural networks.

Bidirectional Encoder Representation from Transformer (BERT) (Devlin et al. 2019) provides contextual representation and transfer learning with a different architecture. While ELMo uses deep biLSTM representation and task-specific architectures including additional features, the so-called *feature based* approach in transfer learning (Devlin et al. 2019), BERT focus on *fine-tuning*. This implies minimal task-specific parameters and fine-tuning of all pre-trained parameters. Instead of LSTM, a multilayer Transformer encoder (Vaswani et al. 2017) is the main component of the architecture. While ELMo uses a bidirectional language model, BERT's objective is a masked language model (Taylor 1953), where tokens from the input are randomly masked and the model is trained to predict these tokens based on the context and a next-sentence-prediction objective. Fine-tuning on a task is implemented by adding a task-specific layer on top of the pre-trained model and fine-tune the model and classifier simultaneously. With this architecture, BERT achieved SOTA performance in many tasks, including classification related tasks like GLUE[6]. Interesting to notice is that BERT requires only a single additional classification layer to achieve SOTA, while former word or contextual embeddings were fed into deep neural networks for TC.

[5]Stanford Question Answering Dataset, were accomplished `https://rajpurkar.github.io/SQuAD-explorer/`

[6]https://gluebenchmark.com/

2.4 NLPsych, Motivation and Emotions

According to Johannßen et al. (2018), NLPsych yields the potential to access psyche, understand cognitive processes and detect mental health conditions. NLPsych covers many different domains like mental health detection systems, which includes i.e. suicide attempts, crisis and dementia, dream language, and motivation and emotion. The focus of this section is on the latter.

Pool et al. (2016) is set on a SVM for an emotion detection system. As training data they utilized Facebook posts, as labels, they utilized the corresponding Facebook reaction feature, also known as *emoticons* and tested it on three different emotion datasets. Similar works with these three datasets with different approaches are from S. M. Kim et al. 2010; Strapparava et al. 2008; Danisman et al. 2008.

Budhkar et al. (2018) used, similar to this thesis, topic models and language models to detect dementia. Another topic, more related to emotions than motivation, is *hate-speech*, which is defined as any communication that disparages a person or a group on the base of some characteristics such as race, color, ethnicity, gender, sexual orientation, nationality, religion, or other characteristic (Schmidt et al. 2017). A hate-speech detecting system is introduced by Serrà et al. (2017) and Warner et al. (2012), a survey is shown by Schmidt et al. (2017).

The most related work in terms of the specific domain is Johannßen et al. (2019), where the OMT is evaluated using linguistic features like type-token-ratio and ratio of spelling mistakes. Additional, features derived from the *linguistic inquiry and word count* (LIWC, Wolf et al. 2008) are used and classified with a logistic model tree. The machine learning model achieved an F_1-score of 80.1, approaching the human pairwise agreement of 85.33% (Johannßen et al. 2019). Furthermore, findings showed a significant correlation between the predicted power motive and bachelor thesis grades. To our knowledge, there are no other scientific works on operant methods utilizing machine learning.

3 Background

3.1 OMT Theory

The OMT, originally developed by Kuhl et al. (1999) is a projective test to measure the motives of a person. It is an extension of the thematic apperception test (TAT) by Murray (1943) that aims to classify the answers of participants to one of three main motives. This sections provides an introduction to motives and motivation from a psychological point of view as well as an overview of methods to measure them, the projective methods.

3.1.1 Motives & Motivation

To understand the meaning of the term motives, it is important to delimit it from motivation. *Motivation* is a current process, activated through stimulation of a *motive* to strive for a goal. It is a combination of personal and situational influences as well as the anticipated results (Heckhausen et al. 2005), illustrated in Figure 2. The situational influences (Part 2. in Figure 2) are possibilities, chances, and incentives and will change from time to time. The personal influences are stable over time. These influences can be split into three subsystems: (i) The universal behavioral trends and needs of mankind. This includes physical needs (food, water, etc.) and perceived self-efficacy. The physical needs are very similar for individuals, while the perceived self-efficacy can differ a lot. (ii) The motive dispositions, also called *implicit motives*, which are different from person to person. The implicit motives are relatively stable but unconscious needs (McClelland 1980) representing affective preferences that evolved through leaning and experience (McClelland 1988). Affective preference describes the habitual readiness of an individual to expose oneself to similar stimuli. Implicit motives are a key factor to explain, why individuals differ in actions, but an individual's actions are consistent through different situations. Furthermore, individuals will be energized in situations corresponding to their implicit motives. An example of an activated implicit motive is an individual who has an implicit motive to master new challenges: The individual will feel energized when assigned a challenging task. The unconscious nature of implicit motives results in the problem of measuring and determining them. Projective tests, such as

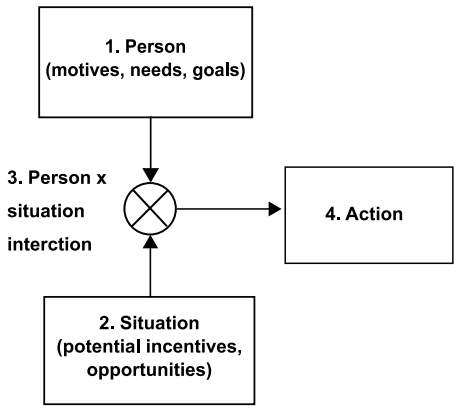


Figure 2: Model of determinants and course of motivated action (Heckhausen et al. 2005)

the TAT and the OMT are a useful tool to access implicit motives and are further explained in Section 3.1.2. (iii) The third factor is the goals or *explicit motives*. These goals are shaped by social norms, tangible rewards, and the beliefs of individuals about themselves, the self-concept (McClelland 1988). In contrast to implicit motives, explicit motives are self-attributed, actively pursued, and also verbally present values, goals and the self (Heckhausen et al. 2005).
Implicit and explicit motives are disjunct concepts and therefore necessarily in alignment. Often they are active in different circumstances, so there cannot be a conflict between them. Nevertheless, there are situations where an explicit motive overrides an implicit and vice versa. This process results in inner conflicts, and chronicle discrepancy between implicit and explicit motives do compromise well-being (McClelland et al. 1989) and can result in health issues (Heckhausen et al. 2005).

3.1.2 Implicit Motive Measures

To measure implicit motives, ordinary questionnaires are not suitable: The motives are usually not actively accessible for a subject. Thus, they are inferred from imaginative verbal or textural material. These methods are called projective tests and enable a psychologist to measure motives indirectly. The tests are called projective, because, according to the theory, subjects *project* implicit motives into the seen scenery and their imagined story.
In the majority of projective tests, subjects are asked to imagine a story to a presented, ambiguous stimulus, such as presented in Figure 3, an example from the TAT. The resulting stories are further evaluated for motive content. Most research focus on the three basic motives, need for power, need for affiliation, and need for achievement (see Chapter 1).

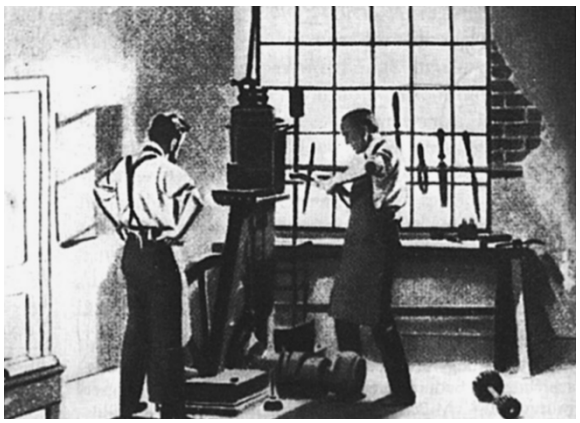


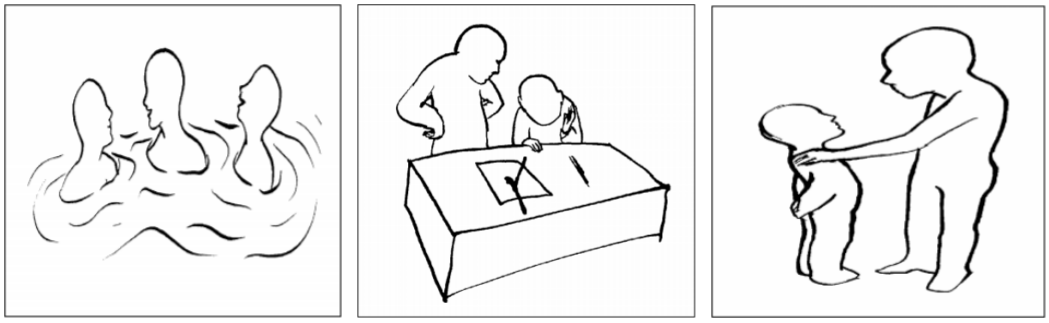
Figure 3: A so-called TAT card; an example of the stimuli presented to subjects in the TAT (Heckhausen et al. 2005).

The TAT (Murray 1943) is one of the first projective tests. The first version was designed already in the 1930s to reveal underlying dynamics of the subjects personality, and published as "[a] method for investigating fantasies" (Morgan et al. 1935). As the title reveals, the main subject of the TAT was to investigate hidden fantasies, "since the exposition of such hidden fantasies is one of the fundamental aims of analysis" (Morgan et al. 1935).
Therefore, a series of pictures is presented to the participant, each of them depicting a different event with the instructions to interpret each event, to give an imaginary reconstruction of the happenings that led to this event and the final outcome. In

contrast to questionnaires, these methods are based on an individual's sensitivity to motivational theme, while questionnaires rely on stable self-assessment. Figure 3 shows such a picture, which is often used to measure the activation of the achievement motive. In the TAT, each picture is related to one of the main motives. If the description or imagined story of the subject contains achievement related termini or formulations (e.g refers to a good grade, to extraordinary achievements like inventions or mentions achievement related long term goals), the authors concluded a strong achievement motive in the subject, respectively one of the other motives for other pictures.
Even though the TAT has been utilized for a long period of time and is still in use, it has several limitations. The imagined stories can be very long. This results in two issues: First, the test itself and the evaluation process become very time consuming, and second, the informative content of the written stories decreased after six pictures. Thus, in most application, only six pictures are used to maintain reliability (Atkinson 1958). Furthermore, while writing down imagined stories, complex processes can overwrite the implicit content of the story, e.g. the desired self-presentation influences the story, so that implicit motives are not perceptible (Kuhl et al. 1999). Beside that, the TAT showed variable reliability, thus no acceptable test-retest correlation could be achieved (Heckhausen et al. 2005).

3.1.3 The Operant Motive Test

The OMT is, similar to the TAT, a projective test to measure implicit motives. The OMT applied several innovations to address the limitation of the TAT, but still utilize the positive aspects of imagining a story: First, instead of six, the OMT utilizes at least 15 pictures. The general idea remains still the same as in the TAT: subjects are encouraged to imagine a story for the presented pictures. The difference is that the OMT asks the subjects for brief answers to the four questions (see Chapter 1) as spontaneous as possible. This less time consuming process enabled (Kuhl et al. 1999) to utilize the significant higher number of pictures. Additionally, spontaneous answers are linked more directly to implicit motives (Baumann et al. 2005).
Another change is the usage of simple drawings instead of photographs, which yields more neutral stimuli to facilitate the subjects identification with the depicted character (see Figure 4). In contrast to the TAT, each picture is utilized to stimulate each motive, even though each picture tends to stimulate one of the main motives more than the others. So instead of rating the activation of one motive from 1 to -1, each answer is labeled with one of the three motives.
To each of these motives (Kuhl et al. 1999) added levels of affective valence ranking from 1 to 5 to distinguish the motives even further. Level 1 represents self-regulating affect, 2 incentive driven, 3 self-driven, 4 active avoidance and 5 passive avoidance (see Kuhl et al. (1999)). If no motive could be assigned, there is an additional zero motive and level, annotated as 0 for both, level and motive. All in all, there are 4



(a) Affiliation (b) Achievement (c) Power

Figure 4: Ambiguous pictures utilized in the OMT. Each of them is suitable to activate all motives, not solely the mentioned, but the chosen examples tend to stimulate the mentioned motive (Kuhl et al. 1999).

times 6 different possibilities. Even though it is possible to assign affiliation, power, or achievement with the valence level 0, in practice these cases rarely occur.

Implicit motives, as evidenced e.g. in Schüler et al. (2015), Kuhl et al. (2003), Wegner et al. (2014), and Hofer et al. (2005), allow for characterization of behavior, subsequent success in academical and professional life and personal long term development. The OMT already reduced costly and time-consuming annotation, which is a major reason for the decline of motive research. With our attempt to automate the annotation, we hope to further reduce this process to support motive related research in the future.

3.2 Relevant Machine Learning Concepts

Machine learning tasks are usually divided into two categories: supervised and unsupervised. In supervised learning, we have access to the input and the desired output. The supervised algorithm tries to map the input to the output and thereby built a generalizing model, which should map new, unseen inputs of the same kind to an output. In unsupervised tasks, as there is no such output. Nevertheless, unsupervised algorithm can be utilized to gain additional knowledge from the input, which can be utilized in the supervised classification process. This approach is known as semi-supervised task.
Chapter 2 provides a general overview of different approaches in classification tasks. In this chapter, we present the concepts of classification algorithm, features, models, and metrics utilized in the classification of the answers of the OMT.

3.2.1 Classification Task

The task of assigning a category or class to an instance of a given dataset can formally be described as $\langle d_j, c_i \rangle \in D \times C$ where D is the domain of the given data and $C = \{c_1, \ldots, c_{\|n\|}\}$ is the set of classes. The underlying problem can more precisely formalized by the unknown target function $\hat{\phi} : D \times C \mapsto \{T, F\}$ that the function $\phi : D \times C \mapsto \{T, F\}$ approximates.T, F refer to the boolean variables True and False, ϕ is the classifier which should coincide with $\hat{\phi}$ as much as possible. The classes C shall be symbolic and thus carry no further meaning. All knowledge is extracted from the data and can therefore be called *content-based* classification.

There are basically two different types of TC: You either map each instance $d_j \in D$ to exactly one class $c_i \in C$. If there are only two classes, each instance $d_i \in D$ needs to be assigned to either c_i or its complement $\bar{c_i}$. This is considered a binary classification.
Assigning exactly one class $c_1, \ldots, c_n \in C$ with $n > 2$ to each $d_j \in D$ is called *multiclass classification* or the non-overlapping case, because classes do not overlap, so there are no instances that are classified to more than one class. In the other case, more than one class can be assigned to each instance, called *multilabel categorization* or the overlapping case. Multilabel-classification allows the assignment of any combination of labels to a instance, so it is a task of exponentially combinatorial difficulty, resulting $2^{|C|}$ possible assignments.
This work applies multiclass classification, due to the design of the OMT and the focus on classifying solely the motives, not the affective levels.

3.2.2 Classification Algorithms

This introduce the classification algorithm we applied to the given problem and their theoretical foundations.

Support Vector Machine

SVMs are a group of supervised learning algorithm, utilized for both classification and regression. SVMs are binary classifier, so to solve a multi-class problem, there are two different approaches: The *one-vs-rest* approach, where each hyperplane separates a class from all other classes and the *one-vs-one* approach (Knerr et al. 1990), where each classifier is trained to separate class $c_i \in C$ from every other class individual, resulting in $\frac{|C| \times (|C|-1)}{2}$ classifiers.
In practice, the *one-vs-rest* approach is often utilized, as results are equal to the one-vs-one approach, but computation times is noticeably lower, especially on a increasing number of classes C (Rifkin et al. 2004). Thus, we focus on the one-vs-all approach.

The SVM is a so-called large margin classifier. Given a set of n datapoints, the goal is to linearly separate the datapoints, given their class hyperplane. The SVM selects the hyperplane with the largest margin. As Figure 5 indicates, the optimal hyper-

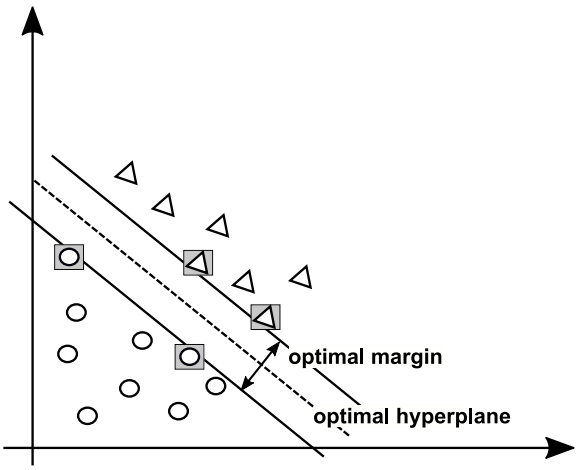


Figure 5: A SVM on a linearly separable problem in a two-dimensional space. The support vectors, marked with gray squares, define the largest margin of separation of the two classes (Cortes et al. 1995).

plane lays between the optimal margin, the margin with that maximizes distances between the classes.

