Masterarbeit

Conversation-aware Classification of News Comments

Klassifikation von Zeitungs-User-Kommentaren unter Berücksichtigung der Konversation

Johannes Filter
hi@jfilter.de

Eingereicht am 15. April 2019

Universität Potsdam
Digital Engineering Fakultät
Fachgebiet Informationssysteme
Betreuung: Dr. Ralf Krestel

To all detained journalists.

Acknowledgements

I thank Professor Felix Naumann and his whole chair, especially Dr. Ralf Krestel, Julian Risch and Samuele Garda for their advise.

I thank my brother Julian Filter for commenting on drafts of this master's thesis.

I thank my family, the *Studienstiftung* and the European Commission in the Erasmus program for supporting me financially during my time as a student.

I thank the whole Open-source Software community.

I thank all the people who fight for high-quality public education.

Abstract

Online newspapers close their comment section because they cannot cope with the sheer amount of user-generated content. Natural-language processing allows to automatically classify news comments in order to efficiently support moderators. Identifying hate speech is only a special case of comment classification and in this master's thesis we focus on classifying along any classification criteria, e.g., sentiment, off-topic, controversial. In contrast to prior work, we consider the conversational context to be essential for understanding a comment's true meaning. We introduce a preprocessing technique to prepend previous comments to training samples in order to apply state-of-the-art language-model-based text classification technique ULMFIT. We conducted experiments on nine categories of the research dataset Yahoo News Annotated Comment Corpus. With conversation-aware models, we increased the $F_{1\ \text{micro}}$ and $F_{1\ \text{macro}}$ scores on average by 1.53% and 3.08%, respectively. However, the differences to conversation-agnostic models vary among the categories. We achieved the biggest improvements when identifying whether a comment is off-topic or if it agrees or disagrees with other comments.

Zusammenfassung

Online-Zeitungen schließen ihren Kommentarbereich, weil sie mit der schieren Menge an nutzergenerierten Inhalten nicht fertig werden. Die maschinelle Sprachverarbeitung ermöglicht es, Zeitungs-User-Kommentare automatisch zu klassifizieren, um Moderatoren effizient zu unterstützen. Die Identifizierung von Hasskommentaren ist nur ein Sonderfall der Kommentarklassifizierung und in dieser Masterarbeit konzentrieren wir uns auf die Klassifizierung nach beliebigen Klassifizierungskriterien, wie z.B. Sentiment, Off-Topic, Kontroversivität. Im Gegensatz zu früheren Arbeiten betrachten wir den Konversationskontext als wesentlich für das Verständnis der wahren Bedeutung eines Kommentars. Wir stellen eine Vorverarbeitungstechnik vor, um vorherige Kommentare an Lernbeispiele anzufügen, um die neueste sprachmodellbasierte Textklassifikationstechnik ULMFIT anzuwenden. Wir haben Experimente an neun Kategorien des Forschungsdatensatzes *Yahoo News Annotated Comment Corpus* durchgeführt. Bei konversationsbewussten Modellen haben wir die Werte für $F_{1\text{ micro}}$ und $F_{1\text{ macro}}$ im Durchschnitt um $1,53\%$ bzw. $3,08\%$ erhöht. Die Unterschiede zu konversationsagnostischen Modellen sind jedoch je nach Kategorie unterschiedlich. Wir haben die größten Verbesserungen erzielt, wenn es darum ging, festzustellen, ob ein Kommentar nicht zum Thema passt oder ob er mit anderen Kommentaren übereinstimmt oder nicht.

Contents

1. Introduction

For hundreds of years, newspapers inform the public and hold the powerful accountable. Two things contributed to its establishment. First, the invention of the printing press by Johannes Gutenberg was a technological catalyst. The flow of information was immensely increased. Before, writings such as books had to copied by writing each single page by hand. Second, the enlightenment spurred the critical reflection of power structures. But for the longest time in history, the communication in a newspaper was unidirectional. The journalists wrote their articles and the readers had to consume them. Nevertheless, the writing and their political implications were discussed in a smaller circles. For instance in the 19th century, the bourgeoisie organized in reading societies as portrayed by the painter Johann Peter Hasenclever in Figure 1.1.

Figure 1.1.: Johann Peter Hasenclever: *Das Lesekabinett* (The Reading Society), 1843.[1]

[1] `https://commons.wikimedia.org/wiki/File:Lesegesellschaft,_um_1843.png`

The audience is limited when discussing an issue in person. A way for a non-journalist to join the debate in a newspaper are *letters to the editor*. Everybody can send their opinions in a letter to the editors of newspapers. Editors can then decide to print a selection. Until today, this is a common tool for newspapers to give voice to the ordinary people. However, it is rather unlikely that the letters are printed since space in a printed newspaper is sparse. The introduction of the Web revolutionized media. News are now delivered online as well and the space restrictions of the Internet are more relaxed. Since the era of so called *Web 2.0*, users generate content themselves. These developments established the *comment section* as a special place on blogs, social media platforms or newspapers where users can leave their thoughts.

Figure 1.2 displays two typical comments on an opinion piece of the New York Times.

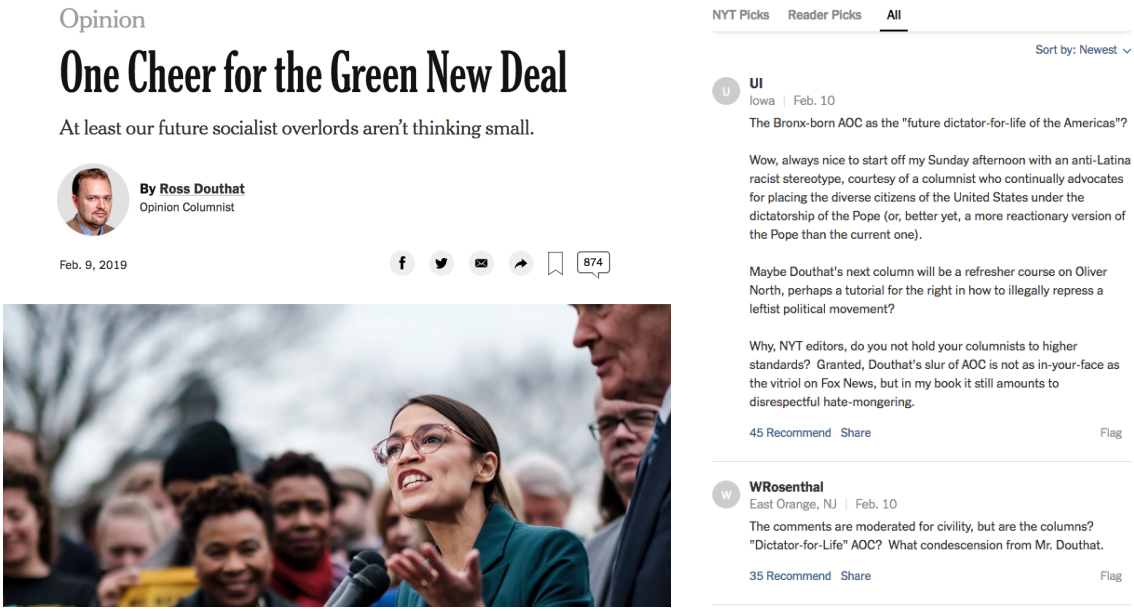


Figure 1.2.: An option article of the New York Times received 878 comments before the comment section was closed.[2]

In the beginning, comments were hailed as a democratization of the debate. In contrast to the previous unidirectional communication, the communication is now multidirectional. Readers can discuss an article with one another. They can also give new perspectives to an issue by contributing personal stories or adding expert knowledge on a specific subject. Furthermore, they hold journalists accountable as Glenn Greenwald[3] states:

> "Journalists often tout their responsibility to hold the powerful accountable. Comments are a way to hold journalists themselves accountable."

[2] https://www.nytimes.com/2019/02/09/opinion/alexandria-ocasio-cortez-green-new-deal.html
[3] https://theintercept.com/2017/12/18/comments-coral-project/

In addition, the introduction of online news comments was an emancipatory act of depriving journalists of their role as gatekeepers. For the first time, a debate was open for everybody with an Internet connection – so almost everybody. This sets the idea of discourse ethics as proposed by Jürgen Habermas into practice. The essence of this thought is the belief, that a collective by exchanging arguments can come to superior conclusions than a sole person. This is a stark contrast to the prior philosophical idea of Immanuel Kant's categorical imperative which focuses on individuals' convictions. The concept of Habermas is vague and implementation-agnostic but there are certain rules to follow. One rule, i.e., declares that arguments must not be repeated. And he promises if all participants obey the rules, the community will eventually reach a common conclusion which then manifests the morale. So in the best case, readers discuss an issue under an article and in the end, all participants share the same opinion. Nevertheless, this in an idealistic scenario and Habermas did not have news comments in mind when he developed the idea in the beginning of the 1970s. In the following, we examine the current problems of news comments and show that repeated arguments may not the biggest one.

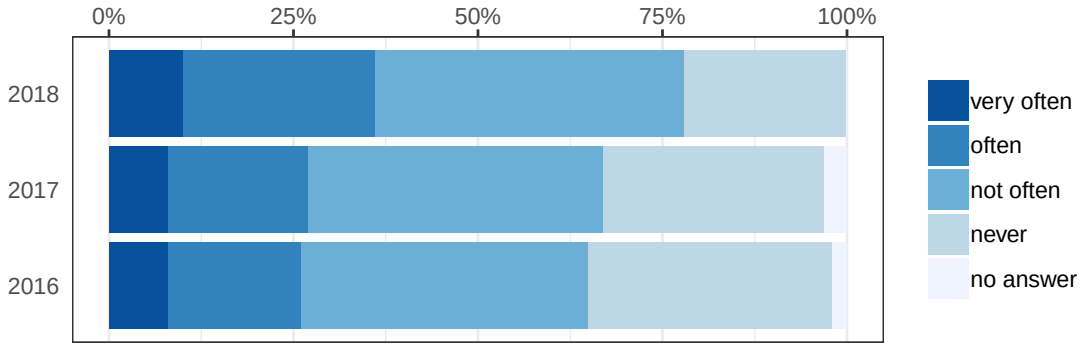
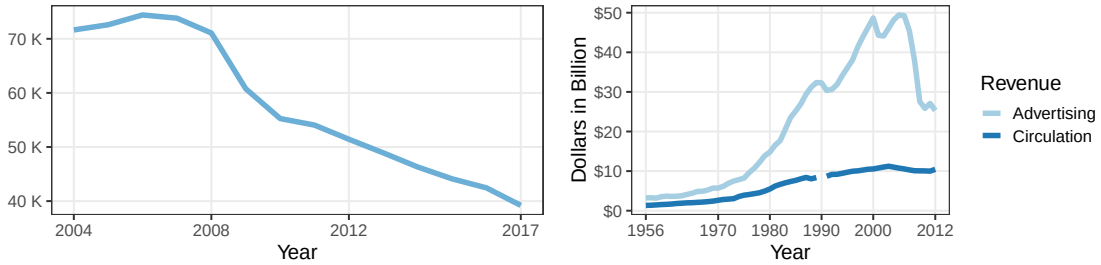


Figure 1.3.: "Have you witnessed hate speech or hast posts on the Internet?". This question is part of a survey among 1000 Germans by the *Landesanstalt Medien NRW*.[4]

Online news outlets are drowning in the vast quantity of user comments. While there are certainly high-quality comments that follow the high hopes of emancipation, there are also comments which are rather unwelcome. They range from out-of-topic discussions to the use of offensive language. A survey by the *Landesanstalt Medien NRW* among German Internet users is shown in Figure 1.3. The survey reveals that users increasingly witness hate on the Web. To remove problematic comments, the comment section needs to be moderated. But the moderation process involves specially trained humans who decide about the existence of comments. Consequently, newspapers abolish their comment section altogether. The biggest German daily newspaper, *Sueddeutsche Zeitung*, closed their comment section in 2014 as one of the earliest newspaper[5].

[4] `https://www.medienanstalt-nrw.de/fileadmin/user_upload/lfm-nrw/Foerderung/Forschung/Dateien_Forschung/forsaHate_Speech_2018_Ergebnisbericht_LFM_NRW.pdf`

[5] `https://www.freitag.de/autoren/jan-jasper-kosok/die-sz-schliesst-ihre-kommentarfunktion`



(a) Employees in the US news sector.[7]

(b) Revenue in the US news sector.[8]

Figure 1.4.: The decline of the US news sector at the example of number of employees and revenue.

In 2016, a survey[6] among German news outlets showed that two in five restricted commenting on their website. Although moderators could help, they are expensive and the news industry already has economic problems and is on a decline. In Figure 1.4, two graphs illustrate the problem on the example of the US news sector. The number of employees was almost cut by half from 2004 to 2017. In mid 2000s, the advertising revenue decreased rapidly. One reason are ad-blocking tools for users to hide advertisement to increase the overall user experience. Another reason are new, strong competitors in the online advertising market. The Internet enables new and more creative forms of advertisement with, e.g., Google's AdWords or influencers on Instagram. The news sector is in a ambivalent state. While recent technological development allows novel ways of storytelling with, i.e., interactive graphics, decreasing revenue streams, that go along with it, remain an open problem. Solving it is out of scope of this master's thesis but Natural-Language Processing (NLP) using machine learning techniques promises to improve efficiency in managing news comments. As a consequence, less manual moderation is required so the comment sections remain open which fosters our democracy.

There already exist works to automatically detect hate speech or abusive language in user-generated content [10, 47, 57, 64]. In this work, we focus on classifying comments in a more general way. Comments are assigned to different categories. For instance, what the sentiment of a comment is or whether it is off-topic. It can be any criteria for classification. This classification can aide comment moderators and also creates new opportunities for other features. For instance, comments are currently mainly ranked chronologically or based on their up- and down-votes. The classification of comments allows to rank them by specific criteria. A feature which is enabled by classification are comment filters. One could choose to hide a specific kind of comments. This is already in production on social media platforms such as Twitter or Facebook. They automatically hide low quality or unpleasant comments. With a fine-grained

[6] `https://netzpolitik.org/2016/umfrage-zeitungsredaktionen-schraenken-kommentarfunktionen-2015-weiter-ein/`

[8] Source: Pew Research Center analysis of Bureau of Labor Statistics Occupational Employment Statistics data.

[8] Source: News Media Alliance, formerly Newspaper Association of America (through 2012); Pew Research Center analysis of year-end SEC filings of publicly traded newspaper companies (2013-2017).

comments classification, one could think of even more possibilities. For example, a user is tired of negativity and only wants to read comments with a positive sentiment.