Given a transformed dataset X_{Train} of n points of the form $\{(\vec{x_i}, y_i)|i = 1, ..., n; y_i \in \{-1, 1\}\}$, with $\vec{x_i} \in X_{Train}$, the p-dimensional vector and y_i, the according label 1 or -1, a SVM determines the most appropriate separating hyperplane between the two classes.

Let $\vec{w}$ be a normal vector to the hyperplane. $\vec{x_i}$ is a p-dimensional feature vector and b is the distance between $\vec{w}$ and the hyperplane orthogonal to $\vec{w}$, called *bias*. The hyperplane is described by the set of points satisfying $sep(x) = 0$ with $sep(x) = \vec{w} \cdot \vec{x} - b$. For every $\vec{x_i}$ a label $y_i \in \{-1, 1\}$ is assigned, depending on $sep(\vec{x_i})$ being higher or lower than 0. By minimizing the generalization term $\frac{1}{2}||\vec{w}||_2^2$ we maximize the minimum distance from all examples to the hyperplane and increase the generalization of the model. By ensuring every x_i is assigned to the right label, the hyperplane with the largest margin is selected.

To handle feature spaces, which are not perfectly linearly separable, the hinge loss function $max(0, 1 - y_i(\vec{w} \cdot \vec{x_i} - b))$ is applied. The function returns 0 when $\vec{x_i}$ is classified right, otherwise the distance from the margin is returned. We further utilize the slack variable ξ to allow non-perfect classification. C is a trade-off parameter between allowing false classification and minimizing the loss. It is a hyperparameter that has

to be selected beforehand. The optimization problem is described as:

$$\begin{aligned} &\textit{minimize} \quad \frac{1}{2}||\vec{w}||_2^2 + C\sum_{x_i \in X} \xi_i \\ &\textit{subject to} \quad y_i(\vec{w}\cdot\vec{x_i}+b) \geq 1-\xi_i | \forall \vec{x_i} \in X_{Train} \end{aligned}$$

The formulated optimization problem, called the primal, can be solved efficiently. Application sometimes use the optimization problem in its dual form. The normal vector $\vec{w}$ with a Lagrange multiplier α can be described as

$$\vec{w} = \sum_{\vec{x_i} \in X_{train}} \alpha_i y_i \vec{x_i},$$

the linear combination of training examples. Now, the dual optimization problem is described as follows:

$$\textit{maximize for } \alpha: \quad \sum_{\vec{x_i} \in X_{Train}} \alpha_i - \frac{1}{2} \sum_{\vec{x_i} \in X_{Train}} \sum_{\vec{x_j} \in X_{Train}} \alpha_i \alpha_j y_i y_j \langle \vec{x_i}, \vec{x_j} \rangle,$$

with $\langle \vec{x_i}, \vec{x_j} \rangle$, the pairwise dot product of two vectors. In the dual formulated problem, the margin is defined by the subset of all points, where $\alpha \neq 0$, laying exactly on the margin or, if $\xi \geq 0$, within.

For classification problems, which are not linearly separable, the so-called *kernel trick* is applied: All dates are projected into a higher dimensional space, formally:

$$\phi : \mathbb{R}^{d_1} \to \mathbb{R}^{d_2}, \vec{x} \mapsto \phi\vec{x}, \text{ with } d_1 < d_2.$$

In the higher dimensional space, the number of possible linear separations increases (Cover's Theorem, Schölkopf et al. 2001). Therefor, the SVM can linearly separate most problems, even though they are not linearly separable by default. To solve the transformation efficient, there are different kernel tricks applicable, like the polynomial kernels or radial bases function.

Naïve Bayes

NB classifier are based on Bayes' Theorem that describes the probability of an event based on prior knowledge of conditions related to the event. Naïve refers to the simplified assumption of conditional independence between every pair of features given the class. We focus on Multinomial Naïve Bayes (MNB), because this implementation yields often best results in TC tasks (S. Wang et al. 2012).

Formally, Bayes' theorem states the following relationships for the classes y and dependent feature vectors $x_i, \ldots, x_n$, where n is the number of features. In case of TC often the count of unique token of a corpus (see also Chapter 3.2.3):

$$P(y|x_1, \ldots, x_n) = \frac{P(y)P(x_1, \ldots, x_n|y)}{P(x_1, \ldots, x_n)}.$$

Assuming $P(d_i|y)$ is independent from all other $d_j \in D$, the relationship of classes and instances of D is simplified to:

$$P(y|d_i, \ldots, d_n) = \frac{P(y)\prod_{i=1}^{n} P(d_i|y)}{P(d_i, \ldots, d_n)}$$

In the training process $P(d_i, \ldots, d_n)$ is constant. Thus, the classification rule is defined as follows

$$P(y|x_1, \ldots, x_n) \propto P(y)\prod_{i=1}^{n} P(x_i|y)$$

$$\hat{y} = argmax P(y)\prod_{i=1}^{n} P(d_i|y),$$

where $\hat{y}$ is the inferred label. $P(y)$ is the relative frequency of class y. $P(x_i|c)$, the probability of feature i appearing in a sample mapped to class y.

MNB implements the algorithm for multinomially distributed data. This distribution is parameterized by vectors $\Theta_y = (\Theta_{y1}, \ldots, \Theta_{yn})$ for each $y_i \in y$, with n the number of features. Θ_{y_i} is $P(x_i|y)$, the probability of a feature i appearing in a sample of class y.

Θ_c is estimated as the relative frequency count:

$$\hat{\Theta}_{y_i} = \frac{N_{yi} + \alpha}{N_y + \alpha n},$$

with α, the smoothing parameter to prevent zero probabilities in further computations and for features not present in learning examples. N_{yi} is the number of times feature i appears in a sample of class y, N_y is the total count of all features for class y.
The MNB classifier is known to be robust against irrelevant features, such as words that occur frequently in every class, cancel each other out in the probability computations. Additionally the MNB is a very fast to compute. A negative aspect of the algorithm is that it cannot handle continuous representations well, as it relies on already known, discrete features.

3.2.3 Text Feature Extraction

Machine learning classifiers as mentioned above cannot use raw text as input. Therefore it is necessary to transform the input to different representations, which are so-called features. In this section, we focus on features that can be directly extracted from the textual data, in opposite of following sections, where complex transformation and calculation are utilized to create features.

There are numerous possibilities on creating features out of text, like word count models, number of spelling mistakes, co-occurring of words, or the number of utilized adjectives in a text, to name a few. Each of these features *might* yield information with respect to the classification task. An exhaustive search through all possibilities is therefore not practicable, so the approach of this thesis is to focus on features that proved their usability in TC tasks, namely the BOW and the tf-idf representations of textual data.

Bag of words (BOW)

BOW is a count-based vector representation of a document or collection of documents. The feature utilized is the occurrence of each word. BOW is fast to compute but does not pay attention to the varying importance of terms in a document and relatedness of terms, but it is still a considerable model and utilized as baseline in TC tasks.
In the BOW-model, each document d of a corpus D is represented as $|V|$-dimensional vector, where V is the total number of unique terms, called the vocabulary of corpus D. This example will elaborate on the simplicity and effectiveness of the BOW model:

Assuming we have a corpus D consisting of the four documents ‹I like cats›, ‹I like dogs›, ‹cats i like›and ‹cats hate dogs›. For simplicity, we assume each document is processed in upper-case letters, so that the resulting vocabulary $V_D = \{I, LIKE, CATS, DOGS, HATE\}$. The documents get transformed into an $|V_D|$-dimensional vector, where each position represents one term of V_D. In this example, the first value of the resulting vector represents the occurrence of the term ‹I›. Each occurrence of this term in a document increases the value by one, while all other values remain zero, as shown in Table 1. This model does not take respect

	I	LIKE	CATS	DOGS	HATE
I LIKE CATS	1	1	1	0	0
I LIKE DOGS	1	1	0	1	0
CATS I LIKE	1	1	1	0	0
DOGS HATE CATS	0	0	1	1	1

Table 1: BOW representation of example corpus D

of the order of the word. The documents ‹I like cats›and ‹cats i like›are represented identically.
The BOW model does not only ignore the order of words, it also assumes that all words yield the same information value for the classification task, as there is no weighting or any other highlighting involved. As the mentioned example elaborates, documents with different meanings but composed of the same words obtain identical vector representation, which seems to be insufficient. But in a lot of classification tasks, the representation are sufficient: Assuming the TC task is to classify docu-

ments into two classes, *animal related* or *not animal related*. In this special case, the identical representation of the two sentences is sufficient, as they both are related to the topic animals. The good results of this approach on various tasks indicates that for many TC tasks simple occurrence and co-occurrence of words is already discriminative for classes. The model can also be extended and use n-grams in a similar way, which enables the model to consider at least minimal local context.
Applied to the classification of the answers of the OMT, there are some pitfalls that might influence the utility of this approach, already discussed in Chapter 2: The vector representation with BOW are very sparse, as there are answers containing less than ten words while the vocabulary contains more than 70,000 words. The sparsity makes it difficult to classify them sufficient, as word occurrences and co-occurrences will be very rare. Additional, noise in the data reduces the usability of the approach as well.

Term frequency-inverse document frequency

Tf-idf is a weighting method for BOW features, frequently applied in information retrieval and text mining. The method enables a highlighting of indicative and important terms for a document while reducing the influence of frequent, less important words.
This weighting can be useful for several reasons: If a specific term appears frequently in the corpus, it is unlikely for this term to be significant for a single document. If a term appears frequently in a document (high document frequency), than it is reasonable to assume this term is *important* for the specific document, especially if the term does not appear frequently in other documents (the document frequency). Based on this logic, tf-idf weights words based on their occurrence statistics.

To elaborate this, assume a collection of books as document corpus. Each book is very likely to contain the term ⟨and⟩, thus, it has a very high document frequency. If the corpus contains a scientific book about ⟨litoria⟩, a genus of frogs native in Australia, this book is likely to contain the term ⟨litoria⟩ very often (high word frequency). Additional, the term is very unlikely to occur in many other books of the corpus (low document frequency). Thus, the term is assumed to be more indicative for this book than the term ⟨and⟩.

Tf-idf consist of two parts: The term frequency (tf) measures how frequently a term occurs in a document. The inverse document frequency (idf) measures how relevant a term is in context of the whole corpus. We obtain the idf of a term by dividing the number of documents in the corpus by the count of documents where term t occurs, formally:

$$idf(t) = log\frac{1+n}{1+df(t)} + 1$$

with $n \in N$, the number of documents and $df(t)$ the count of documents that contain

t. The 1 is added for smoothing the idf. The tf-idf value of a term t in a document D is defined as

$$tf - idf(t, D) = tf(t, D) * idf(t)$$

While tf-idf representation often yield better results as BOW, noisy and short text can limit the approach. The tf in very short texts will be one for every word while the idfs might also be more similar compared to the example with books mentioned above. Misspellings will also be misleading, because both tf and idf will handle them as independent tokens.

3.2.4 Topic Models

TMs are statistical models for discovering abstract *topics* occurring in a collection of documents. In general, these statistical methods discover hidden semantic structures in a document. The topics can also be described as *clusters* of similar words, because words that occur often together in documents refer to the same kind of concepts and will therefore be clustered into the same topic. As there are many different approaches to model latent topics on text collections, this thesis focus on the frequently utilized LDA as well as promising approaches to short text classification, the TMN.

Latent Dirichlet Allocation

LDA was developed not explicitly to find latent topics, but to find short descriptions of the members of a collection, mainly for large text collections. These short descriptions of the members should inherit the essential statistical relationships of the collection for basic tasks, such as TC. Thus, LDA is also considered a *dimensionality reduction technique*.

Given a corpus of m documents $D = \{d_1, \ldots, d_m\}$, where each document is a sequence of N words, denoted as $d = (w_1, ..., w_N)$. The basic assumption of LDA is that each document is a random mixture over K latent topics. These topics are each characterized by a distribution over all words in the corpus. Figure 6 illustrates the influence of variables on the texts. The variable names are defined as follows:

- α, the parameter of the Dirichlet prior on the per-document topic distribution
- η the parameter of the Dirichlet prior on the per-topic word distribution
- β the word distribution for topic k
- θ_i the topic distribution for document i
- z_{ij} the topic for the j-th word in document i
- w_{ij} the specific word

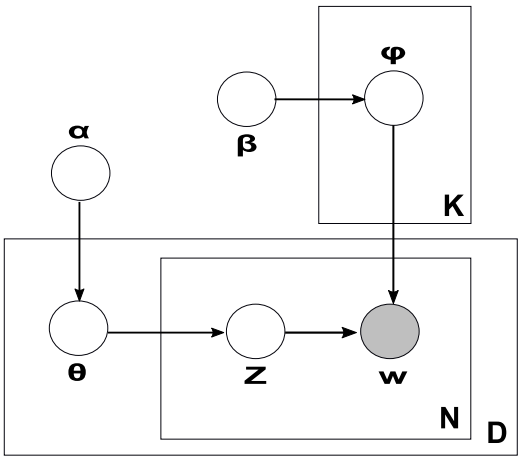


Figure 6: Graphical model of LDA (Blei et al. 2003)

This plate notation represents replicates. The outer plate, or box, represent documents, the annotation D is the quantity of documents. The inner box represents the repeated choice of topics and words within a given document. The order of words, and therefore of the topics inside of a document is of no further interest in this model. Important to notice at this point is that the W in Figure 6, representing the words in a document, is grayed, because it is the only observable variable in the model. LDA assumes the following generative process for each word in each document:

- Choose $N \sim Poisson(\xi)$
- Choose $\theta \sim Dir(\alpha)$
- For each of the N words w_n:
  - Choose a topic $z_n \sim Multinomial(\theta)$.
  - Choose a word w_n from $p(w_n|z_n, \beta)$, a multinomial probability conditioned on the topic z_n

As only the words are observed in the model, the parameters α and η are estimated to infer which topic distribution θ and word distribution β have most likely generated the documents. In applications, these are hyperparameter to be defined beforehand.

The posterior distribution of the hidden variables is in general intractable to compute, but a variety of approximate inference algorithm can be considered for LDA. These algorithms focus either on Markov chain Monte Carlo sampling (Griffiths et al. 2004) or variational inference (Hoffman et al. 2010).
We use the Online variational Bayes method (Hoffman et al. 2010), which is based on stochastic optimization and converges to a local optimum, not necessarily to a global optimum. It is shown to approximate the posterior as good as other variational Bayes methods, but converges faster and is able handle streaming document

collections. LDA is an unsupervised algorithm. If we fit it on a corpus, we obtain the approximated word distribution of every topic. With this distribution, we can infer of which topics the new document is composed of, based on its observed words. Thus, we can transform a document into a $|K|$-dimensional vector, where each position k_i represents the proportion of this topic present in the document.
This vector is further utilized as classification feature, as we assume documents of the same class to be composed of similar topics.

Topic Memory Network

The TMN is a neural network built to overcome the problems in handling short text in NLP-tasks such as TC. The basic idea of the TMN is to use indicative words for classification of new instances based in discovered latent topics in seen documents, similar to LDA. To encode the latent topic representations, memory networks (Weston et al. 2014; Graves et al. 2014) are applied.
The model consists of three components, namely a Neural Topic Model (NTM), a topic memory mechanism, and a classifier. The three components can be updated simultaneously via a joint learning process.

Formally, given $X = \{x_1, ..., x_M\}$ inputs with M short text instances. In the TMN each input is processed into two representations: A BOW vector $X_{BOW} \in \mathbb{R}^V$ and embedded sequence vector $X_{Seq} \in \mathbb{R}^L$ with V the vocabulary size and L, the chosen sequence length.
X_{BOW} is fed into the neural topic model to induce the latent topics, which are further matched with embedded X_{Seq} to learn classification features in the topic memory mechanism. For TC, representations produced by the topic memory mechanism and the embedded sequences X_{Seq} are concatenated and fed into the classifier to predict classification label y_i for x_i. The neural topic model is based on the variational autoencoder (Kingma et al. 2013), involved with a continuous latent variable $z \in \mathbb{R}^K$ as the intermediate representation, with K the number of topics. In the NTM generation, it is assumed that the corpus of documents has a topic mixture θ represented as K-dimensional distribution, created via Gaussian softmax construction (Miao et al. 2017). Similar to LDA, each topic is also described by a word distribution ϕ_k over V. x is generated as follows

- Draw latent variable $z \sim N(\mu, \sigma^2)$
- $\theta = softmax(f_\theta(z))$
- For the n-th word in x:
  - Draw word $w_n \sim softmax(f_\Phi(\theta))$

with $f_*(\cdot)$ as neural perceptron that linearly transforms inputs, activated by a nonlinear transformation. The authors proposed rectified linear units (ReLU Nair et al.