In contrast to prior work, we assume that the article's context, namely, the whole conversation, is essential to make a classification of a single comment. The article acts as a conversation-starter and the comments leading up to the comment are the conversation partners. To illustrate our motivation, we take two humanly-annotated comments from a Canadian newspaper [31]:

> "Time for the elders and chiefs to stand up to the plate and take a leadership role!"

This comment is labeled as constructive (or in other words: high-quality) and

> "Maybe this will motivate the cabbies in TO to clean their filthy cars! That are a disgrace."

is labeled as non-constructive. For humans, it is hard to make a judgment without reading the corresponding article and previous comments first. Moreover, the annotators were required to read the article first before deciding whether an article is constructive. So we should be fair and also give the machine the possibility to obtain the context before classifying. The term 'context' describes different aspects in this domain. One could also think of context by considering a commentator's previous comment history. While this would probably help to improve the performance of a classifier, it is unfair. A classification of text should depend only on a very comment. We illustrate this differentiation using following example. In a fair judicature, a conviction must be based on clear evidence not on the criminal history. The history only matters for determining the sentence. This should apply for the classification of comments as well: only the current text is important, neither previous comments nor misbehavior. Also from a privacy perspective it is important to point out, that creating information about users means profiling them. Storing personal information has further implications since they are under severe protection with the European Union General Data Protection Regulation (GDPR) in force. It is recommended to only store personal information when it is absolutely necessary. For this work, we do not use any user information for classification.

In order to formally define the problem of conversation-aware classification, let C and A be the set of all comments and articles respectively. In addition, we define:

$\text{isArticle} = \{(c, a) | \forall (c, a) \in C \times A : c \text{ is a comment of } a\}$
$\text{isPrevious} = \{(c, o, a) | \forall (c, o, a) \in C \times C \times A : \text{isArticle}(c, a) \wedge \text{isArticle}(o, a) \wedge c_{time} > o_{time}\}$

The training set $T = \{(c_n, a_n, s_n, y_n)\}_{n=1}^{N}$ consists of quadruple-wise data, where $c \in C$, $a \in A$, $s \subseteq \{o | \forall o \text{ isPrevious}(c, o, a)\}$ and $(c, a) \in \text{isArticle}$. Further, $y \in Y$ is the corresponding label for $Y = \{1, \ldots, l\}$ classes. We wish to train a classifier γ that maps a comment, its article,

and its previous comments (in the conversation) to classes: $\gamma : C \times A \times C^* \to Y$.

The field of journalism is tightly coupled with the technological advancement of our society. As mentioned earlier, only through the printing press could humans spread information so fast. This allowed the journalistic profession to establish and flourish. The ongoing digital revolution also affects journalists and there is a long tradition of supporting their work with digital technologies. It started in the late 1960s with computer-assisted reporting and lead to current ideas about automatic reporting. Right now, there is a larger effort by supporting newspapers in managing their user comments. On example is the unprecedented Coral Project[9], a cooperation among Mozilla, the New York Times, and the Washington Post, that interviewed more than 400 experts in 150 newsrooms to develop an IT system to manage comments. Research on news comments or other user-generated content is done extensively in other research fields. For instance, there exists research on news comments in the field of communication studies. Recent work by Ksiazek and Springer [33] gives an overview over current trends and future directions in this area. The increasing hate in the comment section is one object of investigation. Besides, there is a multitude of publications that highlight the importance of a comment's context. Loosen et al. [35] formulated several comment quality indicators after conducting interviews with news professionals as well as developing a software mockup. Among other things, they list 'reference to the article' and 'references to other comments' as an indicator for high-quality comments. Their work is built upon earlier research done by Diakopoulos et al. [14] and Noci et al. [48] who also investigate characteristics of comments. The relation of one comment to other comments or the article itself plays a role when looking at the perception of comments. There is also an abundance of comment guidelines that describe comments that newspaper desire. The New York Time writes in their guidelines[10]: "We are interested in articulate, well-informed remarks that are relevant to the article." Also, the community guidelines by the Guardian requires commentators to "keep it relevant"[11]. So, the relation of comment to the article and previous comments is important to consider. Machine learning methods ought to respect the knowledge generated in non-computer-science fields and try to include them into their own work.

This thesis is structured as follows. In Section 2, we give a detailed overview of related scientific work before providing background information in Section 3. In Section 4, we describe the datasets we conduct experiments on. In Section 5, we present our contributions and evaluate them in Section 6. Finally, we conclude in Section 7 and give an outlook for future work.

[9] https://coralproject.net
[10] https://help.nytimes.com/hc/en-us/articles/115014792387-Comments
[11] https://www.theguardian.com/community-standards

2. Related Work

The related literature can be split into two categories. Section 2.1 is about the classification of news comments. While for some approaches consider the context, i.e., an article's title, it is only one feature among others. The comment is not considered embedded in a whole conversational context. This is different to approaches presented in Section 2.2. There the exploitation of the sequential or tree-structures of online discussions is the principal motivation.

2.1. Classification of News Comments

With the beginning of Web 2.0, there is an abundance of user-generated content freely accessible. Some influential earlier work analyzed comments on Digg[1] [20, 34], or predicted the popularity of online content on YouTube[2] and Digg [69] or Reddit[3] [58]. The analysis of news comments is a subset of the research done on user-generated content so it falls in this line. But we focus on news comments so the remaining section is dedicated to them.

Coming from the area of Human-Computer Interaction (HCI), Park et al. [51] build an IT system to support moderators to identify high-quality comments. They incorporate traditional, feature-based machine learning. Besides other components, they automatically assign a quality score to each comment. They hand-crafted various features based on a comment's text as well as metadata. For instance, they consider a comment's text as well as the commentator's history. The authors tried to measure the distance of a comment's text to the article's text and other comments' text as *relevance* of a comment. This relevance metric originates from work by Nicholas Diakopoulos [15]. He uses a simple tf-idf [61] model to create vector representations of comments and articles to use, i.e., cosine distance to measure similarity amongst comments or between an article and a comment. This metric has serious short comings since it only considers identical words matches. So for instance, the heavy use of synonyms results in a comment to be marked as irrelevant to an article. However, for Park et al. the machine learning part is not the focus of their work. They simply demonstrate what can be built by applying NLP and machine learning to news comments. In this sense, well-functioning machine learning techniques, such as classification, are a necessity for those end-to-end systems. Park et al. conclude in their section that more research in the area is required.

[1] `https://digg.com`
[2] `https://youtube.com`
[3] `https://reddit.com`

A similar work by Häring et al. [22] focusses on the classification of German *meta-comments*. Meta-comments are comments that, i.e., address an article's authors directly. So they are not about the topic of an article but *meta*, so meta-comments. In cooperation with a large German newspaper, they highlight the need for journalists to identify those comments. Journalists only have a limited amount of time so they want to spend it wisely. For the classification, they experiment with various traditional manually engineered features. They also conduct qualitative research about identifying also characteristics of those meta-comments. The author's research is only touching machine learning with NLP since it is published at a HCI conference. They also tried to use neural models but traditional machine learning methods outperformed them. They conjecture that the number of annotated data with only a couple hundreds samples is too few. They do not use any conversation or context-aware methods. Only the news department of an articles is considered. Besides proprietary data, they use the research dataset *One Million Posts* by Schabus et al. [63] consisting of Austrian comments. The authors constructed a dataset of one million unlabeled comments and annotated a view thousand of them. They present several approaches and experiment with feature-based machine learning as well as deep learning. On two out of eight categories a Long short-term memory network (LSTM) [24] achieved the best F_1 score. In their follow-up paper, Schabus and Skowron [62] describe how they resort to a feature-based machine method for usage in a production system. They also experiment with including features related to the headline of category of the article but could not get good results. They focus on the implementation of a news comment classification system. While they report baselines, that point out that there are only baselines. Their F_1 scores range from about 0.15 (positive comments) to about 0.7 (personal stories) so there is still room for improvement. They also point out the difficulties of news comment classification since comments are much more "diverse in terms of topic, style, length, author intention [..]" than movie reviews. Detecting sentiment in movie reviews is a common problem setting for NLP research.

News organization realized that high-quality comments are easily overlooked in the vast amount of comments. Even though the community votes on comments, those comments with the most up-votes are not necessarily the ones with the highest quality. Thus, moderators pick comments themselves. On the New York Times[4] (NYT) website, those comments are called *NYT Picks*. Since these comments went through a selection process, one can think about it as a binary classification problem. This caught the interest of research to investigate the characteristics of picked and non-picked comments. But in this kind of post-hoc analysis, it is hard to understand the reasoning process of the moderations and other external factors. It is also more than likely that different people picked and the selection process varied over the time. Altogether, this annotation process is very opaque. Nevertheless, research was working with it. Nicholas Diakopoulos [13] presents nine criteria for comments that distinguish NYT Picks from non-NYT Picks and three of them can be computed: Brevity, Readability, and Personal Experience. For the other six, Diakopoulos paid crowd-annotators to classify them and found significant differences between picked and not picked ones. One finding is that comments are more likely to get picked shorty after the release of an article. However, this could also be explained with the selection process of moderators. Maybe they were following the debate in

[4] `https://www.nytimes.com`

the comment section more closely in the beginning. The authors' work falls more into the area of communication but even there their finding are to question due to uncertainty around the selection process.

Kolhatkar and Taboada [30] try to automatically identify constructive comments. They define constructive as follow:

> "Constructive comments intend to create a civil dialogue through remarks that are relevant to the article and not intended to merely provoke an emotional response. They are typically targeted to specific points and supported by appropriate evidence."

Since there is a scarcity of labeled annotated comments, they used the NYT picks in a creative way. They are aware of the fact that it is hard to draw conclusion from the non-picked comments. So they only consider the picked comments and define them as constructive. These are the positive samples. As negative, they resort to another dataset about news comments, the Yahoo News Annotated Comment Corpus (YNACC) [46]. The YNACC contains annotations on comment-level as well a thread-level. One category for the classification is constructiveness. So Kolhatkar and Taboada choose all comments from non-constructive threads as the negative samples. They conduct experiments on several model variations. They report results on yet another set of annotated comments that they published as the SFU Opinion and Comments Corpus (SOCC) [31]. However, they were mainly focusing on traditional feature-based machine learning. They were using some simple length features, e.g, comment length and average word length, and achieved an F_1 score of 0.79. They were unable to push the score further with more features. With other more complex features argumentation, text quality, and named entity features, they achieved an F_1 score of 0.84. They also investigated approaches based on deep learning, a bidirectional LSTM, and received a F_1 score of 0.81. The number of training samples with about 30k should be sufficient to outperform traditional machine learning. However, it did not beat the feature-based machine learning approaches on the test set. But their values are only on the SOCC dataset which is used as test set. And those comment stem from another newspaper than the comments the model was trained one. When looking at the validation set, the LSTM outperformances all other approaches with an F_1 score of 0.86 and 0.83 on the second place. The comments in the training and validation set are from the same newspaper. So the results should be interpret carefully. In their follow-up work, Kolhatkar and Taboada [29] predicted the comments in the SOCC with an accuracy of 72.59% by using an LSTM. This must also be taken with caution since the number of samples is relatively low (1k).

The aforementioned YNACC dataset was in the focus of the work of Napoles et al. [45]. They identified constructive threads among 2.4k annotated threads. The major focus of their work is traditional machine learning. They employ traditional feature engineering. Furthermore, they use features of the comment's text as well as features of the user such as commenting history and also user contributed up- and downvotes. They compare several results, also a neural model. The best performing model is a pipeline model that comprises several steps of features

created from a comment's text as well as metadata. They achieve a F_1 score of 0.73 on the test set of the YNACC data and F_1 score of 0.91 on another external dataset of an online discussion forum. Even if the numbers on the external datasets sound impressing, no significance testing was applied. Since the datasets with about 1k samples is, once again, relatively low, the results may be created by chance. And it is not clear on how many different external datasets the model was tested. Maybe it was tested on 10 different datasets, and it performed well on only one datasets and so the results may be cherry-picked. Also it is not clear how difficult the classification of this external dataset it. Maybe a simple baseline could have achieved even better results. So while the paper shows the performance of certain features, the results have to be interpret carefully.

In this section, we briefly introduced research datasets of annotated news comments. A detailed comparison of all available datasets is presented in Section 4.1.

2.2. Conversation-aware Text Classification

In this section, we present text classification approaches that exploit sequential or tree structures of text. This is often the case on social media platforms with their nested comment structures. The conversation is often started by a particle item, such as a news article or a social media post. This starts an online conversation where subsequent comments are part of this conversation.

Cheng et al. [5][5] consider the abstract of the news article as well as surrounding comments to classify comments. It adapts the text matching approach that uses an LSTM with attention mechanism. They describe text matching as a general problem to decide whether two text relate to each other in some way. In this sense, question answering or duplicate detection are a special case of a text matching. In this case, the text matching is done between a comment and surrounding comments as well as a comment and article's abstract and title. The text is represented by Word2vec embeddings [41]. The text matching computations are combined in a final layer. Then then a comment gets classified. However, the paper has one weakness. They interpret all comments with over 10 up-votes as positive and the rest as negative samples in a binary classification problem. This is a gross simplification. For instance, earlier comments are more likely to get more up-votes. So they may only predict comments that appear earlier in the discussions than others. Furthermore, certain topics attract more people and thus receive more votes. So there is no justification for the number of 10 up-votes. In order to use the votes as labels, one has to normalize the up-votes and down-votes. The social media platform Reddit has implemented a system to normalize upvotes since time since posted. Even though the authors achieved an accuracy of 70.75% and a F_1 score of 0.8073, their true contribution is unclear.

[5]This publication was not formally published and is only a pre-print. We still consider it relevant because it is recent and closely related to our (niche) topic.

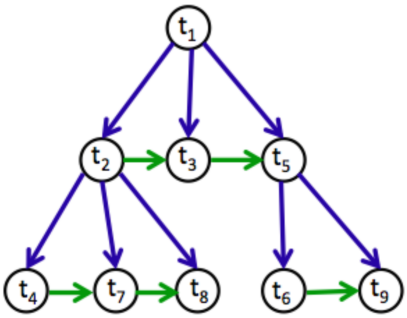


Figure 2.1.: The Graph structure gets introduced to the LSTM with two different previous time steps. The figures is taken from the original paper by Zayats and Ostendorf [73].

Covering the related work on hate speech detection is out of scope for this paper. But there is one particular paper by Gao and Huang [17] that is closely related because they work on context-aware classification. They as well point out that work on comments neglects its context. They developed an architecture of three parallel LSTMs, one for the text, one for the author, and one for the article. Those three LSTMs are combined into a final layer before classifying. They as well used pre-trained Word2vec embeddings for text representation. They constructed their own datasets of annotated tweets that relate to news articles. In their experiments, they claim that their context-aware model outperforms the one without context. Unfortunately, they did not apply their method to a commonly used dataset to put their contribution into perspective.

A new LSTM cell by Zayats and Ostendorf [73] intends to exploit the graph structure of the data. They add an additional previous time step. So there is one previous time step for the hierarchy and one for the time. The two different steps are illustrated in Figure 2.1. To evaluate their contribution, they predict community votings on Reddit comments. With their approach, they outperformed context-agnostic approaches in their work. To represent their text, they used Word2vec embeddings. They did not evaluate their approach on other text classification tasks. Since the votings has some uncertainties as already mentioned earlier, the benefit of this LSTM modification is uncertain. Miura et al. [44] conducted additional experiments with this LSTM cell. But they used Reddits comments that were annotated for their role in the discourse. The dataset is named Coarse Discourse [74] and it will be described in the next paragraph. So it was different setup and the results were below other more traditional machine learning approach of a conditional random field. They hypotheses that predicting user votes is a special kind of problem setting where the text is not a major factor. Other features such as the timing of a comment plays is more important. So this cell is not suited for text classification and thus not for news comments classification.

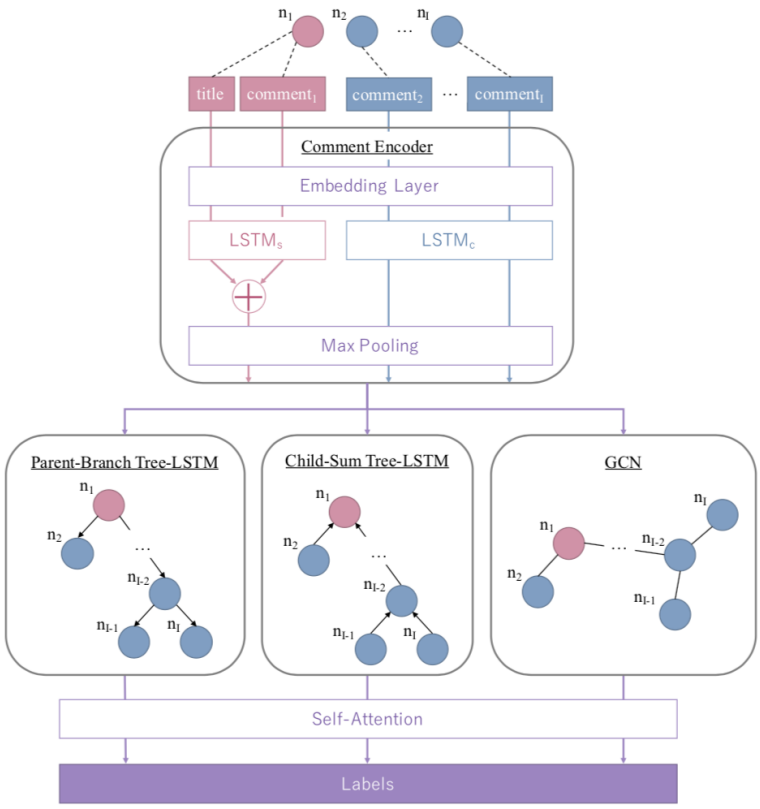


Figure 2.2.: The architecture of integrating tree-structures for text classification. The figures is taken from the original paper by Miura et al. [44].

The aforementioned paper by Miura et al. [44] presented a new graph-based text classification architecture. Overall the architecture is quite complex as visualized in Figure 2.2. First, the comments are encoded into an internal representation. There is the different handling of the first comment (and its title) and all other comments. Second, the comments are given into three context-aware sub-networks before getting concatenating again. To evaluate the performance of their architecture, they conduct experiments on a research dataset of annotated Reddit comments by Zhang et al. [74]. For text representation, they train Word2vec on a millions of Reddit comments. They achieved higher results than previously reported. The Reddit comments are annotated for their role in a discourse. For instance whether a comment is question or answer. This setting is called *discourse classification*.

Another related problem is *stance detection*. In this field, it is about detecting whether a response to a statement is affirmative or opposing. For this, Kochkina et al. [28] introduced Branch-LSTM. They worked with tweets and split the structure into branches. Branch gets created by iterating over all branch nodes of the discussion tree. For each leaf, one goes all the way to the root. This sequence is called branch. The branches are then given as input to the network. For representing the text, they also use Word2vec but average the word embeddings for each comment. A simple averaging is too limiting to capture the nuances of a text [60]. Stops words or commons words such as forms of 'be' or 'have' appear often and distort the whole meaning that should be added with word embeddings. In addition, they use various other hand-crafted features to enrich the tweet representation. There exist various similar ideas by Zubiga et al. [77, 78] which has several works for the conversation. There is also a different problem setting of identifying sarcasm in text. Ghosh et al. [18] identified the role of conversation in this domain. But they only encoded one previous post for each other post. This is quite limiting since a conversation spans among multiple comments. In addition, research on user product reviews is related field. In it, there are as well approaches that take the context of a product review into consideration. For instance, Zheng et al. [75] encoded the product description as well as reviews with an LSTM respectively before combining the two sub-networks into a classifier.

So there exists quite a lot of work on news comments as well as incorporating the conversation for the classification. But all of them a rather simplistic text representation of traditional word embeddings. In the next section, we will give background information on how text representation are derived from language models. They show superior results on various NLP tasks.

3. Background

In this section, we give background information about recent development of NLP research as well as metrics we use for our experiments. Transfer learning with language models achieve state-of-the-art results on text classification. Thus we use them for our work.

3.1. Transfer Learning in Computer Vision and Natural-Language Processing

Transfer learning is a research problem about re-using computation resulting from one problem to another problem. The survey of Pan et al. [50] gives an overview over field as of 2010. With the 'third wave' of artificial neural networks, the field of machine learning changed dramatically and thus transfer learning as well. With AlexNet [32] dramatically outperforming all other approaches in the 2012 ImageNet challenge, it started the revitalization process of neural models. Besides technological progress, one reason for the sudden success were large publicly available annotated research datasets such as ImageNet [11]. More data allowed to train deep networks with more layers networks ('deep learning'). In addition, these datasets gave a new perspective onto transfer learning in computer vision (CV). Nowadays, pre-trained ImageNet models are available in many open source libraries. And it is extremely common to fine-tune only a fraction of the weights as demonstrated by Oquab et al. [49]. Mostly, only the last layers are trained because the early layers only learn very basic features such as edges as shown by Yosinski et al. [72]. Consequently, transfer learning in CV allows to apply large neural models with millions of parameters even on small dataset sizes.

While in CV transfer learning is pre-dominate it was not used as often in NLP. There has been some progress however. Word2vec [43] introduced word embeddings that are trained on large corpora in order to project words into a vector space. The resulting embeddings can be used on other task since they bring 'meaning' to text. The idea of pre-trained word embeddings was especially popularized by pre-trained FastText embeddings available in 157 languages [21]. FastText embeddings [3] are adapting the idea of Word2vec but take sub-word information into consideration. However, the effect of word embeddings for neural networks is limited due to the following reasons: First, since one word can have multiple meanings (polysemy) a single vector is not enough to characterize it. For instance, 'bank' can describe a financial institution or an embankment. The true meaning in a sentence can only be inferred by considering the

context. Second, the message of a sentence can also depend on the context. Irony or satire are hard or impossible to grasp when only looking at individual words. Third, when using embeddings with neural networks, typically the embedding is used only for the first layer. Subsequent, deeper layers do not have access to the 'meaning' of a word that was injected through the word embeddings.

A successor of word embeddings are text representations derived from language models. First we give background information on language models and where they were used originally.

3.2. Language Modeling

The research problem of creating a language model (LM) is, as the name suggests, about abstracting an entire natural language into a model. Instead of formulating rules to describe the construction of a language, the model should capture it by showing a collection of natural language text. There are different ways of defining the problem. One can work on the level of sentences, word n-grams, words, sub-words or characters. We focus on words in this section. We also limit ourselves to modern neural network models since they greatly outperform traditional approaches. For a comprehensive introduction, the interested reader is advised to read Chapter 9 of the book by Goldberg and Hirst [19]. Bengio et al. [2] define LM in their seminal work as follows:

$$P(w_1^T) = \prod_{t=1}^{T} P(w_t | w_1^{t-1})$$

where is the t-th word, and writing sub-sequence $w_i^j = (w_i, w_{i+1}, \ldots, w_{j-1}, w_j)$. This means that the probability of a word depends on all previous words. All previous words can mean a lot of words when the LMs, i.e., is trained on books. For the purpose of simplification, the first approaches worked with a fixed size window. In another groundbreaking work, Mikolov et al. [42] used a recurrent neural network (RNN) to circumvent the fixed window size and Sundermeyer et al. [68] successfully applied LSTMs [24] to language modeling. LSTMs are a powerful variant of RNN because they can capture long-term dependencies in sequences but as well 'forget' previous input sequences. However, since LSTM are strong but contain a lot of parameters, they are known to overfit. Merity et al. introduced AWD-LSTM [39] that use a lot of regularization to overcome overfitting. As of this writing, AWD-LSTM variations still hold various state-of-the-art results on LM competition datasets[1].

LMs are used, e.g., for spell correction, optical character recognition, keyword prediction, or speech recognitions. As a byproduct they also embody a text representation that considers longer sequences of text. This is presented in the next section.

[1] http://nlpprogress.com/english/language_modeling.html

3.3. Transfer Learning with Language Models

The idea of using LMs for text representation is not fundamentally new. As early as 2008, Collobert and Weston [7] demonstrate how they used LM as an auxiliary task for other downstream tasks. But only the ELMO ("Embeddings from Language Models") embeddings by Peters et al. [52] about ten years later popularized the idea. The basic principle works as follows. The first task is to train a LM on a large text corpus. The text does not require special annotations, although the arrangement of words in a text is a type of annotation itself. In such a manner, the model has to learn the nuances of a language over long sequences of text. After training a LM, its capabilities are transformed to other tasks such as text classification. The ELMO embeddings consider only parts of the internal state of the LM for further usage. Other approaches, i.e., ULMFIT [25], adopt the whole trained LM directly and only adds additional layers. ULMFIT will be explained more in detail in the next section. Both use LSTMs for the LM but ELMO operates on character level. There are two approaches based on the Transformer architecture [70]: the OpenAI transformer by Radford et al. [55, 56] and BERT by Devin et al. [12]. BERT improves the OpenAI transformer by allowing the model learn a language forth and backwards at the same time. This requires a different formulation of the language model problem but this is beyond the scope of this brief overview. Abkib et al. [1] use a character-aware LSTM-based LM to transfer-learn to named entity recognition. Peters et al. [53] investigate the different capabilities of LMs. They show, i.e., that language models learn part-of-speech tagging implicitly. However, LM are not the only way to create context-aware text representation [4, 38]. But the current results suggest that they are superior.

3.4. ULMFIT

ULMFIT by Howard and Ruder [25] is the acronym for "Universal Language Model Fine-tuning for Text Classification". As of this writing, the approach holds multiple state-of-the-art results on text classification[2] and sentiment detection[3], i.e., on the IMDb dataset [37]. Since publication, similar models such as the aforementioned BERT achieve state-of-art results on question answering. ULMFIT even with magnitude less parameters, is still undefeated for text classification. In the ULMFIT paper, the authors point out that fine-tuning a LM to a smaller dataset is the crucial part. LMs consist of millions of parameters and when training them on few samples, information may get lost fast ("catastrophic forgetting"). They use three techniques to circumvent it. One technique they use is *discriminative fine-tuning*. It is about assigning different learning rates for each layer. The earlier the layer in the neural network (closer to the input), the smaller the learning rate should be. In general, earlier layers learn fundamentals of the data while latter layers learn high-level features [72]. Another method used is *cyclical learning rates* proposed by Leslie Smith [66]. Smith points out, that networks can be trained in less epochs

[2] `http://nlpprogress.com/text_classification.html`
[3] `http://nlpprogress.com/sentiment_analysis.html`

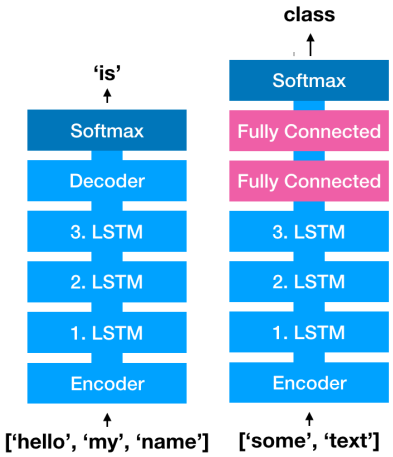


Figure 3.1.: The two different ULMFIT architectures. In Subfigure (a) is the LM with a decoder to predict a news word. In Subfigure (b) is the decoder replaced with two fully connected layers to predict a sample's class.

by changing the learning rate of over time. Howard and Ruder argue that it is important to not fine-tune the LM for many epochs to avoid catastrophic forgetting and overfitting. Finally, they *gradual unfreeze* the layers. Freezing layers means to not update the weights of certain layers. So unfreezing makes them trainable again. This special approach means that they do not unfreeze all layers at once. Instead, they progressively unfreeze layers starting from the final layer. So first, only the last layer is trained. Second, the last layer and second-last layers and so forth until the model is fully unfrozen. The authors state that only by the combination of these techniques, they could achieve high-scoring results. For the LM architecture, they use the already mentioned AWD-LSTM. First, they train a LM on a collection of high-quality English Wikipedia articles *Wikitext103*[4] [40]. After the general training of the LM, it requires fine-tuning to the specific domain. The LM is not powerful enough to hold the knowledge of the complete English language. Since language is different in each domain, they have to fine-tune the LM to the target domain. Then as the final step, a classifier is trained. For this, the LM architecture is augmented with two additional blocks of fully connected layers. The two different model architectures are visualized in Figure 3.1.

The authors claim that even with only a small amount of annotated data, the classification achieves high results. Figure 3.2 demonstrates this at the example on IMDb movie reviews. This strength is particularly useful for newspaper, since resources to annotate data are scarce.