2010) as activation functions. As the prior parameter z, μ and σ are, similar to LDA, unknown. They are estimated from the input data as:

$$\mu = f_\mu(f_e(X_{BOW})), log\sigma = f_\sigma(f_e(X_BOW)).$$

To infer the approximated posterior distribution, variational inference (Blei et al. 2017) is utilized with the loss function $L_{NTM} = D_{KL}(q(z)||o(z|x)) - \mathbb{E}_{q(z)}\big[p(x|z)\big]$ with $q(z)$, a standard Normal prior, $p(z|x)$ and $p(x|z)$ as probabilities to describe encoding and decoding process.
In the topic memory mechanism, to map latent topics from the NTM to features for classification, is inspired by memory networks. Two memory matrices are designed, a source memory S and a target memory T, both with dimensionality $K \times E$, K as number of topics and E as the chosen embedding size. Two ReLU-activated neural perceptrons produce these matrices with the topic-word weight matrix $W^\Phi \in \mathbb{R}^{V \times V}$ as input. The match between the k-th topic and the embedding of the l-th word in X_{Seq} is defined by:

$$P_{k,l} = sigmoid(W^S[S_k; U_l] + b^s).$$

U references the embedded x_{Seq} here, and $[S_k; U_l]$ refers to the concatenation operation in this model. W^S and b^s are parameters to learn. To combine the instance-topic mixture θ with P, the integrated memory weights are defined as

$$\xi_k = \theta_k + \gamma \sum_l P_{k,l}$$

with γ as predefined coefficient. By weighting the target memory matrix T with ξ, we obtain R, the output representation of the topic memory mechanism as:

$$R_k = \xi_k T_k.$$

The match between latent topics and word sequences can be performed multiple times, called *hops*. The concatenation of R and U are utilized for the classification task. The model is trained via joint learning, so the entire model with its three components is updated simultaneously by defining the loss function to be updated as

$$L = L_{NTM} + \lambda L_{CLS}.$$

L_{NTM} refers to the loss of the neural topic model while L_{CLS} refers to the classification loss. λ is the trade-off parameter to control model and classifier.

3.2.5 Language Models

Formally, a language model "is a function that outputs a probability measure over strings drawn from some vocabulary" (Manning et al. 2008). A language model derives the probability $P(w_i|w_1, ..., w_{i-1})$ of a given word w_i given a sequence of previous words $(w_1, ..., w_{i-1})$.
As sophisticated word representations also require knowledge about the context of a word, it is a logical consequence that word embeddings and language models are deeply connected. The embeddings are trained like language models. In this section, we will describe at first the word2vec (Mikolov et al. 2013b) model that is the foundation of utilized models in this thesis: Paragraph Vectors and fastText.

Word Embeddings

Word Embeddings are a projection of words or phrases to vectors into numerical representations. In contrast to the previously mentioned BOW-approach, the distances between these representations provide information about linguistic similarity. It is possible to measure the similarity of words.

We will first briefly describe the two word2vec models, namely Skip-gram and Continuous bag of words, as both fastText and paragraph vectors are based on the word2vec architecture. Word embeddings are trained in language models and can be perceived as weights of the hidden layer in a neural network.

Skip-gram is an architecture for learning vector representations of words from large corpora with efficient training methods, as it does not rely on dense matrix multiplication (Mikolov et al. 2013b). Thus, it can be trained on huge text corpora. The training objective is to learn a vector representation of the target word $w_{(t)}$, which is appropriate to predict its surrounding words $w_{(t-c)}, \ldots, w_{(t-1)}, w_{(t+1)}, \ldots w_{(t+c)}$, also called window with window size n, the observed context. The architecture of the model is illustrated in Figure 7. The size of the windows is not fixed, but a window of $c = 4$ is often applied and proposed in Mikolov et al. (2013b). The objective is to maximize the average log probability

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

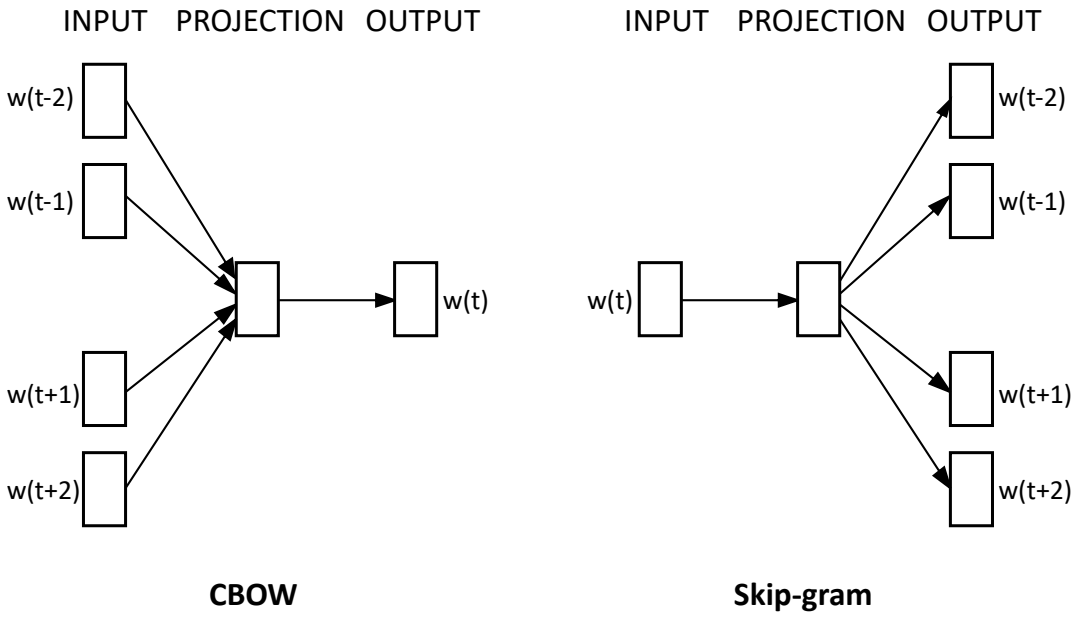


Figure 7: The word2vec model architectures, CBOW (left) and Skip-gram (right). CBOW predicts a word given its context, while Skip-gram predicts the context words given the target word (Mikolov et al. 2013a).

Continuous bag of words (CBOW), utilizes a contrary approach: Instead of maximizing the log-likelihood to predict surrounding words, the model increases the likelihood of surrounding words at predicting the target words, also visualized in Figure 7. Formally, the training objective is to predict the target word w_t based in the given words by maximizing the average log probability of the target word w_t given its window

$$\frac{1}{T}\sum_{t=k}^{T-k} log\, p(w_t|w_{t-k},\ldots,w_{t+k}). \tag{2}$$

fastText (Joulin et al. 2017) is based on the word2vec architecture. The model can be trained using either the Skip-gram or the CBOW architecture. The major difference between fastText and other embedding techniques such as word2vec is that fastText uses character n-grams. In comparison, word2vec and other embedding methods such as GloVe (Pennington et al. 2014) use words as atomic entities of a document.
In fastText, each word w is represented as the sum of its character n-grams. The word ‹apple ›is represented by the sum of the character n-gram token

$$<ap, app, ppl, ple, le>,$$

considering three as size of character n-grams. $<$ and $>$ are special boundary symbols for start and end of the word to distinguish prefix and suffix from normal n-grams.

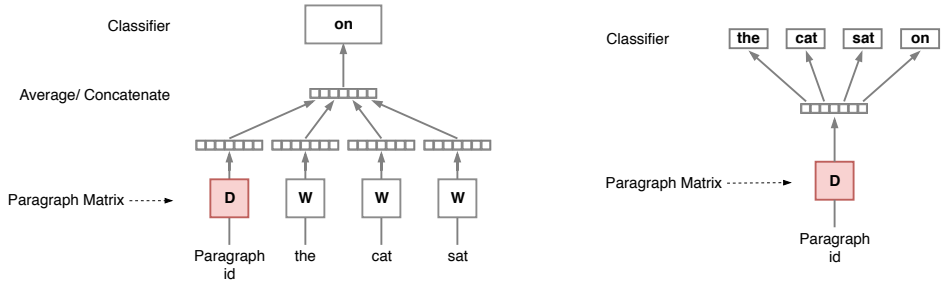
Additionally, the special sequence $< apple >$ is added with the boundary symbols to delimit it from the the sequence in other words, such as in *apples*.

The use of character n-grams provides several advantages. Assume a word w_n that is not in the training data, so an OOV token. In the BOW model it cannot be represented as vector, because if we increase the vocabulary size, all vectors have to be adjusted and the classifier has to be trained again. In word2vec, were we utilize continuous representations, we can embed the OOV, but we have no knowledge about the token and its context. Therefore, without a new training cycle, we cannot embed the token appropriately.
In practice, the word2vec model, either embed the OOV token into a sparse area of the vector space or a zero vector can be utilized to embed these words. In this way, all OOV token share a property: They have no relationship to other words in the model. The most obvious way to overcome the issue would be to train the model on the new data, but that can be unfeasible for many reasons, like the work with continuous data. In fastText, even OOV token can be embedded and yield useful information about the context of the token, if character n-grams, the word consists of, are already in the vocabulary of the model. Especially in noisy corpora, where new instances with similar characteristics such as described in Chapter 2.2) are very likely, we consider this as a big advantage.

To further use word embeddings in downstream tasks, we need all documents to have representations in the same dimensions, as most classifiers and neural networks need input in fixed dimension. As every word is embedded into an n-dimensional space, and each document consists of multiple words, simply concatenating each word representation would result in representations of different dimensionality. There are multiple ways to address this issue, we present three fundamental approaches: (i) setting a maximum sequence length. If a document contains more words, truncate the extra words. If the document is shorter, pad the missing positions e.g. zeros vectors, (ii) adding all vectors into one of the same dimension as the word embeddings or (iii) averaging all vectors into a document representation of the same dimension as the word embeddings.

Contextual Embeddings

Paragraph Vector is a model that allows us to learn contextual vector representation of sentences and longer text instances like paragraphs (here referring to an arbitrary text entity, like a sentence or a document).
Every paragraph is mapped to a unique vector, stored in a matrix D, every word is mapped to unique vector in matrix W, which are trained to represent information about their local context, similar to word2vec. Word vectors and the paragraph vector are combined (averaged or concatenated) to predict the next word in the paragraph. The paragraph vector can be thought of another word in the training process



(a) PV-DM architecture. Three words and the unique token are trained to predict the fourth word.

(b) PV-DBOW architecture. Unique paragraph token is trained to predict words of a small window.

Figure 8: Paragraph vector frameworks: 8a is based on CBOW architecture, 8b is based on the Skip-gram architecture (Le et al. 2014).

that is shared across all contexts in a paragraph, but not across paragraphs. Hence it can be described as memory of what is missing in the actual context compared to the complete paragraph. Therefore the model is named Distributed Memory (PV-DM) shown visualized in Figure 8a. Formally, the training objective is identical to Equation 2 with a slightly different composition of the context.
The word vectors are shared across paragraphs and learn to represent local context across the corpus. After the training process, the word vectors have similar properties as vectors trained with word2vec.

In the PV-DBOW model (Figure 8b), no context words are utilized as input. The model utilizes solely the unique paragraph vector to predict contexts words, which are randomly sampled from the paragraph. In this model, the word order in the paragraph is not discarded and has no influence on the paragraph vector. Thus, it is called a *distributed bag of words* model. Furthermore, the model ignores word vector computation and only updates weights of the paragraph vector. Thus, it is less memory consuming in the training process. The training objective, predicting the context of the given target word (here the paragraph vector for the complete paragraph) is similar to the Skip-gram objective in Equation 1.

Both architectures are usually trained using stochastic gradient descent. The gradient is obtained via backpropagation (Rumelhart et al. 1986).
After the training process, we can extract the unique paragraph vectors from matrix D to further utilize them in downstream tasks. To obtain paragraph vectors for new, unseen documents, the model creates new paragraph vectors as rows in matrix D and repeats the training process with gradient descent. In this *inference stage*, all other weights of the model (i.e. the classifier weights and the word vector weights) are not upgraded and remain fixed. Thus, we can embed new, unseen documents based on the learned word vectors and obtain paragraph representations.

A drawback of Paragraph Vector is its computational complexity. First, as the model requires unique paragraph vectors for every document, the number of parameters grows with the size of the training corpus can easily grow to large to be manageable, and second, compared to previous models, the inference stage to obtain representations of unseen documents is time consuming, especially compared to previous models.

BERT

To explain BERT, we first explain the attention mechanism and the Transformer architecture briefly. This architecture is based on the encoder-decoder architecture, which we use as the starting point to present the BERT model.

Encoder-decoder models

These models, often applied in sequence-to-sequence tasks, such as language translation, consist of basically two neural networks. The first, the encoder, processes an input sequence and generates a fixed-length intermediate representation. The decoder generates output from this representation. Often, RNN are used for both, encoder and decoder, e.g in an early approach by Cho et al. (2014b), shown in Figure 9. The *encoder* receives the input sequence $(x_1, \ldots, x_n)$ and compresses the first

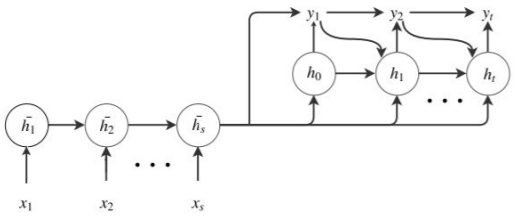


Figure 9: Encoder-Decoder Model with RNNs (Cho et al. 2014a)

word into a fixed-length vector called hidden state $\bar{h}_1$. Subsequently, for each word x_i with $i \in [2; n]$ is compressed into a hidden state $\bar{h}_i$, taking x_i and the last hidden state $\bar{h}_{i-1}$ into account. Formally, this can be described as

$$\bar{h}_i = f(\bar{h}_{i-1}, x_i)$$

with $f(*)$, a non-linear activation function. This recurrent process is applied until $i = n$. The last hidden state, $\bar{h}_s$ captures the entire input sequence into account and is called *thought vector* c.

The *decoder* now predicts a new sequence of length t from the thought vector. For each step $i \in [1; t]$, the decoder creates a hidden state h_i and an output y_i. Each following decoding considers the last output y_{i-1}, the thought vector c and the last

hidden state h_{i-1}, or formally

$$h_i = f(h_{i-1}, y_{i-1}, c).$$

Finally, the hidden state h_1 is fed into an activation function $g(x) = softmax(x)$ to output the probability distribution

$$P(y_i|y_{i-1}, ..., y_1, c) = g(h_i, y_{i_1}, c).$$

The encoder and decoder are trained jointly by minimizing the conditional log likelihood

$$min - \frac{1}{N} \sum_{i=1}^{N} logP(y_i|x_i; \Theta)$$

with N the number of training examples and $P(y_i|x_i; \Theta)$ the probability of output y_i given input x_i and a set of parameter Θ.

Attention
In encoder-decoder models, the strict separation of encoding and decoding results in a single point of communication, the fixed-length thought vector c. Regarding how humans read and understand texts, this seems not to be intuitive. Typically, we try to focus on important sentences in documents and important words sentences. From this standpoint, the described encoder-decoder model is sub-optimal, as it takes all previous words as context, even though some parts of a sequence are more connected to each other than others.
Here, the *attention* mechanism comes into place. We can differentiate between global attention that looks back in to the input sequence in the decoding step, and self-attention that weights the relevance of parts in the input. The use global attention is therefore limited to encoder-decoder models while self-attention is less restricted.

Global attention (Luong et al. 2015) means, in simple terms, that the decoder pays *attention* to parts of the encoded sequence, which can be perceived as *memory*. Between the hidden states and the output of the decoder, an intermediate state is build with an activation function of the input of the corresponding hidden state *and* the context vector of the encoder. The context vector is a weighted sum of the input states and an attention weight for each input. Formally expressed, an encoder-decoder model with word embedding input $X = (x_1, \ldots, x_i)$, the hidden states $\bar{h}_1, ..., \bar{h}_n$ of the encoder, the hidden states $h_1, ..., h_i$ of the decoder and output $y_1, ...y_t$ is extended by intermediate states $\tilde{h}_1$, formally:

$$\tilde{h}_i = tanh(W_c[c_i; h_i])$$

with W_c, a learned parameter and c_i, the context vector. So in other words, the context vector c_i and the hidden state h_i are passed into another layer with a non-linear activation function. The context vector c_i is a weighted sum of the input

states. Formally

$$c_i = \sum_{j=1}^{n} a_{ij} \bar{h}_j$$

with a_{ij}, the *attention weight*. This weight scores the j-th source state for the i-th decoding step and is defined as

$$a_{ij} = softmax(f_{attn}(h_i, \bar{h}_j)),$$

where f_{attn} is a placeholder for different attention functions (see e.g. Luong et al. 2015). Even though this addresses the single point of communication, the thought vector connecting encoder and decoder, it is still lacking a lot of context, namely all context words x_{n+i}. First, this was addressed by using a separate forward and backward encoder, often both LSTM (together called biLSTM), which output is shallowly combined in the end.