An implementation of ULMFIT is available in the Python deep learning library FastAI[5] which

[4] https://blog.einstein.ai/the-wikitext-long-term-dependency-language-modeling-dataset
[5] https://docs.fast.ai/text.html

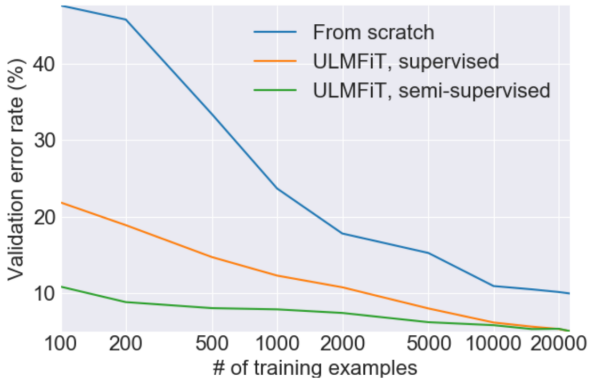


Figure 3.2.: Validation error rates (lower is better) for ULMFIT on IMDb reviews. Semi-supervised means the LM is only fine-tuned on the according number of training samples whereas semi-supervised the LM is fine-tuned on all available samples. The figure was taken from the original paper [25].

itself is built upon PyTorch[6]. It varies slightly from the original publication. The cyclical learning rate was originally not part of the paper but now the authors consider it as an integral part of the method.

3.5. Classification Metrics

There is an abundance of metrics for classification. Sokolova and Lapalme give an overview [67]. We briefly introduce those commonly used for news comment classification.

First, let us revisit precision, recall and F_1 score. With tp as the number of true positives, fp as the number of false positives, tn as the number of true negatives, and fn as the number of false negatives, precision and recall are defined as follows:

$$\text{precision} = \frac{tp}{tp + fp} \qquad \text{recall} = \frac{tp}{tp + fn}$$

[6] `https://pytorch.org`

The F_1 score is the harmonic mean of precision and recall:

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

In the original sense, precision and recall were compute in regard to one class. Hence, true positive and true negatives are calculated separately. But when using the metrics for classification, there is one drawback. The number of true negative samples are neither part of the precision nor recall formula. Thus, these metrics do not lead to meaningful insights. It is a problem when the majority class is treated as positive class. In this case, it is easy to achieve high F_1 scores by simple always predicting the majority class (with a recall of 1 and high precision due being the majority class). To overcome this, typically the scores are computed in regard to each class and then aggregated into a final score. There are three main types of aggregations: *micro*, *macro*, and *weighted*[7]. For *micro*, the samples for true positives, false positives, etc., are counted across all classes and then the final score is computed. For binary and multi-class classification, this is equivalent to the metric accuracy. The accuracy is the share of samples there was assigned to the correct class. For *macro*, the scores are computed for each class separately and the intermediate scores are then averaged to a final score. *Weighted* builds upon the macro-averaged score and weights them in respect to the number of samples per class. Often micro and macro F_1 scores are used in conjunction to asses the quality of model.

Always using two metrics is cumbersome. To have a single metric that evaluates the performance of a model, we use Cohen's Kappa for this thesis. Cohen's Kappa or short Kappa (or κ) was introduced by John Cohen [6] and it was originally meant to measure the agreement among annotators. However, it can be used as classification metric as well [16, 71].

It is defined as follows:

$$\text{Kappa} = \frac{p_o - p_e}{1 - p_e}$$

p_o is the observed agreement (accuracy) and p_e is the expected agreement (accuracy by chance). It takes into account whether classification results could happen randomly. This is especially useful for imbalanced datasets, as it is often the case when working with real-life data such as news comments.

[7] As implemented in the popular Python package scikit-learn.

4. Datasets

In this section, we first give an overview on the available datasets of annotated news comments. Afterwards, we describe chosen datasets in detail and reflect on the ethics of their creation process.

4.1. Datasets of Annotated News Comments

In Table 4.1, we list the characters of four datasets of annotated news comments. As of the writing, they are the only one available for research purposes. Since we are not focusing on hate speech detection, we do not include comments datasets about hate speech here. The interested reader is guided to the datasets section of the survey paper by Schmidt and Wiegand [64].

Table 4.1.: Four corpora of annotated news comments are available for research.

Dataset	Language	Unlabeled	Labeled
SFU Opinion and Comments Corpus (SOCC)	English	10k articles, 663k comments from 303k comment threads	1k comments in responses to 10 articles, labeled for constructiveness and toxicity
Yahoo News Annotated Comments Corpus (YNACC)	English	230k comments from 34k threads under 2.8k articles (not included)	9.2k comments labeled for agreement, audience, persuasiveness, sentiment, tone, off-topic (15 sub-classes)
One Million Posts Corpus (OMPC)	German	12k articles, 1M comments	11k comments with the following labels: sentiment, off-topic, inappropriate, discriminating, feedback, personal studies, argument used
Tencent News Corpus (TNC)	Chinese	200K articles and 4.5M comments	40k comments labeled for quality (from 1 to 5)

Besides the labeled comments, all corpora contain a large number of unlabeled comments. This is helpful for our language-model-based approach. For our experiments, the Yahoo News Annotated Comments Corpus (YNACC) by Napoles et al. [46] and the One Million Posts Corpus (OMPC) by Schabus et al. [62, 63] are especially interesting. The number of annotated comments in the SFU Opinion and Comments Corpus (SOCC) by Kolhatkar et al. [31] is too small.

The Tencent News Corpus by Qin et al. [54] has enough data but only one label for quality. In addition it is in Chinese which makes it harder to work with the text, since we do not read it. YNACC and OMPC have also some overlapping annotation criteria: off-topic and sentiment. This allows us to compare the same method with identical labels on both datasets.

4.2. Yahoo News Annotated Comments Corpus

First, we describe characteristics of the data and how it was annotated. Then, we present reported values and speculate about used metrics. Finally, we explain required data cleaning steps.

4.2.1. Description of the Dataset and Annotation Process

The online web service provider Yahoo operates a news outlet called Yahoo News[1]. They do not produce own news stories but feature the stories of other outlets. They cover a broad range of topic from politics, to sports to lifestyle. Readers can comment under the articles and there are two types of comments. There are *top-level comments* that are directly sequential under the article and *replies* that follow-up on a top-level comment. Unlike other web services such as Reddit, there is no tree-structure. There is only a sequential flow of top-level comments under the article. And each top-level comment can have a sequential flow of replies. The authors used the term *sub-dialogue* to describe a top-level comment including all its replies.

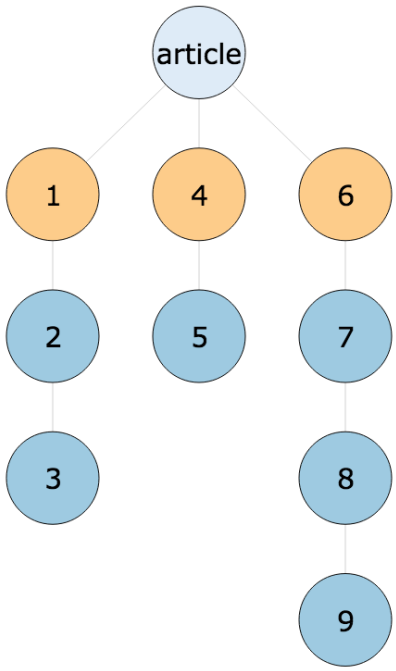


Figure 4.1.: The comments 1, 4 and 6 are top-level comments, all others are replies.

Napoles et al. collected over 230k comments from over 34k sub-dialogues under over 2.8k articles. The comments were written between August 2014 to May 2016. For a subset of 9160 comments, they employed annotators to label them. The dataset contains the comment identifier for training, validation, and test set in a

[1] https://news.yahoo.com

ratio of about 87%, 6% and 6%. It is worth mentioning that there are different groups of annotators for training and validation, and the test set. The authors used crowd workers to annotate the comments from the training and validation set. Different 'Expert annotators' were employed to code the test set. The comments were annotated on the sub-dialogue level as well as the comment level. In this master's thesis, we focus on the comment level and therefore neglect the sub-dialogue annotations. Originally, the authors annotated for over 15 categories but refined to nine categories afterwards. We quote from the annotator notes[2] to describe the different classes:

Persuasive "This is an opinion based on whether you think the user means to be persuasive and how persuasive they seem to be. Generally, the commenter incorporates new information or a personal story, or uses specific language in attempt to convince other users of his or her point. In order for a comment to have true persuasiveness, they must present a well-reasoned argument."

Audience "Choose whether the comment is meant to be 1) a reply to a specific commenter; 2) broadcast message/general audience; [..] Please note that a top-level comment with ALWAYS be a broadcast message with a general audience, by nature of being a top-level comment."

Agreement "Agreement with other commenter: The comment indicates agreement with either another commenter explicitly, meaning, the user is clearly expressing they agree with another person or point of view in the thread. Can occur concurrently with other types of agreement/ disagreement."

Informative "Informative (Constructive, Productive): This comment furthers the discussion by adding new information. Usually an attempt to be persuasive and convince others of the user's argument. It may be passionate, but it is not dismissive. NOT a personal story. Criteria for informativeness: 1. Mention of historical facts or evidence 2. Mention of statistics and other numbers 3. Quote or paraphrase of public discourse made by a popular figure 4. Mention of events, or 'news' 5. Presenting a cogent or logical analysis or argument"

Mean "Mean (Hateful): The comment is intended to be rude, mean, or hateful with no other intent. Be careful to not assign this to comments you personally disagree with. Note that mean, hateful comments/insults can still be on topic with the article or conversation."

Controversial "Controversial (Outspoken): This comment puts forth a strong opinion in a way that others will certainly strongly disagree with."

[2] `https://github.com/cnap/ynacc/blob/master/rater-guidelines.pdf`

Disagreement "Disagreement with other commenter: The comment indicates disagreement with either [sic] another commenter. Can occur concurrently with other types of agreement/ disagreement."

Off-topic "Off topic with article: The conversation/contributions have begun to be irrelevant to the article (Also can be thought of as a digression). Each contribution that is off topic to the article once an off-topic conversation has begun should get an 'off-topic' flag. Once a digression begins, every contribution within the digression is on topic with that conversation but OFF topic with the article."

Sentiment Positive: "A positive sentiment generally expresses feelings of positive emotion, for example 'I love the Yankees! Gunna be a great year.' When users express their opinion on something as being great or good, this is a positive sentiment. Attempts at making jokes or at being funny to 'lift the mood' generally have a positive sentiment, unless they are mean-spirited."

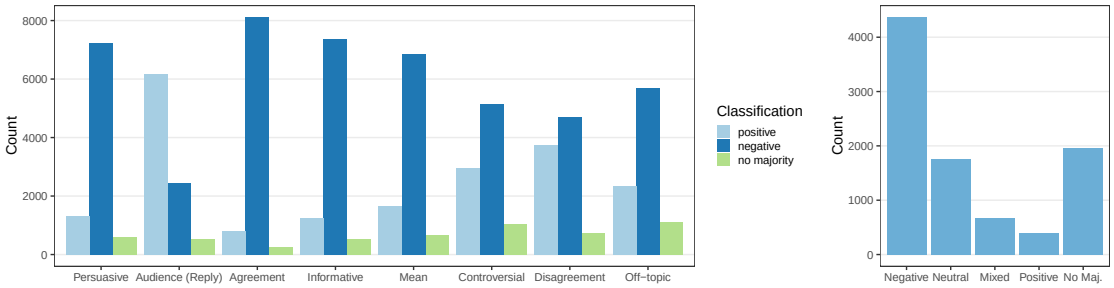
Negative: "A negative sentiment is more common and indicates the commenter is unhappy for some reason. Usually goes in hand with some complaint about the world or the issue. For example, 'Donald Trump sucks. I can't believe he's allowed to be in the public eye.' Additionally, mean or controversial comments are usually coming from a negative emotional place on the part of the commenter. Sarcasm, although funny, is usually used to express discontent or absurdity, is usually negative."

Neutral: "A neutral sentiment has no emotion. Usually when a commenter is just stating fact, like 'I think the season starts in July, actually.' or 'If you water your basil every day, it shouldn't die'."

Mixed: "A mixed sentiment is when a user seems to express both positive and negative emotion about something or several things. A good example is when a commenter expresses that they are in agreement with something one commenter said, but also is offended or takes issue by something else. Longer comments that contain many expressions of different emotions are usually 'mixed'."

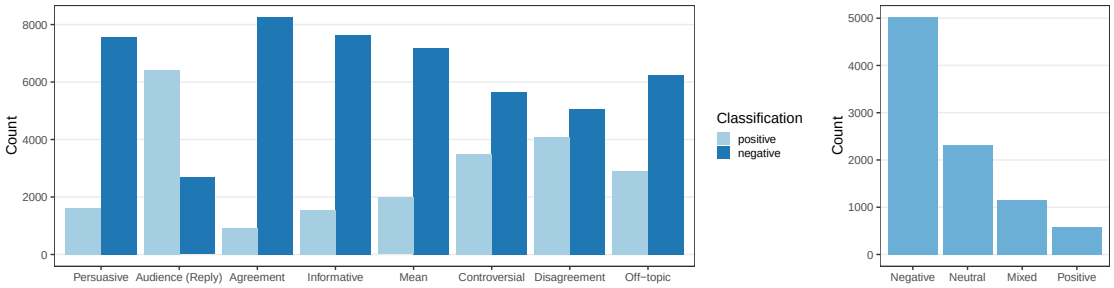
Napoles et al. only publish the raw annotations[3] for each annotator. So for using the data for classification, one has to consolidate the annotations to derive a final assessment. In the authors' follow-up work [45], they choose a majority vote for each sample. If there is no majority for a class, they randomly choose an applicable class, according to the authors' post on GitHub[4]. Consequently, to compare results, it is advised to replicate their approach. In Figure 4.2 is the number is the number of undecided samples visualized. The binary categories Controversial and Off-topic have the largest share of un-decided samples. Over 1000 samples, almost 10% of the data, are randomly assigned to classes. While the share for Sentiment is large as well, it is a

[3] https://webscope.sandbox.yahoo.com/catalog.php?datatype=l&did=83
[4] https://github.com/cnap/ynacc/issues/2



(a) Binary Categories (b) Sentiment

Figure 4.2.: The distributions for each category where the number of samples without majority vote is present.

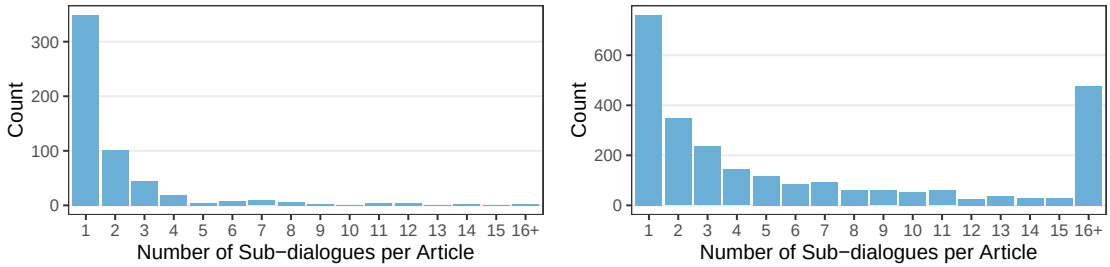


(a) binary categories (b) Sentiment

Figure 4.3.: The final distributions for each category.

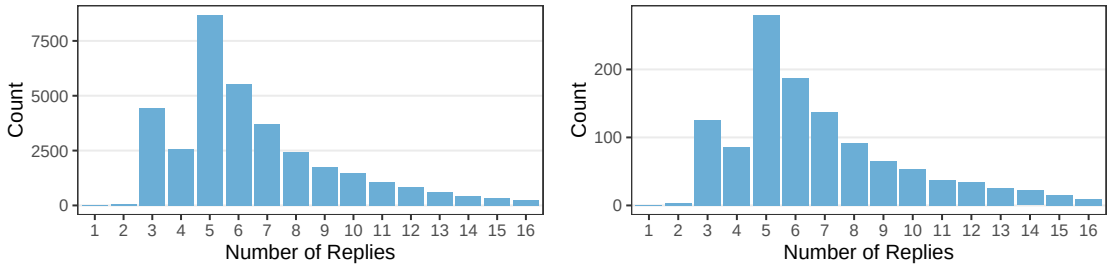
four-class setup so dissension is more likely. The random selection is done for each applicable label. So undecided samples do not get uniformly distributed over the four classes. In Figure 4.3 is the final outcome of this procedure shown. As it is often the case with real-life data, the distributions are mostly imbalanced. Agreement is the most imbalanced while Disagreement is almost balanced.

In addition to the annotated comments, the YNACC contains a large number of unlabeled comments. After cleaning the data, as it will be described in Subsection 4.2.2, about 238k unique comments remain. As we will train a language model which does not require annotations, the unlabeled is also part of this work. In the following, we provide several graphs comparing the characteristics of unlabeled comment set, we denote as YC_{LM} with the portion of labelled training data YC_{CL}. In Figure 4.4 is the number of sub-dialogues per article are shown. There is a stark difference between the two sets of comments with a lot more articles having only one annotated sub-dialogue. In Figure 4.5 is the the frequency of number of replies for each top-level comments. There are only a fews comments in the columns 0–2. This is most likely the reason for some sampling. In Figure 4.4 the frequency of comments per rank is displayed. Here ranks refers to the distance to the parent comment. The top-level comments have a rank



(a) unlabeled comments (YC_{LM}) (b) annotated comments (YC_{CL})

Figure 4.4.: The number of sub-dialogues per article.



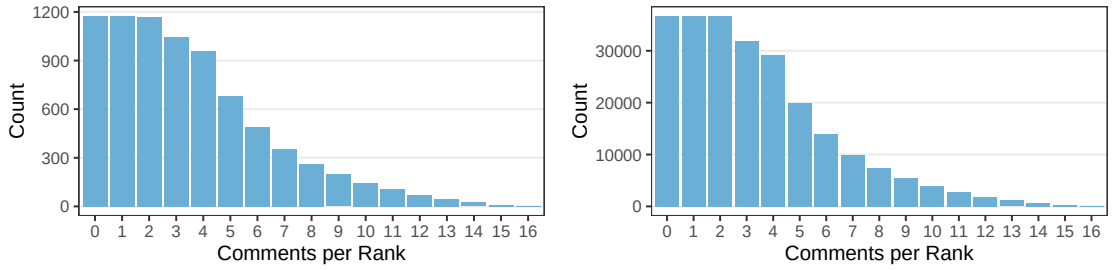
(a) Unlabeled comments (YC_{LM}). (b) Annotated comments (YC_{CL}).

Figure 4.5.: Frequency of number of replies for each top-level comment.

of 0 and no parent comments. While the information is the same in both graphs, Figure 4.4 gives a better feeling of how large the share of comments on the early ranks is. Finally, to get a better sense of YC_{LM}, we list the 30 most frequent tokens in Figure 4.8, excluding stop words and tokens with a length of three or less. In Figure 4.7a is the number of tokens for the comments. This allows to truncate the comments to abolish outliers. And since word based approaches rely on fixed vocabulary, we plot the share of all token covered by choosing only N most frequent tokens in Figure 4.7b.

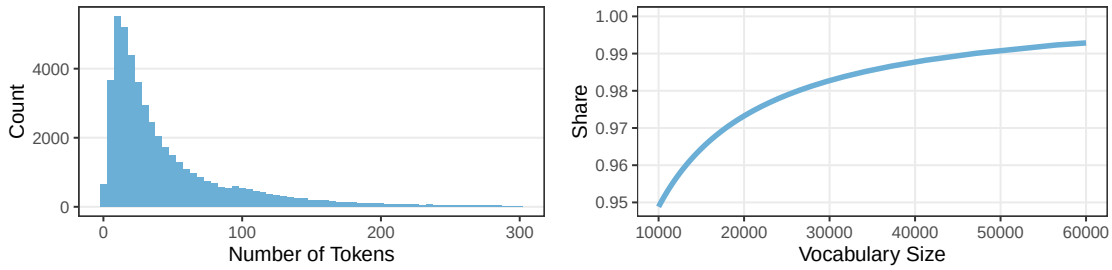
Besides the text, each comment has the number of up-votes, timestamp, parent comment id (if it is a reply), and the article's headline and URL. The article text is not part of the datasets. We were able to crawl about 80 percent of the article texts to not limit ourselves to the headline. Since a lot of articles are offline, we resorted to the Internet Archive's Wayback Machine[5] to fetch them. The data of the articles were extracted using the Python package Newspaper3k[6]. Information about the comment's author is also not part of the datasets. In addition, all mentions of usernames were replaced by the token *@username*.

[5] `https://archive.org/web/`
[6] `https://github.com/codelucas/newspaper`



(a) Unlabeled comments (YC_{LM}). (b) Annotated comments (YC_{CL}).

Figure 4.6.: Frequency of comments per rank.



(a) Number of tokens in a comment, truncated at 300. (b) Share of text covered by fixed vocabulary size.

Figure 4.7.: Text analysis of YC_{LM}.

4.2.2. Data Cleaning

The provided data is 'dirty'. There are duplicated rows as well as corrupted text encodings. In addition, due to the nature of user-generated text, there are other issues, e.g., different forms of quotation marks. So first, the text is cleaned with the Python package *clean-text*[7] in the following way: whitespaces are normalized but (single) newlines are kept, UTF8 encoding issues are fixed, the text is transliterated to ASCII and the casing is kept. All digits are replaced by a '0'. Then, duplicated rows were removed. First, duplicates based on comment ID and text were deleted. Second, rows with duplicated ID but different text were cleaned. This situation occurred because some characters such as quotations marks were stripped from some rows. So in the case of duplicated IDs but different text, we chose the sample with the longest text.

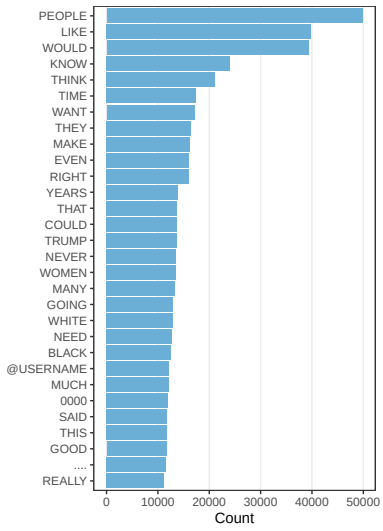


Figure 4.8.: The 30 most frequent tokens in YNACC.

4.2.3. Reported Values

Napoles et al. [45] report in the accompanying publication results for the classification of individual comments. Even though the focus are predictions on thread-level. Unfortunately, it is unclear which kind of metrics they used and they also did not respond to our email. They only write that they use precision, recall and F_1 score. We report their values in Table 4.2 and try to convert their values into our metric of choice: Cohen's Kappa. We gave background information about metrics for classification and shortcomings of the F_1 score in Section 3.5.

It seems like the authors did not average the metrics but only chose one class as the positive one. There are several indications for this. Micro-averaged values are ruled out because precision, recall and F_1 would be identical. For macro-averaging, the values would be extremely strong, i.e., with a recall of 0.99 for the category Audience. They could have used weighted average but this is rather uncommon so they would have mentioned it in the paper. So it is more probable that the authors refrained from averaging. To distinguish this from averaged score, we refer to it as $F_{1\,\text{binary}}$ in Table 4.2. The strong performance of recall for Audience may be due to choose the majority class the positives class[8]. This shows that F_1 scores do not always yield

[7] `https://github.com/jfilter/clean-text`. The package was developed over the course of this master's thesis and borrows code from another Python package textacy: `https://github.com/chartbeat-labs/textacy`.

[8] The F_1 differs depending on what class is the positive and what is the negative class.

Table 4.2.: Reported Results on YNACC and their conversation into Cohen's Kappa.

	Reported			Calculated					
				Presence = Positive			Absence = Positive		
Category	Precision	Recall	$F_{1\ binary}$	$F_{1\ micro}$	$F_{1\ macro}$	Kappa	$F_{1\ micro}$	$F_{1\ macro}$	Kappa
Persuasive	0.81	0.84	0.91	0.947	0.895	0.790	0.701	0.412	−0.173
Audience	0.80	0.99	0.88	0.830	0.781	0.575	0.912	0.907	0.816
Agreement	0.69	0.85	0.76	0.938	0.864	0.728	0.615	0.381	−0.184
Informative	0.76	0.74	0.75	0.919	0.852	0.703	0.599	0.375	−0.250
Mean	0.74	0.78	0.75	0.897	0.680	0.693	0.611	0.379	−0.238
Controversial	0.67	0.64	0.65	0.575	0.563	0.158	0.542	0.488	−0.024
Disagreement	0.60	0.68	0.64	0.688	0.682	0.365	0.539	0.501	0.009
Off-topic	0.62	0.67	0.61	0.492	0.379	−0.237	0.768	0.737	0.474
Sentiment	0.44	0.46	0.43						

meaningful evaluation. As a consequence, we convert the results into Cohen's Kappa. To do so, we reconstruct fp, tp, tn, and fn[9]. But since the binary metrics are always in respect to one class, that we do not know, we have calculate them for both situations. So the table consists of values if the original results came from the presence and the absence of a specific class. We can to the conversation because we have four unknowns and four equations. We have precision and recall. And since we know the distributions of the classes (because we have the samples), we can derive two additional equations:

$$class_0 = tp + fn \qquad class_1 = fp + tn$$

This enables us to solve for the four unknown variables. With them, we can calculate a different metric, in this case Kohen's Kappa. But we only do this for the eight binary classes. The category Sentiment has four classes and thus we cannot reconstruct the original values. In the original paper, the authors reveal that the results for the categories Informative and Controversial are below a simple ridge regression baseline.

We will continue to discuss the values in the evaluation Section 6.1.2 to set our contribution into perspective.

4.2.4. Discussion

There are several issues with the YNACC dataset:

[9]These are the acronyms for false positives, true positives, true negatives, false negatives respectively.

- It is unclear what metric was used for the reported values. Since the authors did not reply to our email, we and others researchers can only speculate. This makes it hard to compare results and sets new inventions into perspective.

- For the consolidation of annotations, the authors decided to randomly assign samples to classes when there is no majority among the annotators. This adds unnecessary noise. It is most likely better to fully remove the samples from dataset.

- The authors did not provide the consolidated assignment for the samples. Therefore, it is not fully clear, how they derived the values. For instance, there are classes marked as 'NA'. It is unclear whether they were removed or kept.

- Two different groups of people annotated the data. One group annotated training and validation and the other group, "expert annotators" the test set. It is true, that the test set is used to evaluate the performance of a model in the wild. However, then the test set cannot be used to asses the performance of the model. A model can only learn what is has been taught.

Nevertheless, it is still the largest corpus of publicly available news comments. We follow the setup of the original paper, including the random assignments to compare our results.

4.3. One Million Posts Corpus

The One Million Posts Corpus[10] (OMPC) is a dataset consisting of German comments from the Austrian newspaper *DerStandard*[11]. The dataset was presented in-depth in the publication by its creators Schabus et al. [63] and thus we only give a brief overview.

It contains 1 million unlabeled comments and 11,773 labeled ones from 2015 to 2016. A comment consists of its text, user votings and pseudonymized author ID but also information about the article is included: the article's text, headline, topic, and timestamps. The comments were labeled by professional comment moderators in seven categories. One category, sentiment, is annotated into three classes. All others are binary classifications. They are presented as follows:

Sentiment The sentiment of a comment for three classes: positive, neutral or negative. For further use, the category is split up into three binary classification setups and denoted as *positive*, *neutral* and *negative*.

Off-topic If a comment is out of the article's topic.

[10] `https://ofai.github.io/million-post-corpus/`
[11] `https://derstandard.at/`

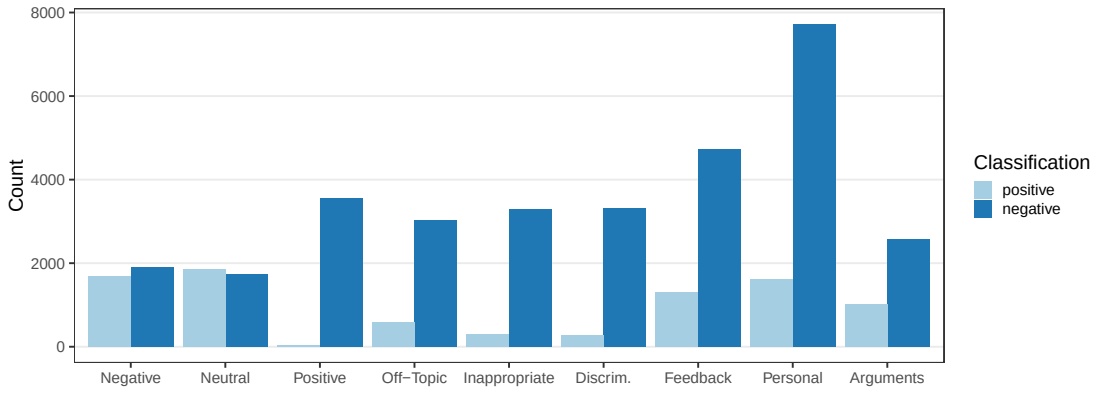


Figure 4.9.: The distributions of classes in OMPC are imbalanced. Also the number of samples varies among the categories.