Self-attention lets the sequence attend to its complete context and weights the relevance of the respective parts of the input, which outputs are later combined. This model is therefore *not* limited to encoder-decoder models, even though it is often applied in such. Lin et al. (2017) proposed the self-attentive model for sentence embeddings. The main idea is not to apply attention to source and target states like global attention, but to weight the parts of the input according to their importance. Let $S = (w_1, ..., w_n)$ be a sequence of n word embeddings. In their model, Lin et al. compute the hidden state h at time step i as concatenation of a forward and backward LSTM, formally every hidden state is defined by

$$h_i = [\overrightarrow{h}_i; \overleftarrow{h}_i].$$

All hidden states together are defined as

$$H = (h_1, h_2, ..., h_n).$$

By linearly combining the n hidden states in H, the sequence can be transformed into a vector representation. Computing this linear combination requires the self-attention mechanism. With the input H, it outputs a single vector of weights a:

$$a = softmax(w_{s2} tanh(W_{s_1} H^T))$$

W_{s1} is a weight matrix of shape $d_a \times 2u$, where u is the hidden unit number for each unidirectional LSTM and d_a, an arbitrary hyperparameter. w_{s2} is a vector of parameters with size d_a. The hidden states H are then summed up according to the weight, provided by a for the vector representation m. Lin et al. (2017) claim, that this vector representations usually focuses on a specific component in a sentence, but, as many sentences are build of more than one component, this representation is not sufficient to represent the sentence. Thus, multiple *hops* of attention are performed

to extract r different parts of the sentence. w_{s2} is extended to an $r \times d_a$, noted as W_{s2}. Thus, the resulting vector a becomes a matrix, annotated as A. Formally,

$$A = softmax(W_{s2}tanh(W_{s_1}H^\top)),$$

with $softmax()$ as function along the second dimension of the input. The embedding vector m becomes an $r \times 2u$ embedding matrix C. Now, we get the r weighted sums by multiplying the annotation matrix A and LSTM hidden states H, resulting in the sentence embedding

$$C = AH$$

Finally, a penalization term P reduces the redundancy in A over multiple hops to ensure the model focuses on different parts of the input sequence. Formally,

$$P = ||AA^\top - I||_F^2$$

where $|| \bullet ||_f^2$ is the squared Frobenius norm of a matrix.

The Transformer

Vaswani et al. (2017) build a model solely based on attention, without any recurrence or convolution, which were the dominant approaches in many NLP-tasks. To combine self-attention and global attention is a logical consequence: The global attention mechanism encourages the decoder to increase the weight of important parts of the input sequence each output step, while the encoder does not benefit from global attention. Self-attention on the other hand aids the encoding part of encoder-decoder models.

The underlying architecture of a Transformer is also an encoder-decoder model shown in Figure 10. The encoder is composed of a stack of N sub-layers. Each of these sub-layers computes the Multi-Head Attention followed by a position-wise fully connected feed-forward network. Each of these cells is surrounded by a residual connection (He et al. 2016), combining the past result with new computations and normalizing the output. The output of the last layer is the input of the top-next layer, the output of the final layer is the decoder's input.
The decoder is of the same stack size as the encoder. The shape is also nearly identical with an additional sub-layer that performs Multi-Head Attention over the output of the encoder stack. Furthermore, the self-attention sub-layer is modified to a masked multi-head attention sub-layer; the masking in addition with the offset by one position in the output ensures that prediction for position i can only depend on outputs $< i$.

The Transformer uses two types of attention, both shown in Figure 11. Both use the concept of separated responsibilities (Daniluk et al. 2017) and thus process three

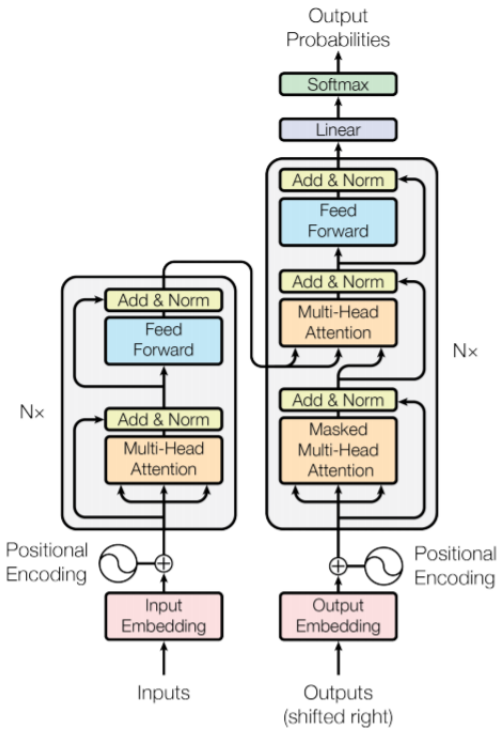


Figure 1: The Transformer - model architecture.

Figure 10: Transformer architecture (Vaswani et al. 2017)

vectors, keys $K \in d_k$, values $V \in d_v = d_k$ and queries $Q \in d_q$. The *scaled dot-product attention* is an extension of the dot-product attention, additionally scaled by the factor $\frac{1}{\sqrt{d_k}}$. It is defined as

$$Attention(Q, K, V) = softmax(\frac{QK^\top}{\sqrt{d_k}})V.$$

The process can be described as follows: Let M be the encoder's output and I the input of a Transformer layer. The global attention mechanism between encoder and decoder weights the relevance of the encoder's output ($V = M$) for the current decoding step. The assigned weights are calculated from the encoder's output ($K = M$) and the current input ($Q = I$). Other implementation use self attention, so that the values are not influenced by an output, hence $Q = K = V = I$. The Transformer model computes attention with multiple heads, so that it allows multiple hops of self-attention. Each of the Transformer layers computes a Multi-Head Attention of $Q, K,$ and V, consisting of h attention heads $head_i$, defined as

$$MultiHead(Q, K, V) = concat(head_1, \ldots, head_h)W^0$$
$$\text{with } head_i = Attention(QW_i^Q, KW_i^K, VW_i^V)$$

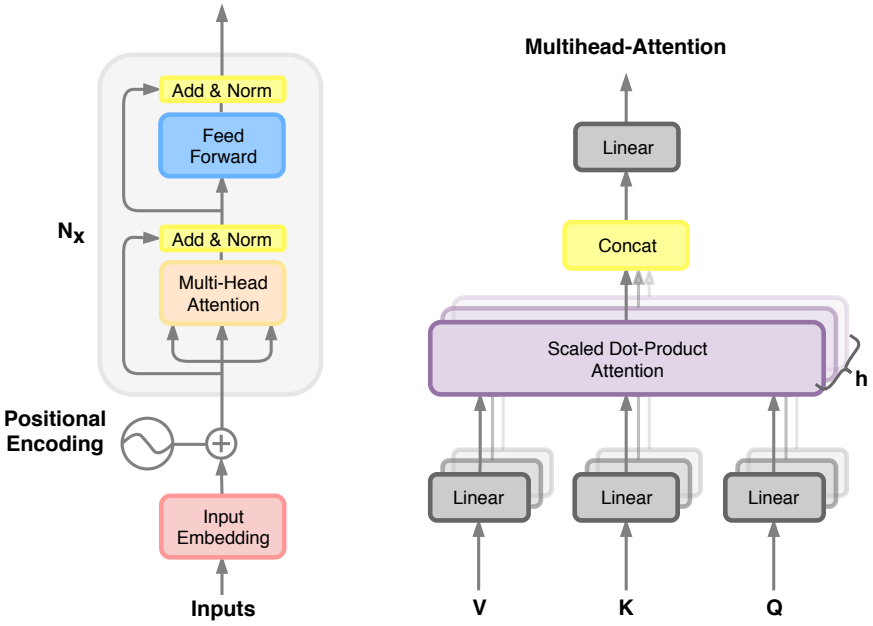


Figure 11: Multi-Head Attention in the Transformer (Vaswani et al. 2017)

where $W_i^Q \in \mathbb{R}^{d_{model} \times d_k}, W_i^K \in \mathbb{R}^{d_{model} \times d_k}, W_i^V \in \mathbb{R}^{d_{model} \times d_v}, W^0 \in \mathbb{R}^{d_{model} \times d_v}$ is a parameter matrices and therefore learned. Furthermore, the Transformer implements a method to keep in track the position of the inputs in the sequence, the *positional encoding*. There are multiple choices for these encodings (see e.g Gehring et al. 2017), Vaswani et al. (2017) applied sine and cosine functions with different frequencies to let the model learn to attend to relative positions, which are discarded using purely self-attention, despite their importance in several NLP-tasks.

The model, based solely on attention instead of recurrence and convolution performed in sequence2sequence task not only significant faster, but also achieve SOTA results in different tasks.

BERT

The model is based on the encoder architecture of the Transformer (see Figure 10). Representations are bidirectionally trained representation of words and documents, which is where it differs from other Transformer-based models, such as OpenAI GPT-2[7]. The main purpose of the language model remains to predict a token given its context. With bidirectional context, the model sees the left and the right context simultaneously in the BERT model. To ensure that the model does not see the searched token itself, which would make the prediction a trivial task in a multilayered context, it introduces the *masked language model*, based on the idea of a cloze task (Taylor 1953). 15% of the input tokens are masked and tried to predict by the

[7] `https://github.com/openai/gpt-2`

remaining words. More precise, the following rules are applied to the randomly chosen masked token

- 80% of the time replaced by the [MASK] token
- 10% of the time replaced by a random token
- 10% of the time, the token stays unchanged

Devlin et al. (2019) use the three different rules, because otherwise they would train the model excessively to predict a token that does not occur during fine-tuning and create a mismatch between pre-training an fine-tuning.
As sequences often have a connection, like two sentences in a document, and many NLP-tasks, such as Question Answering, benefit from learning connections between connected sequences (Devlin et al. 2019), the model is trained on a second task: *next sentence prediction*. Given two sentences A and B, the objective is to predict whether B is the successor of A. Therefore, 50% of the time, the real successor B gets replaced by a random sentence from the corpus.

BERT's architecture is a multi-layer bidirectional transformer encoder. The authors implemented two version, $BERT_{BASE}$ with $L = 12$ transformer layers and $h = 12$ attention heads, with a total of 110 million parameters and $BERT_{LARGE}$ with $L = 24$ Transformer layers and $h = 12$ attention heads and a total of 340 million parameters. The pre-trained models expect the input tokenized with the word-piece tokenizer (Wu et al. 2016) with a vocabulary of 30,000 unique token. The first token of a sentence must be a special [CLS] token, called the classification tokens that should learn to represent the complete first sentence. The final hidden state corresponding to this token is then utilized as sequence representation for classification tasks.
Sentence pairs are put into a single sequence, separated by a special [SEP] token between them and in the end of the sequence.

Fine-tuning BERT is – in words of the authors – straightforward since the self-attention mechanism allows to model different downstream-tasks easily by swapping to appropriate inputs and outputs. All parameters of the model are fine-tuned during the fine-tune stage. In the case of text classification, the [CLS] representation is fed into an output layer for classification, whose parameters are trained also in the fine-tuning stage. For other tasks, like named entity recognition or next sentence prediction, the input and output must be changed, as well as the final classification layer. Devlin et al. (2019) reports a short fine-tune time of only a few hours on GPU (depending on the training corpus size), which can be considered as an advantage of BERT and transfer learning in general.

3.2.6 Evaluation Metrics

To evaluate the performance of machine learning algorithms, a typically metric is the F-score, or F_1 score, which is the harmonic mean of the *precision* and *recall* of the classifier. The precision is defined as

$$precision = \frac{TP}{TP + FP},$$

where TP refers to true positive predictions, FP to the false positive predictions. The recall is defined as:

$$recall = \frac{TP}{TP + FN},$$

with FN, the false negative predicted samples of test data. The F_1 score is defined as:

$$F_1 = 2 \cdot \frac{precision \cdot recall}{precision + recall}.$$

In classification tasks with more than two classes (i.e. the binary case) and in cases unbalanced classes, there are three common approaches to calculate the F_1 score of a problem, called micro, macro and weighted. Micro calculates the metrics TP, FN, and FP globally, macro calculates them for each label independently and calculates the unweighted means, while weighted calculates it for each label and calculates their weighted average, taking into account the class imbalances. If not further mentioned, F_1 score, refers to the weighted average F_1-score.

4 Methodology

This section provides the applied methodology. We aim to automatize the classification of the answers of the OMT by training machine learning model. Thus, we train different sentence representation techniques and train them on labeled data. We further evaluate the trained models on unseen test data. To select the best hyperparameter settings, we either use a randomly selected validation set, sampled from the training data before training (for models with a long training time, i.e LDA, TMN and BERT) or use a k-fold cross-validation, i.e. we split our train set into k parts, train it on $k-1$ parts and evaluate it on the held-out data. For every set of features, we loop over all k parts as held-out dataset and average the results to improve generalization. We use $k = 5$ in this thesis.
We present considered and applied pre-processing steps, followed by the concrete implementational details and considered hyperparameter. This section is separated into three parts, first, the overall pre-processing applied to the dataset, followed by a section where we describe the machine learning models, i.e the learned and engineered document representations and SVM / MNB. The last section provides the implementational details for the deep neural networks, namely TMN and BERT, and applied hyperparameter tuning for both.

4.1 Pre-processing

As raw text has to be transformed into a numerical representation, we apply different pre-processing steps to the data to exclude unnecessary information, so-called *noise*, and make training more efficient. First, we clean up the dataset by these steps:

-Removing all empty entries from the data

-Removing entries in foreign language We utilize langdetect for python[8]. As the detection of language on short texts is not precise, we decide to apply automated language detection carefully and only apply it on documents with more than twelve tokens and manually delete shorter answers in languages other than German.

Following best practice in NLP as well as design specific decisions, we consider the following pre-processing steps with an additional note *why* the specific step might be useful. As we use different models based on different assumptions and statistics, the pre-processing steps not applied to every feature, which will be further elaborated. Table 2 shows, which steps are applied, skipped and tested for different features. Pre-processing of BERT and TMN are elaborated in their respective subsections.

-Lower casing all entries Especially in German, this will reduce the vocabulary but, at least in BOW-based models, eliminate information. Thus, we use this

[8] `https://pypi.org/project/langdetect/`

pre-processing step supervised on created features. We do not apply this step for BERT representations, as we follow the settings of the pre-trained model we utilize (see Section 4.4).

-Removal of corpus specific stop-words Removal of token with a document frequency above a given threshold. If the word is above the threshold, occurring in nearly all documents, it can not be characteristic for a specific class and therefore provides no useful information to the given task. This step has to be applied carefully, as the *quantity* of a stopword in a document can still provide useful information. Nevertheless this is a useful method to reduce the vocabulary size, so that processing can be more efficient. We do not remove language specific stop-words, as we can not preclude the information content of these words given the task. Furthermore, corpus specific stopwords might overlap with these.

-Removal of rare words Words occurring in less documents than a defined threshold. Similar to corpus specific stopwords, if a specific word occurs in very few documents (considered thresholds are < 20), the document instances are so rare that they do not have sufficient coverage to be useful. This is also called *cut off*.

-Remove all characters not in [a-zA-ZöÖüÜäÄß] This allows a focus on the text and reduces noise, at cost of distorting the original. We use regex expression here that matches all characters of the German alphabet, including umlauts and ß.

	lower casing	corpus stopwords	cut off	special char removal
BOW	e	x	x	e
tf-idf	e	x	x	e
fastText	-	x	x	x
LDA	x	x	x	x
Paragraph Vector	-	x	x	-

Table 2: Summary of the applied pre-processing steps. 'x' represents applied, 'e' is experimented with different values, '-' represents not applied.

4.2 Feature Engineering & Machine Learning

This section will explain how we derive features from the pre-processed text with the given models and furthermore provide details about the implementation of the

machine learning algorithm. The section includes BOW, tf-idf, LDA, fastText and Paragraph Vectors. The BERT and TMN settings are explained in the Sections 4.3 and 4.4, because these models combine the feature extraction and classification process. If not explicitly mentioned otherwise, a combination of features describes the concatenation of the given feature vectors.
We further classify all engineered features with a MNB classifier with $\alpha = 1$ and SVM with $C = 0.1$, following the settings of S. Wang et al. (2012). We will report the result with the highest F_1 scores in Chapter 5.2. We seek the best hyperparameter with a 5-fold cross-validation.
As an exhausting grid search is computational too expensive, we apply a greedy search. We do know that the greedy approach will not necessarily find a global optimum. The final results are evaluated on an unseen test set.

For BOW and tf-idf, we consider uni and bigram features, as S. Wang et al. (2012) reported improved results on different datasets. Larger n-grams do represent more contextual representation but tend to generalize poorly because of the resulting high-dimensional representation (Le et al. 2014). We furthermore combine these features with topic distributional features, learned with LDA similar to the Zeng et al. (2018). For the topical features, we remove all words occurring in more than 80 percent and less than 10 documents. We follow the hyperparameter start settings of Riedl et al. (2012) and use 500 model estimation iterations and 100 inference iterations and $\beta = 0.1$. We test multiple topic numbers T with $\alpha = 50/T$. Applying the TM to a document, we will obtain a $|T|$-dimensional distribution over topics of the given document. Note that LDA is a probabilistic model, thus the process is not deterministic and will produce different results, even with the same trained model.