Inappropriate If inappropriate words were used, i.e., swearwords.

Discriminating If a comment is discriminating, i.e., racist.

Personal If it includes a personal story.

Feedback If feedback is given to the article's author.

Arguments If arguments are used in a comment.

The number of samples per class varies as shown in Figure 4.9. The sampling process was done in an interactive way, driven by annotating moderators. Moderators chose articles under which certain kind of comments appeared often. For instance, under articles about refugees are a lot of racists comments. So they chose comments from these articles for the category Discriminating. This resulted in a strong bias in the data and thus the annotated data were not sampled representatively. As a consequence, not all comments were assigned with the same class (as the different number for each category indicates). In the publication, the authors provide feature-based as well as neural models results as baselines. They will be presented in Section 6.2.2 to compare them with our experiment results.

4.4. Ethical Considerations

In this section, the creation of the news comments datasets is reflected critically.

A major problem is the fact, that commentators did not consent for being part of the dataset.

Their comments get taken out of the context of the newspaper website in which they originally created them. Then it gets added into the dataset that explicitly invites other people to use it (for research or other purposes). The commentators who contributed to the dataset are most probable not even aware of it. "Recognize that privacy is more than a binary value" is one of the "Ten simple rules for responsible big data research" that a consortium of 13 researches and philosophers [76] postulated. Even though the comment is publicly visible on the newspaper's website, does not give computer scientist do not the right to do everything they desire. In a research context, when working with user-generated content the same research ethics should apply that exists for social scientists or in the medical domain. One guiding principle is that research participants should consent to being part of experiments. This principle is the result of the discussion after crimes against humanity were done in the name of research during the German National Socialism[12]. For future datasets, comments should only be considers if the authors actively agrees to take part in research dataset. At least information about the comment's author are not in the datasets. The creators of YNACC fully removed user information whereas the ones for OMPC only created pseudonymized user IDs.

[12]The Nuremberg Code lists ten rules for human experiments in the aftermath of the Nuremberg trials in 1947. https://en.wikipedia.org/wiki/Nuremberg_Code

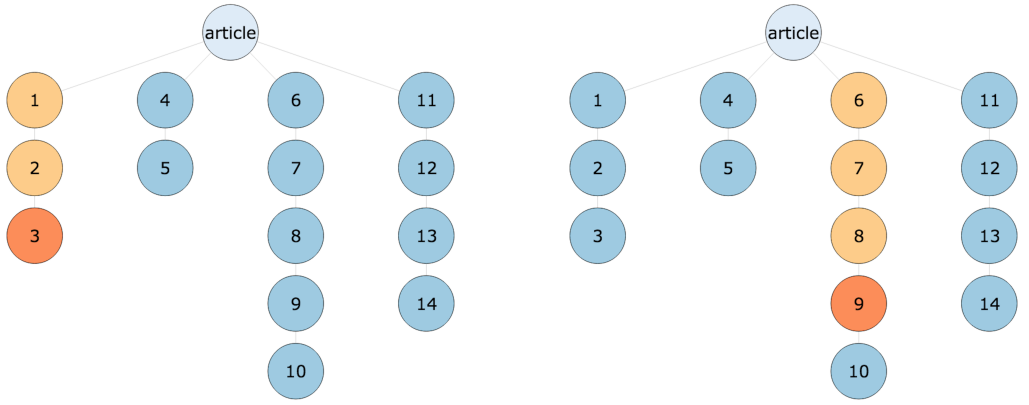
5. Classification of News Comments

Our contribution is two-fold: First, we present a preprocessing technique to capture the conversation of a comment for classification. Second, we adapt ULMFIT for German by training a German language model and publish it for further usage.

5.1. Conversation-aware Classification of News Comments

To exploit the sequential structure of news comment, we propose the preprocessing technique *Prepend Previous* for language-model-based text classification. News comments appear as part of a conversation where the news article acts as conversation starter. Only considering each comment in isolation makes it hard to capture its true meaning. Prepend Previous helps to overcome this restriction. We prepend previous comments or parts of the article to each comment. This is similar to the idea of Kochkina et al. [28] for Branch-LSTM. But they use use only word embeddings to represent text. We use a recent way of using language models for representing text. And since modern language models can capture dependencies over long distances, the content of a comment can be put into the context of the whole conversation.

There exist different ways of how comment section appear on the Web. We focus on sequential discussions where top-level comments appear directly under a news articles and other comments reply to them. For each reply, we prepend the previous comments. To separate the comments, we add special tokens between them. This builds up a long chain of comments. The comment with the corresponding annotation is always the last comment. This is how the model can learn that the last comment in the chain is the comment to classify. The previous comments only give additional information about the conversation. Figure 5.1 illustrates the prepending for two comments. Comment *3* and Comment *9* are the samples to which the previous comments are prepended. For Comment *3*, the final sequence is <t><c>1</c><c>2</c><c>3</c></t> and the original annotation of Comment *3* is attached to it. The <c> and </c> represent the start and the end of comment respectively. Likewise <t> and </t> represent the start and the end of a discussion threads (or sub-dialogue). To accomplish this, one has to iterate over all comments. Then for each comment, one has to gather all comments that are on the way to the parent node up to the root note – the news article. This results in a complexity of $\mathcal{O}(n^2)$ for the algorithm.



(a) <t><c>**1**</c><c>**2**</c><c>**3**</c></t> (b) <t><c>**6**</c><c>**7**</c><c>**8**</c><c>**9**</c></t>

Figure 5.1.: Prepending previous comments at the the example of Comment *3* and Comment *9*. Each comment is encapsulated by the special token (*c*) to mark the start and end of a comment. Also a whole thread is indicated (*t*).

In addition, the article's text and headline can be prepended to top-level comments. There are some variations on how and when top-level comments are enriched. First, there it is up to debate what kind of information about the article is given the model. The article comprises headline, abstract, and the whole text. Also some meta-information such as topic or date of publication are provided. Second, it has to be decided whether to always add information about an article or only top-level comments. To illustrate this with an example of the already mentioned Comment *3*, one could add information before the first comment like so: <t>article<c>1</c><c>2</c><c>3</c></t>. This has one disadvantage. Since multiples comments belong to one article, possibly thousands of comments, the article is duplicated in all the training samples. This may lead to overfitting on the article information such as the title. So the other idea is to not add article's information to Comment *3* and only to top-level comments. For Comment *1*, this results in following encoding: <t>article<c>1</c></t>. The top-level comments do not have any previous comment so no comments can be prepended. Without article information, there would not be any improvement over conversation-agnostic models for them.

With these adaption, there are potential problems. Threads with a lot of comments result in duplication of some comments that appear early on in the discussion. However, in practice internal memory is limited so the samples have to be truncated anyhow. So here it is important to only truncate from the back to prevent losing valuable information of the last comment (with the annotation). So for example only choose the last 1000 tokens of a comment chain. But when cutting the chain on a token basis, it is likely that comments gets cut right in the middle. This

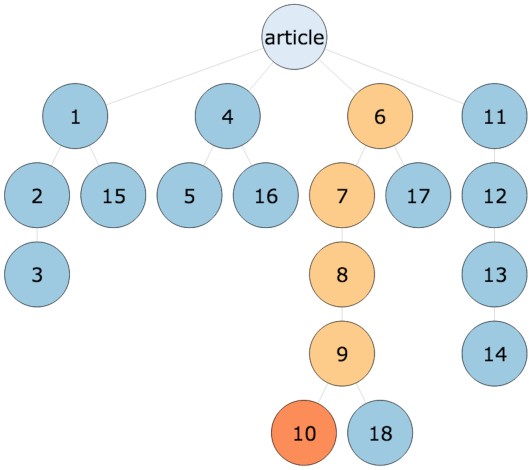


Figure 5.2.: Prepending comments for tree-like comment structures is equivalent. The resulting sequence for Comment *10* is: <t><c>**6**</c><c>**7**</c><c>**8**</c><c>**9**</c><c>**10**</c></t>

is why there are tokens to indicate the start and end of a comment. The model is aware that the beginning of a chain is only a remainder of a previous comment. This way of truncating on a token basis is superior to truncate on a comment basis. One could think to only consider the previous N comments. This would as well keep the length of each sample reasonable. But comments vary in length (see Figure 4.7a) and thus the sheer number of comments is not a guarantor for information. The general idea to give as much information as possible to the model and let it decide which information is useful and which not.

So far, we only considered the sequential structure of news comment. Unlike with discussions on social media on Reddit or Twitter, there is often no nesting of replies. *Prepend Previous* focuses on these sequential forms of discussions. However, it can easily be adapted to to work for tree-like discussion structures. Again, one has to iterate over all comments and go up the tree until the root. This is visualized in Figure 5.2. In this case, the comments chains may have to be truncated more aggressively since it can happen that a comment appear in various sequences. Overfitting on those comment is likely when fine-tuning a language model. However, since there should be no fundamental difference between comments appearing earlier and later in a discussion, the language model should still be able to learn the languages of news comments. Even if it sees comments appearing early in a discussion tree more often.

We implemented[1] the preprocessing in Python with Pandas[2] and Dask[3]. For the language model we used ULMFIT. However, it can be applied to any other language-model-based text

[1] `https://github.com/jfilter/masters-thesis`
[2] `https://pandas.pydata.org`
[3] `https://dask.org`

classification method because the preprocessing remains the same. Recurrent neural network such as LSTMs are especially powerful since last internal state of the LSTM is used for classification. The comment to classify is always the last comment in a chain.

5.2. ULMFIT for German

A general language model is required for using ULMFIT for text classification. Since there does not exists a German pre-trained language model, we create one. Only with it, can we classify German comments of OMPC.

There has been effort to adapt ULMFIT to German but to no avail [59]. A first requirement is the existence of long, high-quality German texts. It is important that the texts are long in order to learn long-term dependencies. This allows the language model to get a deeper understanding of German. We use a dump[4] of the entire German Wikipedia and extract the article's text with WikiExtractor[5]. To gather more data, we use news articles[6]. We crawl news articles with News-Please[7] from several regional and national German newspaper. Only documents with a length of at least 500 characters are kept. The result is a collection of 3.278.657 documents with altogether 1.197.060.244 tokens. Compared to English, German is highly inflective. And the language allows the construction of endless combination of compound nouns. So a simple word-based approach would result in a large vocabulary. It is encouraged to keep the vocabulary small, because in the last layer in a neural language model, a softmax activation function is computed over each entry in the vocabulary. This is computationally intensive and a small vocabulary greatly speeds up the training process. One way to do it, is to split words into sub-word units. For instance, Byte-Pair Encoding (BPE) by Sennrich et al. [65] achieves this. This is a compromise between word-based and character-based approaches. The size of the vocabulary of sub-word units is fixed. So the size of sub-units depend on the vocabulary size. The smaller the vocabulary, the shorter the sub-word units. Czapla et al. [8] successfully applied BPE to ULMFIT for Polish. Since the splitting of word in sub-words is a model (or embedding) on its own, we can use a pre-trained model. Heinzerling and Strube [23] provide pre-trained sub-word models[8] for 275 languages. The authors provide models for the vocabulary size from 1.000 to 200.000. For Polish, Czapla et al. achieved the best results with vocabulary of 20.000. Since Polish and German are both Indo-Germanic languages, a parameter in the similar range should apply for German as well. We choose a the German model with fixed vocabulary size of 25.000.

[4] `https://dumps.wikimedia.org`
[5] `https://github.com/attardi/wikiextractor`
[6] In regard to ethical considerations, we note that we do not have the consent of the journalists who wrote the articles. However, journalists work in commission of a newspaper and no private information is attached to an article. And unlike for news comments, journalists know that their articles get processed automatically by, i.e., search engines.
[7] `https://github.com/fhamborg/news-please`
[8] `https://nlp.h-its.org/bpemb`

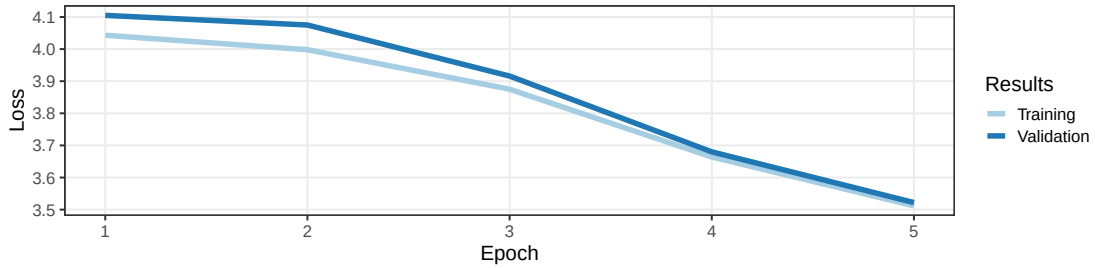


Figure 5.3.: Training curves of the German language model.

It follows an example of how it breaks up the German sentence

> "*Zeitungskommentare sind eine hervorragende Möglichkeit zum Meinungsaustausch*"

into sub-word units:

> ['_zeitungs', 'komment', 'are', '_sind', '_eine', '_hervor', 'ragende', '_möglichkeit', '_zum', '_meinungs', 'austausch', '.']

For instance, the German composite noun *Meinungsaustausch* is split into two sub-word units. The white space is replaced with a special underscore token. We have to follow the preprocessing steps of the pre-trained model. This includes that all text is lowercased and all digits are replaced by a 0.

To train it, we take the default configuration of the English language: an embedding size of 400 and 3 layered LSTM with 1150 hidden activations per layer. But we fully disable dropout. The amount of data is large enough such thats a strong regularization is not needed. We trained for five epochs which took three days on a single GTX1800ti, with a batch size of 128. The learning rate is chosen automatically by the learning rate finder (0.00744..). The training curves are in Figure 5.3 and the final model achieves a perplexity of 33.88... The language model is published for further usage[9].

[9] `https://johannesfilter.com/ulmfit-for-german`

6. Evaluation

We evaluate our contributions on an English and a German datasets of comments datasets: YNACC and OMPC. The datasets were introduced in Section 4.

6.1. Experiments on Yahoo News Annotated Comments Corpus

The Yahoo News Comment Corpus (YNACC) was intensively presented in Section 4.2. We compare across three different approaches: baselines, conversation-agnostic models, and our conversation-aware models.

6.1.1. Setup

For all experiments, we follow the dataset's preprocessing steps of the related work that reported values on YNACC [46]. These were presented in Section 4.2.1. Since there are already published results on this dataset, it sets our contribution into perspective. We train a model for each category separately, as done by the related work. The experiments and their results are managed via Sacred[1]. The setup is different for each method so we describe them in detail in the following.

Baselines

We use three different methods as baseline: Naive Bayes, Ridge Regression and the FastText classifier. All of them are linear classifiers. The first two are traditional machine learning methods and the follow the bag-of-words idea. The FastText classifier also considers N-gram of words. It was introduced by Joulin et al. [26] and it outperforms even neural models while being up to 35.000 times faster. In addition to the data cleaning described in Section 4.2.2, the tokens are lower-cased. The default tokenizer of the respective implementation is used. For Naive Bayes and Ridge Regression, we use the Python package *Scikit-Learn*[2] in version 0.20.2.

[1] `https://github.com/IDSIA/sacred`
[2] `https://scikit-learn.org`

For Naive Bayes, we use *MultinomialNB* with the default parameters. For Ridge Regression, we use the cross validation variant on the training data with, as well, the default parameters. For FastText, we use the Python wrapper of the official implementation[3], version 0.2.0. Because the time to train is short, we use a random search for 1000 times over the following uniformly distributed hyper-parameters: learning rate from 0 to 1, number of epochs from 5 to 50, N-grams for n from 1 to 5, minimum count for each token from 1 to 10.

Conversation-agnostic Neural Models

We use two different language-model-based approaches as comparison models: ULMFIT and BERT. BERT is more advanced than ULMFIT, but it also requires more computation. Because of our financial constraints, we choose ULMFIT as the main reference model. For ULMFIT, we use the author's accompanied implementation in the FastAI library[4] in version 1.0.41. ULMFIT internally uses the SpaCy library[5] version 2.0 to tokenize the text. We as well clean our data as described in Section 4.2.2. We choose a vocabulary size of $30k$ since it covers over 98.2% of the tokens as shown in Figure 4.7. The use all special text preprocessing steps of the default FastAI implementation. These are: tokens are lower cased, but for every uppercase word a special uppercase token is prepended. The same applies for all capitalized tokens. In addition, repeating tokens are replaced by a single token and the number of how often it was repeated gets prepended. For instance, "yes !!!!!" is transformed to "yes xxrep 5 !".

Before training the classifier, the pre-trained WikiText 103 language model needs to get fine-tuned. We choose the datasets of unlabeled comments together with the labeled training comments we noted as $\mathrm{YC}_{\mathrm{LM}}$ in Section 4.2.1. Characteristics of the datasets were presented in the section exhaustively. So we fine-tune the language model until overfitting. Even though the authors suggest to fine-tune cautiously, the number of unlabeled samples should be enough (about 200k). We use random search over the following parameters: epochs 2 to 6, dropout factor from 0.8 to 1.1. The learning rate scheduler is used to determine a learning rate. There are 5 different kind of dropout in the model and the authors suggest to tune a specific dropout multiplier. And the layer factor, which sets different learning rate for each layer, is set to 2.6. The authors obtained the magic numbers for the parameters with grid search and they recommend to follow them. We only save the model with the best validation loss. The validation set consists of the last 10% of the discussion IDs.

For fine-tuning the classifier, we made the experience that results with the one cycle policy scheduling are unstable. The scheduler is suggested by the authors of the ULMFIT paper. The results varied a lot for different runs with the same parameters. This made it hard to compare the results and thus we abandon it. We use the standard Adam optimizer [27] with the weight decay fix [36]. We used a dropout multiplier of 0.6, 0.7 and 0.8 and a batch size of 64. The

[3] https://github.com/facebookresearch/fastText/tree/master/python
[4] https://github.com/fastai/fastai
[5] https://spacy.io

learning rate is set to 0.001, since the learning rate finder did not yield useful results. The maximum number of epochs is 200 but the learning is stopped, when the Cohen's Kappa score does not improve for 10 epochs ("early stopping"). Only the model with the best Cohen's Kappa score is saved.

BERT by Devlin et al. [12] was already briefly mentioned in Section 3.3. While we focus on ULMFIT in this work, the approach of BERT is similar. It requires more computation which is the reason we cannot use it for all our experiments. It is not feasible for us to fine-tune the language model on our domain of news comments. Thus, we use only a pre-trained BERT model. Out of familiarity with PyTorch, we use the port to PyTorch[6], which replicated the results of the original TensorFlow implementation. We chose the smaller English *Bert Cased* model because we hypothesize that the casing in comments bears information. The approach operates on sub-word basis and did not require any preprocessing. First, we manually explored a range of useful hyper-parameters and then did a grid search over them. The parameters were: number of epochs from 3 to 10, the learning $5e^{-6}$, $5e^{-7}$, $5e^{-8}$.

Prepend Previous

For computational reason, we only evaluate Prepend Previous on the ULMFIT language model approach. The setup is the same as described for conversation-agnostic model in the previous section. Only the dropout multipliers are modified to 0.9, 1, 1.1, 1.2, 1.3. Comments are cut to 200, the total maximum token length is 1400 to optimize RAM usage. The learning rate is set to 0.001 and batch size to 64. Maximum number of epochs is 200 and early stopping is used. The samples were truncated in the beginning. There are five different variations of Prepend Previous:

Table 6.1.: Variations of Prepend Previous

Variation	Description
TX_1	text
TX_2	text, the language model is not fine-tuned
HL_1	text, headline is prepended to all top-level comments all the time
HL_2	text, headline is only prepended to top-level comments if they would be alone
ART	text, headline, article (if available)

The difference between TX_1 and TX_2 is whether the fine-tuning of the language model has to be done on preprocessed comments (TX_1) or on non-preprocessed comments (TX_2). The language model for TX_2 was identical to the conversation-agnostic comparison models. For HL_1, information about the article are always included. For HL_2, the information was only

[6] `https://github.com/huggingface/pytorch-pretrained-BERT`

included for top-level comments. Since we do not have all articles, we cannot add all the articles for ART.

6.1.2. Results

As metrics, we use $F_{1\ \text{micro}}$, which is equivalent to accuracy in this setup, $F_{1\ \text{macro}}$ and, Cohen's Kappa (or short Kappa). We explained Kappa in Section 3.5 and gives a meaningful value even on unbalanced categories. The full results including the test values are in Appendix A. We only focus on validation results, since the related work does not report test results. In addition, the test results differ greatly from the validation results across all models – baselines as well as neural models. For instance, Kappa increased in the category Audience by about 0.1 points for almost all models. Also a simple ridge regression pushes the Kappa score from 0.52 to 0.66. Even though the ridge regression baseline generally achieves poor results. For all other categories, the performance deteriorates drastically on the test dataset. For Off-topic, the differences are the most significant. While on the validation sets, the Kappa score drops to below 0.2 for all models while performing on the validation set up to 0.48. The further implications of these results will be discussed later.

In Table 6.2 are the results of the baseline compared to conversation-agnostic models. The left side of the table consists of the comment's text only. The right side of the text and a binary feature of whether a comment is a top-level comment or a reply. This feature greatly improves the performance for the category Audience for all models. However, for Agreement, Informative, Controversial and Off-topic the results decrease. For the remaining categories are the differences negligible. In general, the FastText classifier outperformed the other baselines by far. On the comment's text and reply, it achieved the best results on Audience by a margin of 0.04 $F_{1\ \text{micro}}$ score. BERT is overall the winner even though the language model was not fine-tuned on comments. For Controversial, it outperforms other approaches by a margin of about 0.05 $F_{1\ \text{micro}}$ score. ULMFIT achieves slightly worse performance than BERT but still outperforms the baselines on the vast majority of categories.

In Table 6.3 are the results of conversation-aware model in comparison to three best-scoring conversation-agnostic models from Table 6.2. On average, context-aware models achieve superior results. The $F_{1\ \text{micro}}$, $F_{1\ \text{micro}}$ and Kappa scores increased by 1.53%, 3.08% and 11.33%, respectivly. Off-topic is the greatest beneficiary with an increase by 36.26% in Kappa score. The change in percent is visualized in Figure 6.2. However, there are differences among the categories. For Audience and Mean, the performance decreased. For others such as Agreement, Disagreement, Controversial and Off-topic all metrics improved. We tested several variations of Prepend Previous. The first question was whether it is important to fine-tune the language model with preprocessed tokens. TX_1 outperforms TX_2 in almost all metrics for all categories. Then, it seems that adding more information does not always seem to help. Especially, adding information about the article does in some cases decrease the results. The results of HL_1 and HL_2 differ only slightly. Only for Persuasive and Off-Topic HL_2 outperforms clearly. Adding

Table 6.2.: Experiment results of baselines and conversation-agnostic comparison models on the YNACC validation dataset. The highst value for each row is highlighted. Since BERT (BT) is out of competition, its values are underlined if it would have achieved the highest results. Naive Bayes (NB), Ridge Regression (RR), FastText Classifier (FT), ULMFIT (UF).

		Text					Text + Reply				
Category	Metric	NB	RR	FT	UF	BT	NB	RR	FT	UF	BT
Persuasive	$F_{1\ micro}$	0.814	0.855	0.871	0.845	0.864	0.835	**0.876**	0.874	0.830	0.862
	$F_{1\ macro}$	0.544	0.645	0.681	0.634	0.721	0.578	0.655	**0.694**	0.684	0.721
	Kappa	0.097	0.298	0.369	0.273	0.442	0.168	0.326	**0.393**	0.370	0.443
Audience	$F_{1\ micro}$	0.686	0.698	0.741	0.746	0.795	0.783	0.809	**0.876**	0.834	0.831
	$F_{1\ macro}$	0.586	0.594	0.677	0.708	0.765	0.728	0.753	**0.829**	0.790	0.803
	Kappa	0.207	0.228	0.369	0.417	0.532	0.470	0.524	**0.662**	0.591	0.608
Agreement	$F_{1\ micro}$	0.885	0.879	0.902	0.907	0.903	0.885	0.885	0.905	**0.912**	0.903
	$F_{1\ macro}$	0.497	0.494	0.673	**0.719**	0.709	0.497	0.555	0.672	0.696	0.727
	Kappa	0.046	0.035	0.357	**0.444**	0.423	0.046	0.140	0.357	0.404	0.457
Informat.	$F_{1\ micro}$	0.818	0.818	0.816	**0.866**	0.862	0.826	0.821	0.821	0.847	0.845
	$F_{1\ macro}$	0.558	0.528	0.614	**0.683**	0.748	0.553	0.494	0.623	0.679	0.717
	Kappa	0.136	0.087	0.233	**0.378**	0.496	0.135	0.037	0.251	0.362	0.434
Mean	$F_{1\ micro}$	0.813	0.813	0.826	**0.842**	0.849	0.811	0.813	0.823	0.840	0.835
	$F_{1\ macro}$	0.640	0.630	0.666	0.726	0.740	0.641	0.619	0.674	**0.752**	0.723
	Kappa	0.296	0.280	0.348	0.457	0.484	0.297	0.263	0.359	**0.504**	0.448
Controv.	$F_{1\ micro}$	0.663	0.641	0.687	**0.706**	0.735	0.691	0.703	0.704	**0.706**	0.758
	$F_{1\ macro}$	0.639	0.574	0.644	**0.675**	0.700	0.661	0.590	0.663	0.672	0.722
	Kappa	0.286	0.149	0.289	**0.353**	0.400	0.325	0.212	0.326	0.346	0.445
Disagree.	$F_{1\ micro}$	0.634	0.646	0.703	0.756	0.739	0.660	0.701	0.737	**0.778**	0.777
	$F_{1\ macro}$	0.626	0.639	0.682	0.745	0.729	0.654	0.687	0.724	**0.772**	0.770
	Kappa	0.254	0.280	0.368	0.491	0.459	0.312	0.375	0.448	**0.546**	0.540
Off-topic	$F_{1\ micro}$	0.682	0.663	0.687	**0.749**	0.737	0.704	0.668	0.680	0.706	0.740
	$F_{1\ macro}$	0.543	0.539	0.608	**0.670**	0.663	0.601	0.625	0.609	0.644	0.676
	Kappa	0.127	0.105	0.222	**0.353**	0.334	0.222	0.252	0.222	0.290	0.357
Sentiment	$F_{1\ micro}$	0.548	0.533	0.564	0.622	0.633	0.554	0.536	0.577	**0.628**	0.638
	$F_{1\ macro}$	0.248	0.290	0.381	0.417	0.471	0.256	0.303	0.358	**0.448**	0.450
	Kappa	0.099	0.102	0.228	0.315	0.377	0.107	0.133	0.232	**0.322**	0.374
Average	$F_{1\ micro}$	0.727	0.727	0.755	0.782	0.790	0.749	0.756	0.777	**0.786**	0.798
	$F_{1\ macro}$	0.542	0.548	0.625	0.664	0.694	0.574	0.586	0.649	**0.681**	0.701
	Kappa	0.172	0.173	0.309	0.386	0.438	0.231	0.251	0.361	**0.415**	0.456

Table 6.3.: Experiment results on the YNACC validation dataset comparing conversation-agnostic models with conversation-aware models. The former group of models stems from Table 6.2. The latter were explained in Table 6.1. The highest value in each row is highlighted.