We further utilize pre-trained word embeddings for fastText. We use 300-dimensional representations, trained with character n-grams of length 5, and a window size of 5. The model is trained in CBOW fashion (see Section 3.2.5) on a German Wikipedia dump[9] and German Common Crawl[10], together with over 60,000,000,000 tokens. For further information about the pre-training process, we refer to Grave et al. (2018). We use the pre-trained embeddings similar to Joulin et al. (2017) by summing or averaging all word vector to obtain a 300-dimensional document representation, which is further utilized as feature in the classification process. Similar to the BOW model above, this is a very efficient method to classify texts and obtain context-sensitive representation of words, once pre-trained embeddings are available. Limitations of this approach might be that positional information, so document-specific context is not considered in this model.
We further try to examine, whether this context does provide improved representation with respect to the given task and train the Paragraph Vector model according to Le et al. (2014). We train sentence vector representation with 400 dimensions for

[9] `https://dumps.wikimedia.org/`
[10] `https://commoncrawl.org/`

PV-DBOW and vector and word representation of 400 dimension for PV-DM. The optimal window size is task-dependent, thus we test different window sizes $w = [4, 8]$. Special characters are treated as normal words in the process.
As input for classification, we use the concatenation of both sentence representations, as this is reported to "often work consistently better" (Le et al. 2014).

4.3 Topic Memory Network

One reason for the development of the OMT was to reduce the time-consuming process of measuring latent motives with the TAT, as explained in Chapter 3.1. The OMT produces theoretically shorter statements of the subjects up to only keywords. Following this logic, we apply a Topic Memory Network Zeng et al. (2018) to the given task. We use the implementation of (Zeng et al. 2018)[11]. As input representation, we use fastText embeddings of 300 dimensions, identical to the described model beforehand. We set the number of hops to $H = 1$, due to computational limitations. The influence of the number of hops was discussed in (Zeng et al. 2018) and influenced the accuracy by ± 0.05, which will be considered in results.
Due to computational resources, we use 50 topics, as the results on different datasets showed the best results in Zeng et al. (2018). As sequence length, we choose 25, close to the 75% quantile in pre-processed text. We are aware that many answers will be truncated based on this decision, but accept this due to the evaluation process of the OMT. The examiners are constrained to assign a label to an answer, as soon as an indicator for a latent motive is observed. Thus, the truncation of answers might even reflect the validation process and should be appropriate in length. The training is –compared to the other models applied in this thesis– extreme resource intensive. We aim to train the model with early stopping method (Lawrence et al. 2000). For the final classification we use a CNN with a hidden size of 500.

4.4 BERT

We use the PyTorch implementation of BERT[12] with a linear layer on top of the pooled output. To use the full potential of the BERT model, we use a pre-trained model from deepset.ai[13]. The model is pre-trained on German Wikipedia dump, OpenLegalData dump[14] and news articles, about 11 GB of raw text in "cased" fashion, thus with the original capitalization of the texts. To our knowledge, this is the only pre-trained model in German. The model architecture is identical to the *base* model in Devlin et al. (2019).
We use the trained sentencepiece tokenizer[15] to tokenize texts. It is based on Google's

[11] `https://github.com/zengjichuan/TMN`
[12] `https://github.com/huggingface/pytorch-transformers`
[13] `https://deepset.ai/german-bert`
[14] `http://openlegaldata.io/research/2019/02/19/court-decision-dataset.html`
[15] `https://github.com/google/sentencepiece`

wordpiece tokenizer with a fixed-size learned vocabulary. Out of vocabulary token are split into the largest known sub-token to approximate a contextual representation. We use the German pre-trained model instead of available pre-trained multilingual, as the German model performed better in four out of five downstream tasks in German. We fine-tune the model on a 12 GB TitanX GPU.
Is input for the classification layer we use the $[CLS]$, as proposed by the authors. As the model is, compared to TMN very efficient, it is possible to train the neural network on multiple parameters. We follow the suggestions of the authors and use

- Batch size: 16, 32
- Learning rate (Adam): 5×10^{-5}, 3×10^{-5}, 2×10^{-5}
- Number of epochs: 2, 3, 4

As task specific adjustment, we use sequence lengths of 100, which includes all token in each document of the dataset.

5 Evaluation

5.1 Dataset

This section aims to provide insights into the utilized dataset. It origins from the Universität Trier in Germany and is collected and used as described in Schüler et al. (2015). The collection process followed the original paper (Kuhl et al. 1999). So each subject answered the four questions to 15 different ambiguous pictures (see Chapter 3.1). The answers are mostly in German.

	Number	Answer	Motive	Level
0	185124138106620081063215	ignoranz den anderen gegenüber.schlecht.die Pe...	M	5
1	185124138106620081063215	mitlachen, mit eingeschlossen zu werden.sie la...	A	5
2	185124138106620081063215	den anderen Umamen, ehrlichkeit, vertrauen.gut...	A	1
3	52122529750377312346781011	Sie hält die andere Person, stütz sie. Gut.Sie...	M	1
4	52122529750377312346781011	Entspannung, Spaß zu haben. Sie albern herum. ...	A	2

Table 3: Extract from the unprocessed dataset

The dataset consists of 220,000 documents. After filtering out answers in other languages than German, answers containing only the same letter repeated, and empty entries, the dataset consists of 209,399 values. Each entry is composed of (i) the anonymized number of the participant, (ii) the concatenated four answers, (iii) the motive, and (iv) the affective level as shown in Table 3. The creators utilized a slightly different evaluation method than Kuhl et al. (1999) proposed, as they perceive freedom (F) as distinct motive. In Kuhl et al. (1999), the freedom motive is not a distinct motive. Scheffer, the co-author of the OMT, perceives its characteristic as real subset of the power motive[16]. Thus, the freedom motive is changed to power for generalization aspects. As we classify solely the motive, not the affective levels, there is no need to adjust the affective level as well. We further refer to the motives also as M (power), L(achievement), and A (affiliation). The abbreviations are utilized in the labeling process and refer to the respective German translation.

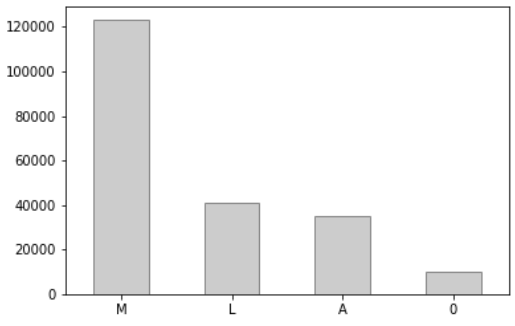


Figure 12: Label distribution of the dataset

The length of the answers differs from $1 - 97$ with a mean length of 22 words and a standard deviation of 12.03. A document containing one word is possible, as there is no restriction on how many answers a participant wants to give and might be

[16]Personal communication, 23.09.2019

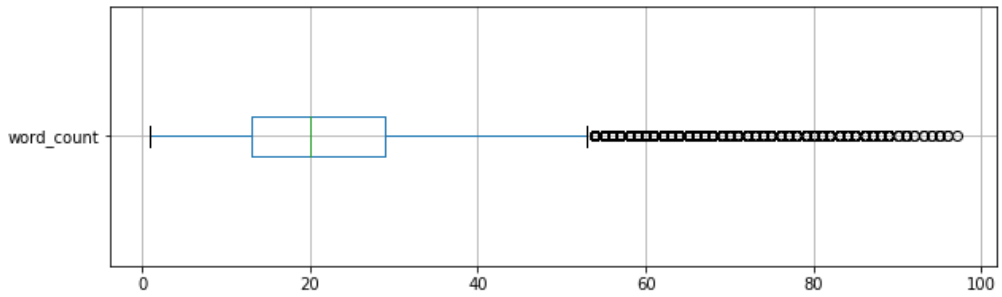


Figure 13: Box plot of the text length.

enough to attribute a motive to. Figure 13 illustrates more about the statistics of the corpus. As the figure reveals, the 75% quantile is at 29. The box plot also shows many outliers to the right side, but in general, the answers are short. We decided not to remove the very short answers, as we do not know the exact influence of the length of answers on the classification by human experts. Additionally, the 0 motive is introduced especially for answers, which could not be assigned to any of the other motives, therefore these special cases are not disruptive to the task. The distribution of the classes, shown in Figure 12, is imbalanced. The M motive is by far the most frequent class. Thus we decided to use a stratified split on the train and test set. The split is chosen by 70% of data as training set, leading to 30% as test set. In numbers, there remain 146579 answers in the training set, 62820 in the test set.

The imbalance of classes also result in the baseline with a zeroR classifier, attributing always the most common label. Another aspect, differing from the majority of NLP research, is the use of German. German uses capitalization for nouns as well as three genera for nouns and a moderate degree of inflection. In short texts, especially capitalization is often ignored. Table 3 illustrates the beginning of 5 answers, in which a German native speaker can identify numerous spelling mistakes (e.g. *Umamen* in third line is, spelled correctly *umarmen / to hug*, *ignoranz* is correctly *Ignoranz / ignorance*.
Furthermore, we will append ten instances of the dataset in the appendix.

5.2 Results

This section will present the results of the experiments described in Chapter 4. A discussion of the results follows in the Chapter 6. For simplicity, we refer to the combination of feature and classifier as model. The results presented in Table 4 are the best results achieved by the individual models. Each result is the mean of multiple runs to increase confidence in replicability. On the given task, the pre-trained BERT model performed best out of all models. The best results were obtained training all parameters for two epochs with a learning rate of $1e-5$, slightly lower than proposed in Devlin et al. (2019).

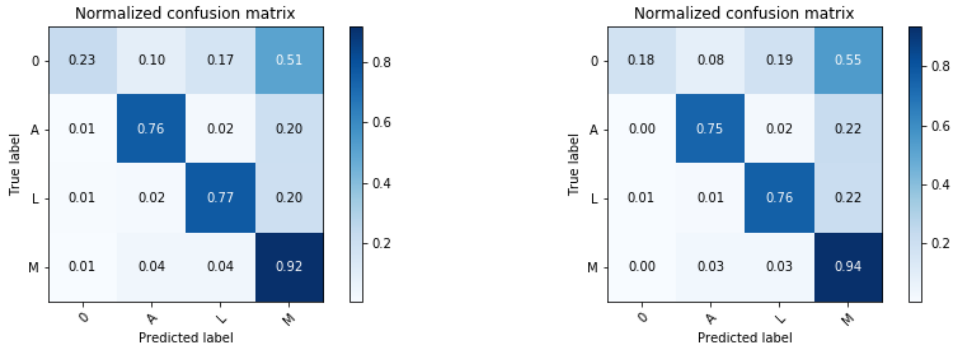
Table 4: Results

Model	F_1	Settings
zeroR	0.5886	
LMT[17]	0.801	
BOW - SVM	0.8308	cut off=10, unigrams
TFIDF - SVM	0.8354	cut off=10, uni & bigrams
LDA - SVM	0.7473	topics = 80, alpha = 50 / 80
fastText - SVM	0.7911	averaged, 300 dimension
BOW + LDA - SVM	0.8223	see above
Paragraph Vector	0.8109	window size =[8,4]
BERT	**0.8443**	Learning rate = $1e-5$, epochs = 2

Also all features classified with SVM achieved a higher F_1 score than the identical features with MNB, thus we consider the the SVM inferior on this dataset. As we applied many different techniques to classify the answers of the OMT, we further list the models and report specific observations on each of the models.

BOW, tf-idf: The models performed third respective second best with very similar results. Both models benefit from noise removal, where especially the minimum document frequency of terms is the most important, but also lower casing and the removal of special characters improved the performance.
Figure 14 shows confusion matrices for both models and reveals that both models tend to classify answers to the M motive that is by far the most occurring class. Even though the BOW-model achieved a lower F_1 score, it tends to classify the motives L, A, and 0 with slightly higher accuracy on the cost of losing accuracy on the M motive. The accuracy decreases with the number of occurrences for each label.

LDA Topic Model: Of all features tested, the LDA topic models performed worst and had a negative influence combined with BOW features.
Of all tested numbers of topics, the model with $T = 80$ topics performed best. We observed similar results in the classification task with other parameters, especially with $T < 80$ by adjusting the α parameter at the cost of higher variance in the F_1



(a) Confusion matrix of the test set results of BOW features classified with SVM

(b) Confusion matrix of the test set results of tf-idf features classified with SVM

Figure 14: Confusion matrices of the count-based features classified with SVM. All results are normalized.

score (see Chapter 6).
Even though the F_1 scores vary a lot, all models share that the 0 motive is not predicted.
As LDA is an unsupervised model, it is also possible to fit it additionally on the test set, or in real life examples on new instances to fit the model to all data. Interestingly, training the TM on all available data does not increase the classification performance nor increase stability of the F_1 scores. Due to the significant weaker performance, we do not apply methods to increase stability of the results (see e.g. Riedl et al. 2012).

fastText: The difference between the applied sentence representation strategies (i.e. summing up and averaging all word vectors) is very small, considering the other results, with a slightly better score for the averaged sentence representation (0.7883 compared to 0.7911). Nevertheless, the scores are significant lower than the scores achieved by the tf-idf and BOW models.

OOV-token	Nearest neighbor	similarity	Translation
diszplin	Disziplin	0.5380	discipline (f), discipline
Körpersignalen	Sensorsignalen	0.6970	body signals, sensor signals
Behüterrolle	Behüter	0.6215	role of a guardian, guardian
gesprächsleitung	Gesprächsleitung	0.6468	moderation (f), moderation
aushegen	R07	0.5435	? , non-word

Table 5: Overview of OOV token embeddings. '(f)' represents minor spelling mistakes in the OOV token, '?' a non-interpretable token, and 'non-word' an unknown word, most likely a proper noun.

Out of a total of 70,000 unique tokens, 23,000 are OOV token. All token are embedded based on their character n-gram composition, including all misspelled words.

Table 5 shows examples for the OOV embeddings, which sometimes, especially on slightly misspelled words (e.g. swapped letters in a word) and ignored capitalization.

The confusion matrix (Figure 15) of the model shows that the it overfits to the M motive and tend to classify to many documents in to that class.

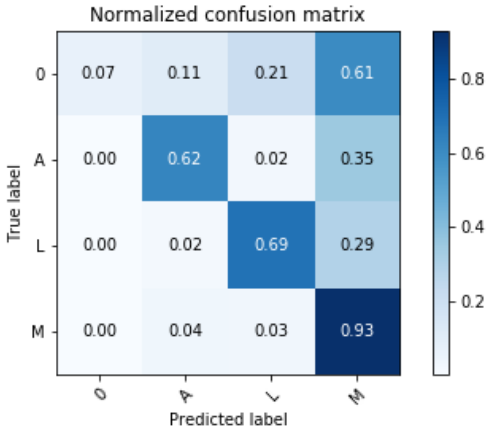


Figure 15: Confusion matrix of the fastText model. The right column reveals that the model classifies many documents to the M class and therefore shows poor results for the other classes, especially 0

Paragraph Vector: The model performed best when concatenating both implementations, PV-DBOW and PV-DM. Using both models individual, the PV-DBOW model is superior by a large margin (0.8088 to 0.6984). Unlike in the combination of LDA and BOW, the combination of both model perform better when combined. DBOW shows best results using a window size of 8, while the DM model results with a window size of 4 are the best.

BERT: The pre-trained deep neural network model is the best out of all tested models. The results of all applied learning rates peaked after two iterations before dropping lower than after the first iteration. As we observed this behavior especially on the higher learning rates, we additionally used 1×10^{-5} as learning rate and reported best results. The best set of hyperparameter is training for two epochs and a learning rate of $1e-05$. Figure 16 shows the confusion matrix of the model. The right column indicates, that the model is likely to classify to many documents as M motive. The effect in this model is lower compared to all other model, at the cost of a lower accuracy on the M motive. Even though the accuracy in the 0 class is still beneath 50%, the score is about twice as good as the second best accuracy in this class. Also the accuracy on A and L is 5% higher than in the second best model (tf-idf).

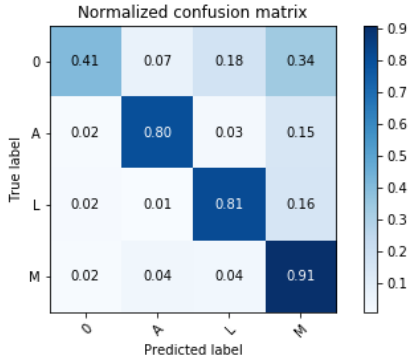


Figure 16: Confusion matrix of the test set results of the BERT model

TMN: On the given hardware setup, we are not able to train the TMN sufficiently and therefore report validation set results, shown in Table 6. The model trained for 250 epochs, but we can not ensure convergence, thus these results have to be taken with care. It is important to notice that the model consists of two networks that are trained in a joint learning process. Thus, not the complete network trained for 250 epochs. Most iteration are for improving the topic representation in the NTM part of the model. Additionally, the final classification input is the concatenation of the sequence embedding and the topic induced document representation. Thus, it is up to future work du test the complete model and to test classification of embedded sequences with CNN and RNN.
Nevertheless, to this points results are inferior to the results of BERT as well as the tf-idf features classified by SVM.

Table 6: F_1 score on validation set

Model	F_1	Acc
TMN	0.824	0.8354
BERT	**0.848**	**0.847**

6 Discussion

This chapter we discuss the results of the experiments. We split this section into three parts. In the first part we discuss general results of the classification of the answers of the OMT, in the subsequent section we focus on the classification with TM and BERT.

6.1 OMT Classification

The results of Section 5.2 show success in the goal of this work, the classification of the answers of the OMT. All applied methods perform substantially better than the the zeroR classifier, most of them additionally improved the results of Johannßen et al. (2019). The BERT model achieved SOTA performance.
Nevertheless, even though the F_1 scores are satisfying compared to the previous results, the accuracy on the A, L, and especially the 0 motive are poor. This most likely has two reasons: (i) The imbalance of the data: More than 50% of the answers of the OMT are classified as M, thus classifier tend to classify more answers into the majority class. (ii) The 0 class, or 0 motive are the answers, which can not be attributed to another motive. Thus, it can be seen as residue of the other classes and contains all *meaningless* answers such as "i don't know" repeated multiple times, and stories without connections to the stimuli. So it is likely that there is no real characteristic for a 0 motive answer, what would explain the low accuracy shown in the confusion matrices in the previous chapter.
An analysis of the wrong classified documents revealed, that the length of all false classified answers is very similar to the correct classified answers, shown in Table 7 as well as the mean number of spelling mistakes[18]. The only difference are the false positives for the 0 class: There, the average length was only 14, compared to 22 on the corpus. As these includes only a very small fraction of the set (e.g. 230 documents in tf-idf - SVM model).

Group	∅ length	∅ spelling mistakes
Correctly labeled	22.22	0.3593
Wrongly labeled	21.86	0.3759

Table 7: Statistics from the test corpus

For a human reader who is not trained to classify the answers of the OMT, there are no latent characteristics observable in false classified answers indicating *why* these answers are not correctly classified. It seems like the answers tend be too similar to be classified with higher F_1 score.

[18]Measured with the count of spacy (`https://spacy.io/`) OOV token.

The corpus contains the typical high percentage of rare occurring words (Zipf's law, Zipf (1929)). More than 55% or 40,000 out of 70,000 token of the train corpus vocabulary occur in less than three documents. Another 10,000 occur in under ten documents. These rare words contain a high amount of misspelled words, which are likely to be in the corpus, as well very context specific words. For the count-based (i.e. BOW, tf-idf) models, minor improvements could be achieved by using a proper spell correction, which should increase word co-occurrences As all count-based models benefit from lower casing, this might be useful as well.
Some participants seem to answer the questions to a presented stimuli by referring to the specific situation in which the seen stimuli plays in their imagination. An example of this is "Architekt (architect)". A related document with the word Architekt is

> Der Architekt legt seine Gründe in Form von Argumenten dar warum er jenes Gebäude so und nicht anders realisieren möchte. Angespannt. Er möchte den Auftraggeber überzeugen und befürchtet zugleich nicht verstanden zu werden. (The architect explains his reasons in the form of arguments why he wants to realize that building so and not otherwise. Strained. He wants to convince the client and feared at the same time not to be understood).

The participant describes the feelings of the acting person in the stimuli projected into the specific context of an architect at work. As we do not have the presented stimuli present we are not be able to further analyze specific answers like this, as we can not determine the content of the stimuli. This makes it impossible to conclude, whether this kind of answers are influenced by the stimuli or by participant-specific situations, like Chapter 3.1 described. Considering the rare occurrences of the token "architekt", it is unlikely that the stimuli presented architectural stimuli like drawings from buildings. This would support the theory of operant tests but cannot be verified with the given data, but is also not the objective of this thesis.

Our expectation, that the fastText embeddings may yield better classification results based on their ability to express relatedness between terms did not materialize. We suspect three possible reasons for this, (i) the 300 dimensional vector representation of documents is not sufficient to represent information at least on par with de count-based methods, (ii) the pre-trained embeddings need further, domain or corpus specific fine-tuning be able to represent terms sufficient for the task and (iii) in the context of the OMT a strict discrimination of terms is useful. (i) and (ii) can be implemented and tested in future work, by fine-tuning and/or using sequential embeddings and deep neural networks such as CNN for classification.
(iii) might be a limitation of this approach in general, as it can be important in this context to discriminate related words based on minor changes. In our pre-trained model, the most similar term to "will (to want something)" is "möchte (would like

to)". In many context this measurable similarity is very useful, for our use case maybe not, as "He would like to..." is more passive and less aggressive than "He wants to...", which can be a reason for the poor performance. Most likely, a combination of all three reason influences the model to some degree and should be be further investigated. In the applied method, the fastText embeddings classified with SVM are ineffective.
Even though the Paragraph Vector model performed better than the LMT, it is still not preferable to the simpler and more efficient count-based models. Considering training time for the model in combination with the F_1 scores, we would recommend to use BERT, which trains slightly longer but achieves best results or to use tf-idf in combination with SVM, which trains more than 30 times faster and maintains interpretability. Noticeably is also, that the PV-DBOW model solely achieved an F_1 score of 0.8088 whilst the PV-DM only achieved 0.6984 and is the worst of all applied model. Unlike in the combination of BOW and LDA features, the combined model performs better than both model standalone. We append confusion matrices of the models in the appendix.

6.2 OMT Classification with Topic Models

As mentioned in Chapter 5.2, we focus on LDA and overlook the results of the TMN, because we have not sufficient data to analyze it.
Compared to the other applied models, LDA performed bad in the given task. Considering the mean length of the texts, it is not very surprising that LDA as single feature does not achieve good results, considering we represent the documents with just T dimension. Interestingly the performance of BOW combined with LDA dropped, to when combining both. Via weighting of the features, we can minimize this effect, but only approach the performance of the BOW model solely. We conclude that a LDA TM does not yield information that goes beyond a BOW model considering the classification task. The reported best F_1 for LDA score can nearly be achieved in multiple ways. Following Riedl et al. (2012) and using $\alpha = 50/T$, we achieve an score of 0.7473 with a standard deviation of 0.0082, which is not only the best result but also the result with the lowest deviation over five runs with identical hyperparameter. As Figure 18a reveals with this dynamic setting of the *alpha*, other topic numbers perform slightly weaker with

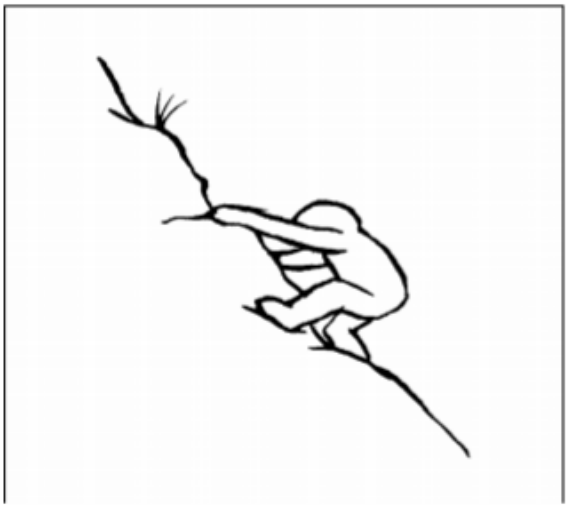
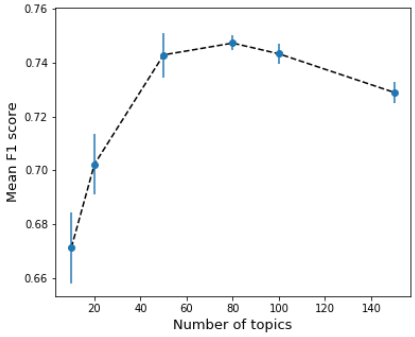
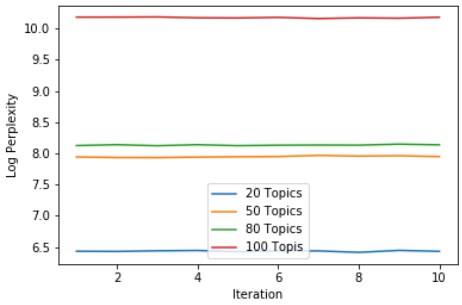


Figure 17: Example stimulus from Kuhl et al. (1999), possibly related to climbing

a higher deviation. We can achieve similar performance with other topic numbers by adjusting the alpha parameter. E.g setting the topic number to only 20 topics with an *alpha* of 1 achieves similar F1 scores but suffers from a very high standard deviation over different runs. This behavior shows the probabilistic nature of LDA topic models, where variations can have a high influence on the resulting model. We achieve scores up to 0.76, which were best results with the model. Thus, reported LDA results should always be normalized over multiple runs.
In Figure 18b we additional report normalized perplexities of ten runs. We observe that our best model with 80 topics has not the best perplexity, which shows that perplexity is not necessarily useful to measure the quality of LDA topic models with respect to a TC task. Beside the problem of to few co-occurrences in short text, the model might suffer from another issue that is task specific. Participants tend do describe the observed stimuli, observable in Topic 19 from Table C1 in Appendix. The top words are berg, den, kommt, oben, halt, kommen, angestrengt, gipfel, hoch (mountain, the, comes, top, stop, coming, exhausted, summit, high) that are clearly correlated to mountain climbing, which is likely to be visible in the stimuli. Kuhl et al. (1999) present example stimuli like Figure 17, which could be the stimuli to Topic 19. Thus, we maybe found a topic that describes the stimulus, but does not capture task specific information to classify the answers of the OMT sufficiently.



(a) LDA results with different T and $alpha = 50/T$



(b) Stability of perplexity over 10 runs visualized. We use $alpha = T/50$

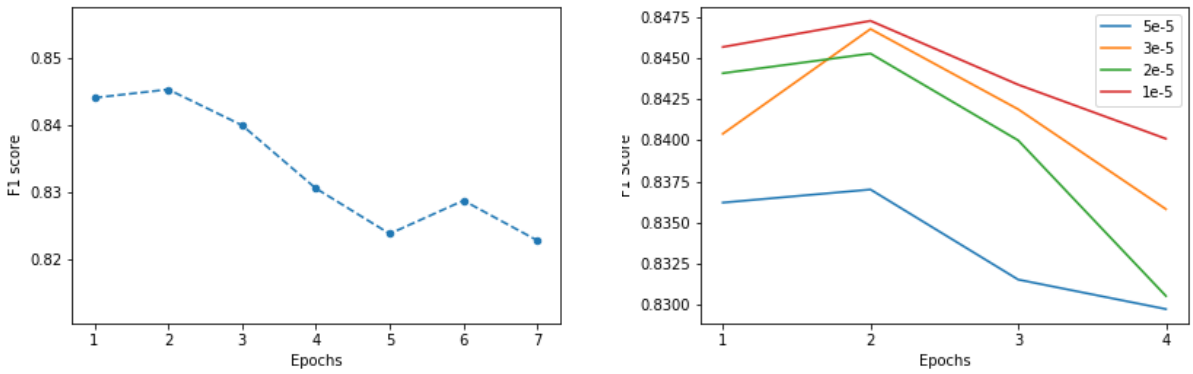
Figure 18: Statistics from the LDA topic model

In future research, it can be useful to observe whether particular topics are stronger correlated to stimuli. Then, we will be able to determine, whether the model learns which stimulus is related to which motive or whether our model learns to classify language specific characteristics to a latent motive. Furthermore, the top words of each topic are very difficult to analyze. E.g Topic 46 contains top words like "mächtig (mighty)", "eingeschüchtert (intimidated)", "überlegenheit (superiority)" and could be linked to the power motive M. Topic 70's top words seem to be correlated to the L motive, containing "leistung (performance)", "anerkennung (appreciation)" and "jubeln (cheer)". Topic 0 covers affiliation related terms like "einander (each other)",

"vertrauen (trust)" and "verantwortung (responsibility)". Many other topics contain similar words that can also be considered as stopwords and yield no information with respect to the task for a human reader but are not captured by the pre-processing methods, where further improvements might be possible.

6.3 OMT Classification with BERT

Similar to many other NLP-tasks, BERT achieved best results considering the F_1 score. The result is very close to the pairwise human agreement score of 0.85 with an accuracy of 0.8460, thus it is hard to argue whether further improvements are possible.



(a) Decreasing F_1 score over time (b) F_1 score of different learning rates over time

Figure 19: BERT model statistics

We observed strong signs of overfitting during the training, as statistics from Figure 19 reveal. For every applied learning rate, the validation set accuracy decreases after 2 epochs and did not recover to former scores over time. Considering that Devlin et al. (2019) recommend only 2-4 epochs, this is maybe typical behavior on small

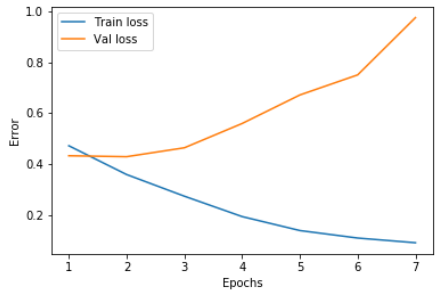


Figure 20: Training and validation loss over time

datasets considering the huge amount of pre-training data and 110,000,000 parameter to adjust. Not only the validation loss is increasing over time, the training loss is also converging. For further analysis and usage of the model, sophisticated and exhaustive fine-tuning should be applied, so that it is (i) maybe possible to slightly improve F_1 and accuracy score and (ii) increase stability of the model. In this thesis, we consider the training successful, as we performed reproducible SOTA results within the recommended range of fine-tuning epochs. Nevertheless, we are aware of multiple methods that would increase stability and maybe performance of the model, namely (i) increasing the amount of training data, (ii) exhaustive hyperparameter tuning, including different batch sizes, maximum sequence length, optimizer and learning rate schedules and (iii) using the BERT Large model with 24 Transformer blocks, 1024 hidden layers and 16 attention heads, resulting in 340 million parameter to train.

7 Conclusion

The use of machine learning to classify the answers of the OMT into latent motives shows to be very effective. Thus, we can confirm our hypothesis and established a new benchmark with the BERT model.

We introduced a dataset of answers to the OMT with difficult properties for a TC task. The classes are highly imbalanced, single answers suffer noise such as abbreviations, spelling mistakes and are in general short. We encountered this with different pre-processing and text representations in combination with different classification methods. We showed that simple count-based document representations classified with a SVM yield very promising results, even better than more sophisticated sentence representations and word embeddings.
Despite an improvement compared to the zeroR classifier, we consider LDA topic model not as a useful feature to classify the answers of the OMT, even though the resulting topic models showed tendencies to model motive-related terms together.

Furthermore we showed the effectiveness of a pre-trained attention based deep neural network for classifying a psychometric test, trainable in considerable time and with affordable hardware[19]. We found these results are in accordance with actual research results in downstream NLP-tasks, where closely related models performed SOTA performance on a wide range of tasks.

With a final F_1 score of 0.84 we substantial improved previous results and approach the human performance, measured in human intraclass correlation coefficient of 0.85. These metrics are not directly comparable but are to our knowledge the best measurement of human performance.

For further analysis of the texts and characteristics, we found the classification with BERT significantly less useful than other techniques, e.g. BOW models and the LMT trained by Johannßen et al. (2019), which enable a better analysis of classified texts and should also be considered in further human motive related research. Especially Johannßen et al. (2019) could detect the most influential features in the classification process, while BERT remains a black box model.

[19] Sufficient hardware setup is available at `https://colab.research.google.com/` for free

References

Allahyari, Mehdi, Seyedamin Pouriyeh, Mehdi Assefi, Saied Safaei, Elizabeth D. Trippe, Juan B. Gutierrez, and Krys Kochut (2017). "A brief survey of text mining: Classification, clustering and extraction techniques". In: *arXiv preprint arXiv:1707.02919*.

Atkinson, John W. (1958). *Motives in fantasy, action, and society: A method of assessment and study*. Princeton, N.J: Van Nostrand.

Baumann, Nicola, Reiner Kaschel, and Julius Kuhl (2005). "Striving for unwanted goals: stress-dependent discrepancies between explicit and implicit achievement motives reduce subjective well-being and increase psychosomatic symptoms." In: *Journal of personality and social psychology* 89.5, p. 781.

Bengio, Yoshua, Réjean Ducharme, Pascal Vincent, and Christian Jauvin (2003). "A Neural Probabilistic Language Model". In: *Journal of Machine Learning Research* 3.2, pp. 1137–1155.

Bengio, Yoshua, Ian Goodfellow, and Aaron Courville (2017). *Deep learning*. MIT Press.

Blei, David M., Alp. Kucukelbir, and Jon D. McAuliffe (2017). "Variational inference: A review for statisticians". In: *Journal of the American Statistical Association* 112.518, pp. 859–877.

Blei, David M., Andrew Y. Ng, and Michael I. Jordan (2003). "Latent Dirichlet Allocation". In: *Journal of Machine Learning Research* 3.1, pp. 993–1022.

Bojanowski, Piotr, Edouard Grave, Armand Joulin, and Tomas Mikolov (2017). "Enriching Word Vectors with Subword Information". In: *Transactions of the Association for Computational Linguistics* 5.1, pp. 135–146.

Breiman, Leo (2001). "Random Forests". In: *Machine learning* 45.1, pp. 5–32.

Budhkar, Akshay and Frank Rudzicz (2018). "Augmenting word2vec with latent Dirichlet allocation within a clinical application". In: *arXiv preprint arXiv:1808.03967*.

Chen, Qiuxing, Lixiu Yao, and Jie Yang (2017). "Short text classification based on LDA topic model". In: *ICALIP 2016 - 2016 International Conference on Audio, Language and Image Processing - Proceedings*. Prague, Czech Republic, pp. 749–753.

Chen, Stanley F. and Joshua Goodman (1999). "An empirical study of smoothing techniques for language modeling". In: *Computer Speech & Language* 13.4, pp. 359–394.

Cheng, Xueqi, Xiaohui Yan, Yanyan Lan, and Jiafeng Guo (2014). "BTM: Topic modeling over short texts". In: *IEEE Transactions on Knowledge and Data Engineering* 26.12, pp. 2928–2941.

Cho, Kyunghyun, Bart Van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio (2014a). "On the Properties of Neural Machine Translation: Encoder–Decoder Approaches". In: *Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*, pp. 103–111.

Cho, Kyunghyun, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio (2014b). "Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation". In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*. Doha, Qatar, pp. 1724–1734.

Conneau, Alexis, Holger Schwenk, Loïc Barrault, and Yann LeCun (2016). "Very Deep Convolutional Networks for Natural Language Processing". In: *arXiv preprint arXiv:1606.01781 2*.

Cortes, Corinna and Vladimir Vapnik (1995). "Support-Vector Networks". In: *Machine Learning* 20.3, pp. 273–297.

Daniluk, Michal, Tim Rocktäschel, Johannes Welbl, and Sebastian Riedel (2017). "Frustratingly Short Attention Spans in Neural Language Modeling". In: *arXiv preprint arXiv:1702.04521*.

Danisman, Taner and Adil Alpkocak (2008). "Feeler: Emotion classification of text using vector space model". In: *AISB 2008 Convention Communication, Interaction and Social Intelligence*. Vol. 1. Aberdeen, Scotland, pp. 53–59.

Devlin, Jacob, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova (2019). "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding". In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Vol. 1. Minneapolis, MN, USA, pp. 4171–4186.

Dumais, Susan T., George W. Furnas, Thomas K. Landauer, Scott Deerwester, and Richard Harshman (1988). "Using Latent Semantic Analysis To Improve Access

To Textual Information". In: *Proceedings of the SIGCHI conference on Human factors in computing systems - CHI '88*. Washington D.C, USA, pp. 281–285.

Elman, Jeffrey L. (1990). "Finding structure in time". In: *Cognitive science* 14.2, pp. 179–211.

Freund, Yoav and Robert E. Schapire (1996). "Experiments with a New Boosting Algorithm". In: *Proceedings of the Thirteenth International Conference on International Conference on Machine Learning*. Vol. 96. Bari, Italy, pp. 148–156.

Gehring, Jonas, Michael Auli, David Grangier, Denis Yarats, and Yann N. Dauphin (2017). "Convolutional sequence to sequence learning". In: *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. Sydney, Australia, pp. 1243–1252.

Gildea, Daniel and Thomas Hofmann (1999). "Topic-based language models using EM". In: *Sixth European Conference on Speech Communication and Technology*. Budapest, Hungary, pp. 2167–2170.

Grave, Edouard, Piotr Bojanowski, Prakhar Gupta, Armand Joulin, and Tomas Mikolov (2018). "Learning word vectors for 157 languages". In: *arXiv preprint arXiv:1802.06893*.

Graves, Alex, Greg Wayne, and Ivo Danihelka (2014). "Neural turing machines". In: *arXiv preprint arXiv:1410.5401*.

Griffiths, Thomas L. and Mark Steyvers (2004). "Finding scientific topics". In: *Proceedings of the National Academy of Sciences* 101.1, pp. 5228–5235.

Griffiths, Thomas L., Mark Steyvers, David M. Blei, and Joshua B. Tenenbaum (2005). "Integrating topics and syntax". In: *Advances in Neural Information Processing Systems*. Vancouver, Canada, pp. 537–544.

Hayes, Phillip J., Peggy M. Andersen, Irene B. Nirenburg, and Linda M. Schmandt (1990). "TCS: a shell for content-based text categorization". In: *Sixth Conference on Artificial Intelligence for Applications*. Santa Barbara, CA, USA, pp. 320–326.

He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun (2016). "Deep residual learning for image recognition". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778.

Heckhausen, Jutta and Heinz Heckhausen (2005). *Motivation und Handeln: Einführung und Überblick*. Springer-Verlag.

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

Hofer, Jan, Athanasios Chasiotis, Wolfgang Friedlmeier, Holger Busch, and Domingo Campos (2005). "The measurement of implicit motives in three cultures: Power and affiliation in Cameroon, Costa Rica, and Germany". In: *Journal of Cross-Cultural Psychology* 36.6, pp. 689–716.

Hoffman, Matthew, Francis R. Bach, and David M. Blei (2010). "Online Learning for Latent Dirichlet Allocation". In: *Advances in Neural Information Processing Systems*. Vancouver, Canada, pp. 856–864.

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

Hong, Liangjie and Brian D. Davison (2010). "Empirical study of topic modeling in twitter". In: *Proceedings of the first workshop on social media analytics*. Washington D.C., US, pp. 80–88.

Ikonomakis, M., S. Kotsiantis, and V. Tampakas (2005). "Text Classification Using Machine Learning Techniques". In: *WSEAS Transactions on Computers* 4.8, pp. 966–974.

Johannßen, Dirk and Chris Biemann (2018). "Between the Lines: Machine Learning for Prediction of Psychological Traits - A Survey". In: *International Cross-Domain Conference for Machine Learning and Knowledge Extraction*, pp. 192–211.

Johannßen, Dirk, Chris Biemann, and David Scheffer (2019). "Reviving a psychometric measure: Classification and prediction of the Operant Motive Test." In: *Proceddings of CLPsych 2019*. Minneapolis, MN, USA, pp. 121–125.

Joulin, Armand, Edouard Grave, Piotr Bojanowski, Matthijs Douze, Hérve Jégou, and Tomas Mikolov (2016). "Fasttext. zip: Compressing text classification models". In: *arXiv preprint arXiv:1612.03651*.

Joulin, Armand, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov (2017). "Bag of Tricks for Efficient Text Classification". In: *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*. Valencia, Spain, pp. 427–431.

Kim, Sang-Bum, Hae-Chang Rim, Dong-Suk Yook, and Heui-Seok Lim (2002). "Effective Methods for Improving Naïve Bayes Text Classifiers". In: *Pacific Rim International Conference on Artificial Intelligence*. Tokyo, Japan, pp. 414–423.

Kim, Sunghwan M., Alessandro Valitutti, and Rafael A. Calvo (2010). "Evaluation of unsupervised emotion models to textual affect recognition". In: *Proceedings of the NAACL HLT 2010 Workshop on Computational Approaches to Analysis and Generation of Emotion in Text*. Los Angeles, CA, USA, pp. 62–70.

Kingma, Diederik P. and Max Welling (2013). "Auto-Encoding Variational Bayes". In: *arXiv preprint arXiv:1312.6114*.

Knerr, Stefan, Léon Personnaz, and Gérard Dreyfus (1990). "Single-layer learning revisited: a stepwise procedure for building and training a neural network". In: *Neurocomputing: Algorithms, Architectures and Applications*. Fogelmann, pp. 41–50.

Kowsari, Kamran, Kiana Jafari Meimandi, Mojtaba Heidarysafa, Sanjana Mendu, Laura Barnes, and Donald Brown (2019). "Text classification algorithms: A survey". In: *Information* 10.4, pp. 2078–2489.

Kuhl, Julius and David Scheffer (1999). *Der Operante Multi-Motiv-test (OMT): Manual [The Operant Multi-Motive-Test (OMT): Manual]*. Universität Osnabrück.

Kuhl, Julius, David Scheffer, and Jan Eichstaedt (2003). "Der Operante Motiv-Test (OMT): Ein neuer Ansatz zur Messung impliziter Motive". In: *Diagnostik von Motivation und Selbstkonzept*, pp. 129–149.

Lawrence, Steve and C. Lee Giles (2000). "Overfitting and neural networks: Conjugate gradient and backpropagation". In: *Proceedings of the IEEE-INNS-ENNS International Joint Conference on Neural Networks. IJCNN 2000. Neural Computing: New Challenges and Perspectives for the New Millennium*. Como, Italy, pp. 114–119.

Le, Quoc and Tomas Mikolov (2014). "Distributed representations of sentences and documents". In: *International conference on machine learning*. Beijing, China, pp. 1188–1196.

LeCun, Yann, Léon Bottou, Yoshua Bengio, and Patrick Haffner (1998). "Gradient-based learning applied to document recognition". In: *Proceedings of the Institute of Electrical and Electronics Engineers*. Vol. 86. 11. Pasadena, CA, US, pp. 2278–2324.

Li, Lingyun, Yawei Sun, Xu Han, and Cong Wang (2018). "Research on Improve Topic Representation over Short Text". In: *2018 IEEE Third International Conference on Data Science in Cyberspace (DSC)*. Guangzhou, China, pp. 848–853.

Lin, Zhouhan, Minwei Feng, Cicero Nogueira dos Santos, Mo Yu, Bing Xiang, Bowen Zhou, and Yoshua Bengio (2017). "A structured self-attentive sentence embedding". In: *arXiv preprint arXiv:1703.03130*.

Liu, Yang, Zhiyuan Liu, Tat-Seng Chua, and Maosong Sun (2015). "Topical word embeddings". In: *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*. Austin, TX, USA, pp. 2418–2424.

Luong, Minh-Thang, Hieu Pham, and Christopher D. Manning (2015). "Effective approaches to attention-based neural machine translation". In: *arXiv preprint arXiv:1508.04025*.

Manning, Christopher D., Prabhakar Raghavan, and Hinrich Schütze (2008). *Introduction to Information Retrieval*. Cambridge University Press.

McClelland, David C. (1980). "Motive dispositions: The merits of operant and respondent measures". In: *Review of personality and social psychology* 1, pp. 10–41.

— (1988). *Human motivation*. Cambridge University Press.

McClelland, David C., Richard Koestner, and Joel Weinberger (1989). "How Do Self-Attributed and Implicit Motives Differ?" In: *Psychological Review* 96.4, pp. 690–702.

Melamud, Oren, Jacob Goldberger, and Ido Dagan (2016). "context2vec: Learning Generic Context Embedding with Bidirectional LSTM". In: *Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning*. Berlin, Germany, pp. 51–61.

Miao, Yishu, Edward Grefenstette, and Phil Blunsom (2017). "Discovering discrete latent topics with neural variational inference". In: *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. Sydney, Australia, pp. 2410–2419.

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

Mikolov, Tomas, Martin Karafiat, Lukas Burget, Jan Cernocky, and Sanjeev Khudanpur (2010). "Recurrent neural network based language model". In: *Eleventh Annual Conference of the International Speech Communication Association*. Makuhari, Japan, pp. 1045–1048.

Mikolov, Tomas, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean (2013b). "Distributed Representations of Words and Phrases and their Compositionality". In: *Advances in Neural Information Processing Systems*. Lake Tahoe, NV, USA, pp. 3111–3119.

Mitchell, Thomas M (1997). *Machine Learning*. 1st ed. New York, NY, USA: McGraw-Hill, Inc.

Moody, Christopher E. (2016). "Mixing Dirichlet Topic Models and Word Embeddings to Make lda2vec". In: *arXiv preprint arXiv:1605.02019*.

Morgan, Christiana D. and Henry A. Murray (1935). "A method for investigating fantasies: The Thematic Apperception Test". In: *Archives of Neurology And Psychiatry* 34, pp. 289–206.

Murray, Henry A. (1943). *Thematic apperception test.* Cambridge, MA, US: Harvard University Press.

Nair, Vinod and Geoffrey E. Hinton (2010). "Rectified Linear Units Improve Restricted Boltzmann Machines". In: *Proceedings of the 27th International Conference on Machine Learning*. Haifa, Israel, pp. 807–814.

Nguyen, Dat Quoc, Richard Billingsley, Lan Du, and Mark Johnson (2015). "Improving topic models with latent feature word representations". In: *Transactions of the Association for Computational Linguistics* 3.1, pp. 299–313.

Nigam, Kamal, Andrew K. McCallum, Sebastian Thrun, and Tom Mitchell (2000). "Text Classification from Labeled and Unlabeled Documents using EM". In: *Machine Learning* 39.2-3, pp. 103–134.

Niu, Liqiang, Xinyu Dai, Jianbing Zhang, and Jiajun Chen (2016). "Topic2Vec: Learning distributed representations of topics". In: *Proceedings of 2015 International Conference on Asian Language Processing*. Suzhou, China, pp. 193–196.

Pennington, Jeffrey, Richard Socher, and Christopher D. Manning (2014). "Glove: Global Vectors for Word Representation". In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Doha, Qatar, pp. 1532–1543.

Peters, Matthew E., Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer (2018). "Deep contextualized word representations". In: *arXiv preprint arXiv:1802.05365*.

Phan, Xuan-Hieu, Le-Minh Nguyen, and Susumu Horiguchi (2008). "Learning to classify short and sparse text & web with hidden topics from large-scale data collections". In: *Proceedings of the 17th international conference on World Wide Web*. Beijing, China, pp. 91–100.

Pool, Chris and Malvina Nissim (2016). "Distant supervision for emotion detection using Facebook reactions". In: *arXiv preprint arXiv:1611.02988*.

Riedl, Martin and Chris Biemann (2012). "Sweeping through the topic space: bad luck? Roll again!" In: *Proceedings of the Joint Workshop on Unsupervised and Semi-Supervised Learning in NLP*. Avignon, France, pp. 19–27.

Rifkin, Ryan and Aldebaro Klautau (2004). "In defense of one-vs-all classification". In: *Journal of machine learning research* 5.1, pp. 101–141.

Rumelhart, David E., Geoffrey E. Hinton, and Ronald J. Williams (1986). "Learning representations by back-propagating errors". In: *Nature* 323, pp. 533–536.

Schmidt, Anna and Michael Wiegand (2017). "A Survey on Hate Speech Detection using Natural Language Processing". In: *Proceedings of the Fifth International Workshop on Natural Language Processing for Social Media*. Boston, MA, USA, pp. 1–10.

Schneider, Karl-Michael (2010). "Techniques for Improving the Performance of Naïve Bayes for Text Classification". In: *International Conference on Intelligent Text Processing and Computational Linguistics*. Iasi, Romania, pp. 682–693.

Schölkopf, Bernhard and Alexander J. Smola (2001). *Learning with Kernels*. MIT press.

Schüler, Julia, Veronika Brandstätter, Mirko Wegner, and Nicola Baumann (2015). "Testing the convergent and discriminant validity of three implicit motive measures: PSE, OMT, and MMG". In: *Motivation and Emotion* 39.6, pp. 839–857.

Sebastiani, Fabrizio (2002). "Machine learning in automated text categorization". In: *ACM computing surveys (CSUR)* 34.1, pp. 1–47.

Serrà, Joan, Ilias Leontiadis, Dimitris Spathis, Gianluca Stringhini, Jeremy Blackburn, and Athena Vakali (2017). "Class-based Prediction Errors to Detect Hate

Speech with Out-of-vocabulary Words". In: *Proceedings of the First Workshop on Abusive Language Online*. Vancouver, Canada, pp. 36–40.

Song, Ge, Yunming Ye, Xiaolin Du, Xiaohui Huang, and Shifu Bie (2014). "Short text classification: A survey". In: *Journal of multimedia* 9.5, pp. 635–643.

Strapparava, Carlo and Rada Mihalcea (2008). "Learning to identify emotions in text". In: *Proceedings of the 2008 ACM symposium on Applied computing*. Fortaleza, Brazil, pp. 1556–1560.

Subramaniam, L. Venkata, Shourya Roy, Tanveer A. Faruquie, and Sumit Negi (2009). "A survey of types of text noise and techniques to handle noisy text". In: *Proceedings of The Third Workshop on Analytics for Noisy Unstructured Text Data*. Barcelona, Spain, pp. 115–122.

Taylor, Wilson L. (1953). ""Cloze procedure": A new tool for measuring readability". In: *Journalism and Mass Communication Quarterly* 30.4, pp. 415–433.

Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin (2017). "Attention Is All You Need". In: *31st Conference on Neural Information Processing Systems*. Long Beach, CA, USA, pp. 5998–6008.

Wallach, Hanna M. (2006). "Topic Modeling : Beyond Bag-of-Words". In: *Proceedings of the 23rd international conference on Machine learning*. Pittsburgh, PA, USA, pp. 977–984.

Wang, Bing-kun, Yong-feng Huang, Wan-xia Yang, and Xing Li (2012). "Short text classification based on strong feature thesaurus". In: *Journal of Zhejiang University SCIENCE C* 13.9, pp. 649–659.

Wang, Sida and Christopher D. Manning (2012). "Baselines and Bigrams: Simple, good sentiment and topic classification". In: *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers*. Vol. 2. Jeju Island, Korea, pp. 90–94.

Warner, William and Julia Hirschberg (2012). "Detecting Hate Speech on the World Wide Web". In: *Proceedings of the 2012 Workshop on Language in Social Media*. Montreal, Canada, pp. 19–26.

Wegner, Mirko and Thomas Teubel (2014). "The implicit achievement motive predicts match performances and the explicit motive predicts choices for target

distances in team sports". In: *International Journal of Sport Psychology* 45.6, pp. 621–638.

Weston, Jason, Sumit Chopra, and Antoine Bordes (2014). "Memory Networks". In: *arXiv preprint arXiv:1410.3916*.

Wolf, Markus, Andrea B. Horn, Matthias R. Mehl, Severin Haug, James W. Pennebaker, and Hans Kordy (2008). "Computergestützte quantitative Textanalyse. Äquivalenz und Robustheit der deutschen Version des Linguistic Inquiry and Word Count". In: *Diagnostica* 54.2, pp. 85–98.

Wu, Yonghui, Mike Schuster, Zhifeng Chen, Quoc Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Lukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean (2016). "Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation". In: *arXiv preprint arXiv:1609.08144*.

Yin, Jianhua and Jianyong Wang (2014). "A Dirichlet Multinomial Mixture Model-based Approach for Short Text Clustering". In: *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. New York, NY, USA, pp. 233–242.

Yin, Wenpeng, Katharina Kann, Mo Yu, and Hinrich Schütze (2017). "Comparative Study of CNN and RNN for Natural Language Processing". In: *arXiv preprint arXiv:1702.01923*.

Young, Tom, Devamanyu Hazarika, Soujanya Poria, and Erik Cambria (2017). "Recent Trends in Deep Learning Based Natural Language Processing". In: *arXiv preprint arXiv:1708.02709*.

Zelikovitz, Sarah and Haym Hirsh (2000). "Improving short text classification using unlabeled background knowledge to assess document similarity". In: *Proceedings of the Seventeenth International Conference on Machine Learning*. Stanford, CA, USA, pp. 1183–1190.

Zeng, Jichuan, Jing Li, Yan Song, Cuiyun Gao, Michael R. Lyu, and Irwin King (2018). "Topic Memory Networks for Short Text Classification". In: *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Brussels, Belgium, pp. 3120–3131.

Zhang, Renxian, Wenjie Li, Dehong Gao, and You Ouyang (2013). "Automatic Twitter Topic Summarization With Speech Acts". In: *IEEE Transactions on Audio, Speech and Language Processing* 21.3, pp. 649–658.

Zhang, Xiang, Junbo Zhao, and Yann LeCun (2015). "Character-level Convolutional Networks for Text Classification". In: *Advances in Neural Information Processing Systems*. Montreal, Canada, pp. 649–657.

Zhou, Zhi-Hua (2012). *Ensemble Methods: Foundations and Algorithms*. Chapman and Hall/CRC.

Zipf, George Kingsley (1929). "Relative Frequency as a Determinant of Phonetic Change". In: *Harvard Studies in Classical Philology* 40, pp. 1–95.

A Dataset Examples

761229275366460367810 11;Sie müssen Arbeit fertig stellen, Leistung erbringen. Handwerklich tätig sein. Je nachdem wie das von ihnen geforderte Pensum ist. Sie stehen entweder unter Druck oder können sich zeit lassen.ihnen bleibt keine andere wahl. das ist ihr leben, ihre arbeit. ;L;4

761229275366460367810 11;Sie entschuldigt sich.Unterdrückt, klein. Sie hat den Stift eines anderen kaputt gemacht).... Der stift kann ersetzt werden, alles halb so schlimm.;M;5

11311412943387617M10;Die Person gegenüber soll überzeugt werden.überlegen, unverständlich.Die Person fühlt sich in seiner Intelligenz beleidigt, da die Selbstverständlichkeit, für die er argumentiert nicht verstanden wird.;M;4

1131141294291431 2M7;Zuwendung.einsam.weil sie allein ist.;A;5

13491831192726807846K15;sie sin eine Gemeinschaft.unsicher.sie wissen nicht, was auf sie zukommt.;M;5

13491831192726807846K15;sie maßregelt.gut.sie glaubt, sie hat ein Recht dazu.;M;4

6241478198361714M7;unterhalten.freudig unterhalten.da sie eine gute Konversation führen.;A;2

624147819836171 4M7;versucht zu ignorieren.ausgeschlossen.da sie nicht mitmacht und über sie gesprochen wird.;A;5

6081477664560133M10;Wichtig ist, dass sie nachdenken kann und Ruhe hat.Sie fühlt sich ganz in Ihrer Kraft, was Ihre Intellektuellen Fähigkeiten betrifft. Weil sie dasitzt und den Blick nach unten gerichtet sinniert und nachdenkt. ;F;1

1551757410308674421K15;dass die Weisse wieder zu Atem kommt.behütend für die andere.weil sie mehr Erfahrung hat als die Weisse.harmonisch.;M;1

B Paragraph Vector Confusion matrices

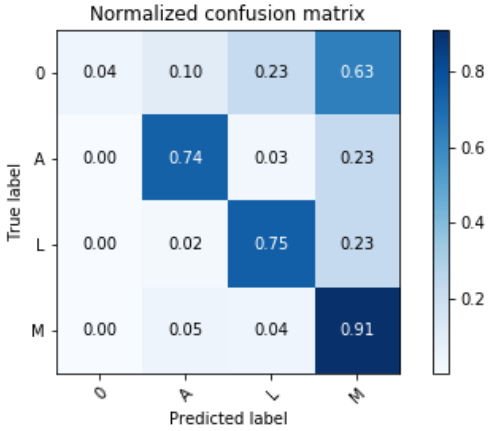


Figure B1: Confusion matrix of PV-DBOW model. Similar to most other classes, the right column reveals, that the model classifies many documents to the M class and therefore shows poor results for the other classes, especially 0

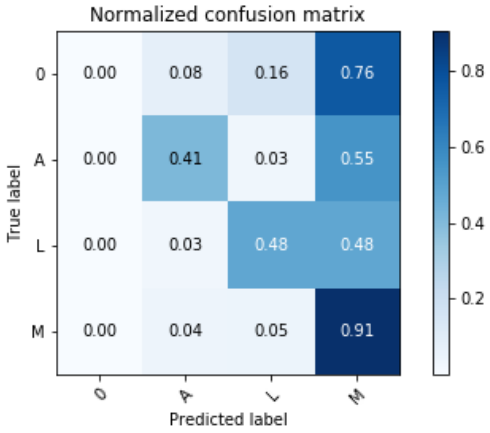


Figure B2: Confusion matrix of the PV-DM model. The poor perfomance can be explained by the overfitting to the M class

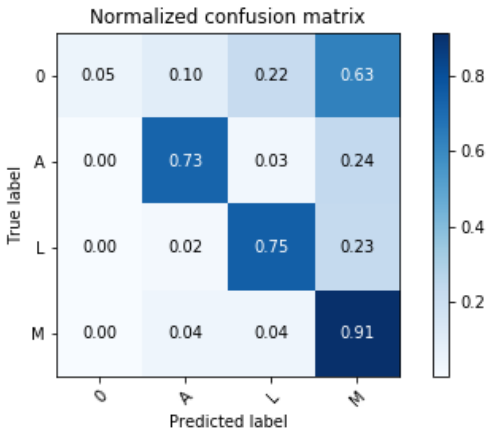


Figure B3: Confusion matrix of the combined Paragraph Vector model

C Topic Model from LDA

Table C1: Top words per class from a topic model trained with best hyperparameter settings

	WORD 1	WORD 2	WORD 3	WORD 4	WORD 5	WORD 6	WORD 7	WORD 8	WORD 9	WORD 10
0	durch	des	beide	einander	vertrauen	arm	schulter	gutes	vertraut	verantwortung
1	schlecht	klein	unterlegen	schuldig	schämt	kopf	beschämt	sich	dafür	entschuldigt
2	der	und	sie	die	ist	chef	vom	eigene	froh	eigenen
3	machen	sie	ganz	schon	noch	fertig	gelangweilt	und	weil	ist
4	diese	sie	geben	geborgen	jemanden	und	geborgenheit	wohl	umarmt	braucht
5	sie	tun	nichts	soll	die	hören	zu	und	aufgeregt	weil
6	stolz	glücklich	zufrieden	sie	und	weil	lob	hat	erhält	gelobt
7	zur	schaut	arme	sie	sieht	die	verschränkt	und	überlegt	seite
8	hat	einen	gemacht	fehler	sie	weil	streit	sich	und	begangen
9	kann	aufgabe	helfen	lösen	problem	hilfe	bei	hilft	unterstützung	rätsel
10	nicht	verstanden	dabei	sie	die	zu	bleibt	und	der	bringen
11	als	wollen	unter	team	ruhe	druck	spiel	müssen	arbeiten	schnell
12	anderen	der	sie	die	verantwortlich	und	person	position	ist	dominant
13	und	mehr	ihrer	sie	ihre	die	mutter	kritik	sich	erfüllt
14	ein	sie	und	der	ist	weil	freund	sich	bisschen	die
15	sie	ihrem	zu	lassen	und	die	weil	sich	überfordert	muss
16	sie	hört	weil	zu	tut	und	sich	begeistert	ist	die
17	sich	und	selbst	zeit	sie	gerne	würde	zu	bewusst	die
18	sie	und	die	ist	weil	sich	selbstsicher	zu	person	der
19	berg	den	klettert	kommt	oben	halt	kommen	angestrengt	gipfel	hoch
20	auf	sie	und	die	gehalten	sich	leute	weil	liegt	augenhöhe
21	sie	und	zu	die	ist	sich	weil	verständnis	der	missverstanden
22	nähe	und	zurück	sucht	sich	sie	die	war	der	eher
23	sie	man	wenn	nicht	zu	weil	und	ist	sauer	sich
24	über	geht	reden	weg	die	sie	und	damit	beschäftigt	darüber
25	für	die	ist	sie	dieser	und	der	neugierig	zu	sicherheit
26	alle	sie	erfolg	und	den	im	weil	mittelpunkt	feiern	hat
27	wird	von	sie	und	sich	akzeptiert	schutz	weil	die	respektiert
28	sich	der	und	sie	die	anderen	weil	zu	körpersprache	person
29	sie	die	weil	und	sich	der	ist	zu	person	nicht
30	wie	sie	zuhören	aufmerksamkeit	zeigen	und	weil	der	erhalten	die
31	fühlt	sich	sie	nicht	dazu	die	und	kleine	mut	weil
32	mit	die	sie	sitzt	und	person	ist	armen	der	weil
33	sie	und	besser	menschen	weil	die	ist	zu	sich	sein
34	gut	sie	sehr	weil	und	sich	die	schreiben	beraten	weint
35	sie	gerade	wurde	weil	sich	zu	und	verletzt	der	nicht
36	möchte	sein	sie	zu	und	sich	genervt	weil	die	muss
37	gegenüber	nimmt	halten	ohne	thema	vortrag	vermitteln	rede	sie	viel
38	ziel	konzentriert	erreichen	an	erreicht	angespannt	motiviert	schaffen	bzw	erfolgreich
39	eine	lösung	finden	findet	gedanken	nachdenken	sie	muss	die	zu
40	aber	auch	sie	jetzt	ist	nicht	besorgt	die	mag	und
41	person	andere	die	hand	jedoch	linke	rechte	angegriffen	trösten	linken
42	nicht	keine	weiß	weiss	ich	nur	ob	dann	unsicher	ahnung
43	steht	dem	die	hinter	sie	rücken	und	einfach	weil	sich
44	macht	da	sie	sich	und	die	weil	nicht	der	ist
45	werden	zu	sie	aufmerksam	ernst	und	weil	gehört	genommen	nicht
46	körperhaltung	gelassen	sie	mächtig	wirkt	wegen	eingeschüchtert	und	überlegenheit	der
47	das	kind	wieder	dem	vater	sohn	sagt	der	seinem	beziehung
48	sicher	sie	haltung	sache	die	der	und	sich	klar	weil
49	zusammen	arbeit	gemeinsam	sind	arbeiten	bauen	immer	jeder	zwei	erledigen
50	was	sie	wissen	sagen	zu	weil	der	die	anderen	ratschläge
51	sie	und	den	zu	kontakt	freude	die	der	weil	sich
52	personen	anderen	beiden	allein	ab	ausgeschlossen	sich	wendet	einsam	den
53	und	lässt	die	erzählt	sie	weil	ist	eines	sich	den
54	zu	versucht	erklärt	die	erklären	sie	verstehen	versteht	überzeugen	lernen
55	zu	sie	scheint	und	die	sich	der	bleiben	ist	weil
56	es	wichtig	ist	dass	sie	so	anliegen	und	dennoch	für
57	gespräch	einem	am	zeigt	tisch	dem	wenig	andern	kollegen	setzt
58	der	die	person	sie	anderen	und	verhalten	anderer	unzufrieden	sich
59	nach	traurig	oder	alleine	denkt	vielleicht	nachdenklich	sie	leben	passiert
60	hat	bekommt	ängstlich	falsch	angst	enttäuscht	bekommen	ärger	schlechte	sie
61	sie	weil	hält	und	die	ist	besprechen	der	gleichen	sich
62	haben	sind	entspannt	miteinander	unterhalten	gehen	spaß	ihnen	sprechen	lachen
63	gruppe	einer	vor	die	stehen	teil	sie	kennt	und	unsicher
64	sie	sich	weil	beobachtet	könnte	die	und	ist	nahe	denn
65	etwas	sie	interessiert	spricht	der	die	von	überzeugt	anderen	und
66	um	sie	die	jemand	und	zu	ruhig	keinen	weil	sich
67	sie	und	sich	freuen	sehen	bis	die	voller	weil	lange
68	ihr	sie	zuhört	neue	und	genau	die	erwartet	zu	weil
69	in	situation	und	sie	sich	ist	der	die	moment	partner
70	leistung	sie	anerkennung	gute	und	die	jubeln	freudig	weil	der
71	aus	können	sie	fühlen	die	dritte	sich	zu	der	austausch
72	stark	den	stein	geschafft	schafft	einen	kraft	hebt	stärke	sie
73	überlegen	im	meinung	recht	ihre	wütend	schimpft	ihren	hauptperson	doch
74	zum	und	beim	freut	mal	leicht	zu	herausforderung	spielen	die
75	sie	zu	und	weil	der	die	weiter	ist	sich	spielt
76	will	sie	gibt	rat	weil	guten	wichtiges	nervös	die	und
77	sie	weil	ist	die	der	und	sich	anderen	ärgerlich	nicht
78	er	ihm	ihn	seine	schüler	lehrer	der	verärgert	aufgaben	seinen
79	dass	sie	ist	die	alles	nicht	richtig	dies	und	erledigt

Eidesstattliche Erklärung

Hiermit versichere ich, Fabian Meyer, an Eides statt, dass ich die vorliegende Arbeit im Bachelorstudiengang Mensch-Computer-Interakion selbstständig verfasst und keine anderen als die angegebenen Hilfsmittel - insbesondere keine im Quellenverzeichnis nicht benannten Internet-Quellen - benutzt habe. Alle Stellen, die wörtlich oder sinngemäß aus Veröffentlichungen entnommen wurden, sind als solche kenntlich gemacht. Ich versichere weiterhin, dass ich die Arbeit vorher nicht in einem anderen Prüfungsverfahren eingereicht habe und die eingereichte schriftliche Fassung der auf dem elektronischen Speichermedium entspricht.

_____ _____

Ort, Datum Unterschrift

Erklärung zur Veröffentlichung

Ich stimme der Einstellung der Arbeit in die Bibliothek des Fachbereichs Informatik zu.

Ort, Datum	Unterschrift