		Conversation-agnostic			Conversation-aware				
Category	Metric	UF_{text}	FT_{reply}	UF_{reply}	TX_1	TX_2	HL_1	HL_2	ART
Persuasive	$F_{1\ micro}$	0.845	**0.874**	0.830	0.847	0.845	0.847	0.871	0.811
	$F_{1\ macro}$	0.634	0.694	0.684	0.703	0.674	0.688	**0.720**	0.651
	Kappa	0.273	0.393	0.370	0.407	0.348	0.378	**0.442**	0.303
Audience	$F_{1\ micro}$	0.746	**0.876**	0.834	0.836	0.827	0.839	0.826	0.833
	$F_{1\ macro}$	0.708	**0.829**	0.790	0.796	0.777	0.794	0.797	0.785
	Kappa	0.417	**0.662**	0.591	0.600	0.569	0.600	0.596	0.583
Agreement	$F_{1\ micro}$	0.907	0.905	0.912	**0.921**	0.912	**0.921**	0.915	0.909
	$F_{1\ macro}$	0.719	0.672	0.696	0.761	0.746	0.761	**0.763**	0.731
	Kappa	0.444	0.357	0.404	0.526	0.494	0.526	**0.529**	0.466
Informative	$F_{1\ micro}$	**0.866**	0.821	0.847	0.847	0.843	0.859	0.845	0.826
	$F_{1\ macro}$	0.683	0.623	0.679	**0.695**	0.672	0.693	0.690	0.666
	Kappa	0.378	0.251	0.362	0.391	0.348	**0.392**	0.381	0.334
Mean	$F_{1\ micro}$	0.842	0.823	0.840	0.814	0.840	**0.845**	0.838	0.826
	$F_{1\ macro}$	0.726	0.674	**0.752**	0.726	0.724	0.749	0.738	0.727
	Kappa	0.457	0.359	**0.504**	0.453	0.453	0.500	0.477	0.455
Controversial	$F_{1\ micro}$	0.706	0.704	0.706	0.728	0.725	0.727	0.728	**0.739**
	$F_{1\ macro}$	0.675	0.663	0.672	0.679	0.672	0.684	0.686	**0.689**
	Kappa	0.353	0.326	0.346	0.360	0.346	0.369	0.372	**0.380**
Disagreement	$F_{1\ micro}$	0.756	0.737	0.778	**0.794**	0.782	0.792	0.785	0.763
	$F_{1\ macro}$	0.745	0.724	0.772	**0.786**	0.773	0.783	0.777	0.756
	Kappa	0.491	0.448	0.546	**0.572**	0.547	0.567	0.554	0.513
Off-topic	$F_{1\ micro}$	0.749	0.680	0.706	0.754	0.763	0.758	**0.770**	0.753
	$F_{1\ macro}$	0.670	0.609	0.644	0.722	0.712	0.702	**0.740**	0.711
	Kappa	0.353	0.222	0.290	0.444	0.426	0.407	**0.481**	0.423
Sentiment	$F_{1\ micro}$	0.622	0.577	0.628	**0.647**	0.615	0.603	0.610	0.608
	$F_{1\ macro}$	0.417	0.358	**0.448**	0.408	0.420	0.409	0.411	0.421
	Kappa	0.315	0.232	0.322	**0.361**	0.321	0.318	0.328	0.324
Average	$F_{1\ micro}$	0.782	0.777	0.786	**0.798**	0.794	**0.798**	**0.798**	0.785
	$F_{1\ macro}$	0.664	0.649	0.681	0.697	0.685	0.695	**0.702**	0.681
	Kappa	0.386	0.361	0.415	0.457	0.428	0.450	**0.462**	0.420

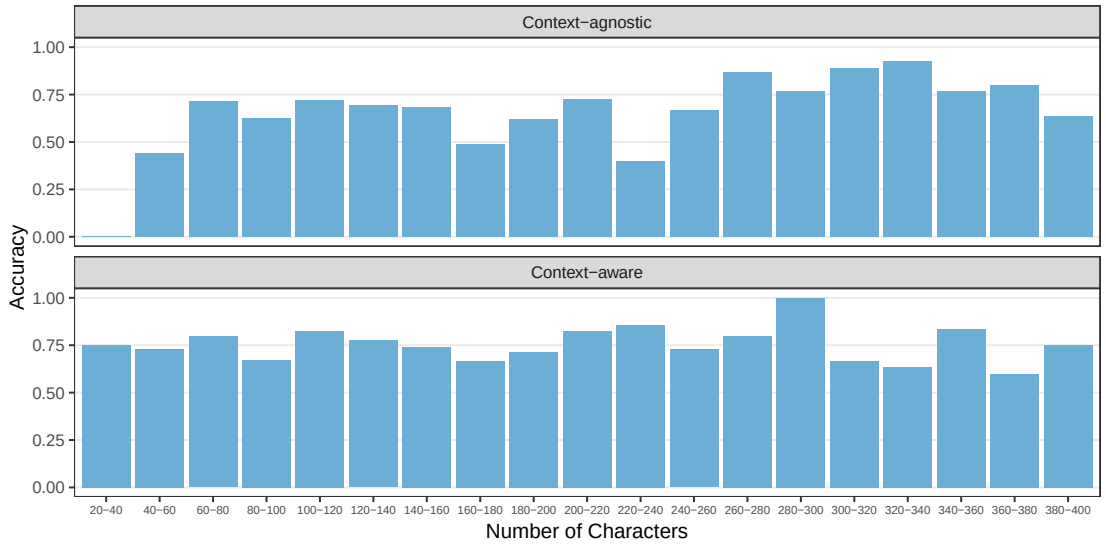


Figure 6.1.: A comparison of the performance between a conversation-agnostic model (only reply) with a conversation-aware model (PP, headline only root) in regard to length of comments.

even more information, such as the whole article text, seem to decrease the performance almost all categories. Only Controversial achieved the best results for ART. In order to understand the differences for the conversation-aware models, we investigate the effect of it at the example of the category Off-topic. The conversation-aware model outperform conversation-agnostic models especially when the comments is short. For longer comments, the performance decreased. This is visualized in Figure 6.1. The overall performance increased since the majority of the comments is short (Figure 4.7a).

Finally, we speculate whether we have outperformed previously reported values. Unfortunately, we cannot be sure how exactly the metrics were created. The values were presented and also discussed in Section 4.2.3. The Kappa values that are presented come from this section. We focus on Kappa to compare the performance of a model with a single metric. There are multiple categories where the reported results are either impressive or poor: Persuasive (0.79 or -0.173), Mean (0.693 or -0.238), Agreement (0.728 or -0.184), Off-topic (0.474 or -0.237). For Audience, the values are so clear (0.816 or 0.575). It is hard to make a definite statement about them. However, for other categories we outperformed them clearly. We can say for sure that we have outperformed YNACC for Controversial (0.158 or -0.024) and Disagreement (0.365 or 0.009). The authors reveal in their publication that Informative (0.703 or -0.25) is below their ridge regression baseline. So the value of 0.703 is impossible. This means we have outperformed them clearly. Also Sentiment with a reported F_1 of 0.43 is below our results of over 0.6 for the majority of approaches. Since this is a four-class setup, the values ought to be averaged in some way. We do not know how but we outperformed them in any way.

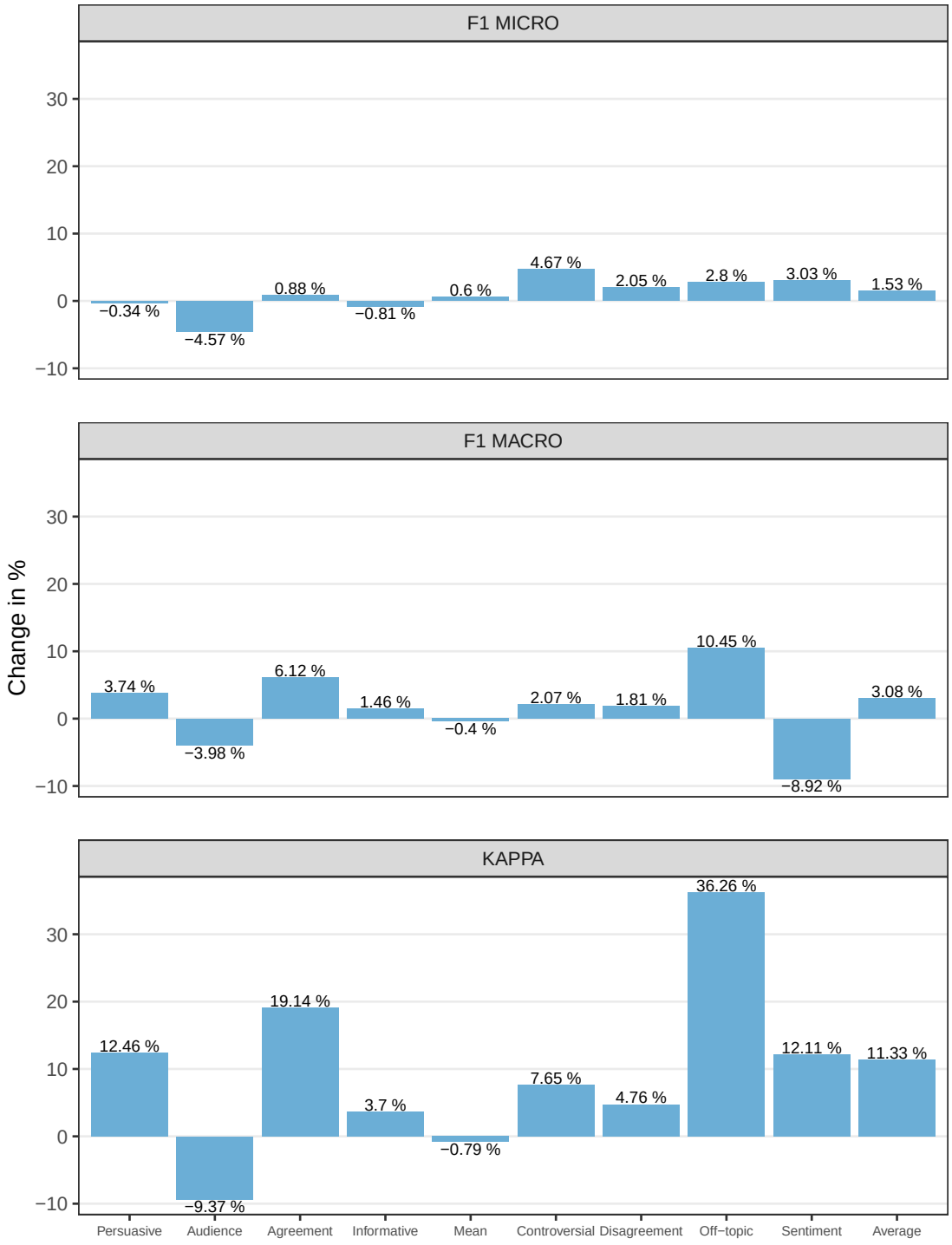


Figure 6.2.: The change in percent of the best conversation-agnostic model vs the best conversation-aware model based on Kappa.

6.1.3. Discussion

The issue concerning the reported values makes it hard to compare our results. However, we could demonstrate that we outperformed previous results in at least three categories. We were mainly reporting on the results on the validation set. This has two reasons. We want to set our work into perspective – even this was only possible for three categories. Moreover, the test values the dropped across all models considerably. Generally, it is advised to report values on the test set, because it is likely that one has overfit parameters on the validation set. So the validation results must be considered carefully. However, our question was primarily whether it is essential to consider the conversational context is important. And this can be answered by working on the validation as well as test results. And the test sets contains further uncertainty, since they were annotated from a different group of people than the training and validation set. Also the unnecessary noise of randomly assigning samples to classes, where no majority wins is a problem. So the drop in the models can be explained by the noise in the data as characteristics of the data. In news comments, the words greatly differ from article to article. More annotated data is needed to create better datasets to create more accurate results.

Our results allow to believe that the conversational context helped for certain categories. It is not needed for all categories. But for categories, that relate in some way to previous comments, the performance with context increased. This allowed the model to get a deeper understanding of news comments. To decide whether a comments agrees or disagrees, it helps to understand what was written before. Also to decide whether a comment is off-topic, the previous comment help to make clear what the actual topic is. That adding information about the article decreases the performance, may a problem of this approach. Information about an article may duplicated immensely in the trainings sets. So here Prepend Previous has some serious short-comings. Further research is needed to overcome this. Using separate sub-networks for the article and the comment may help.

6.2. Experiments on One Million Posts Corpus

In this section, we conduct two experiment on the dataset of German news which was described in Section 4.3.

6.2.1. Setup

The conversation-agnostic model follows the same setup as already described in Section 6.1.1 for experiments on YNACC. A general German language model is required and it was presented in Section 5.2. The language model is fine-tuned on the unlabeled comments until overfitting. The OMPC datasets as well contains cross validation folds. We re-use these to compare our

results. The model is trained for 100 epochs but stopped if the Kappa score did not improve for over 40 epochs. The test value of the last epoch is used to determine a fold's performance. The following parameters were used: a dropout multiplier of 0.8, a learning rate of 0.001, and a batch size of 128.

For the conversation-aware model, the comments were preprocessed with Prepend Previous. Since the comments were originally created in a tree-like discussion structure, the Prepend Previous differs from YNACC. It can happen that an intermediate-comment can have more than one child. Thus this comment appears multiple times in the samples. To avoid duplicates of those comments, all comment chains are truncated to 400 tokens. This should ensure that the number of duplications is low. Since classes are unbalanced, the loss was weighted according to the distribution of classes in the appropriate fold.

6.2.2. Results

The results are presented in Table 6.4. We follow the metrics of related literature and show the Precision, Recall and F_1 in regard to the minority class. The conversation-aware model achieved lower results than a conversation-agnostic model in all categories. However, the context-agnostic model outperformed previously reported results besides for the categories Feedback and Off-topic. There does not exists a previously reported result for Neutral. Häring et al. [22] only reported results for Feedback. The best value of all of the experiments by Schabus et al. [62] are chosen as reference. The F_1 increased for Positive and Inappropriate by 79.3% and 25.1%, respectively.

6.2.3. Discussion

Further research is needed to evaluate whether the conversational context improves the performance for the OMPC datasets. The current results for context-aware models are far below conversation-agnostic models. This may have various reasons. The amount of annotated is low and especially the number of test samples with for the most categories of about 300 is low. This leads to fluctuating performances across the folds. Different variations of early stopping should be explored. Setting the same parameters for all folds may not have been the best decision. For some folds, the model needs to get more carefully trained with more regularization. In other folds, the model did not converge at all. To tune hyper-parameters, the training folds should be split into an addition validation fold. Then, the parameters need to get tuned on the validation set before measuring the final performance on the test set. Nevertheless, the conversation-agnostic model outperformed the featured-based models by Schabus in almost all cases. Even though there is, i.e., no information about the casing in the text. The casing in news comments bears information. So for the future, a German language model with casing should be trained.

Table 6.4.: The cross-validation results on OMPC. The best models by Schabus et al. [62], based on their F_1 score, are chosen as reference. Häring et al.[22] only report results for Feedback. Only the F_1 score is highlighted

Category	Metric	Schabus [62]	Häring [22]	Context-agnostic	Context-aware
Negative	Precision	0.6112		0.6044	0.5802
	Recall	0.6014		0.6168	0.5311
	F_1	0.6063		**0.6089**	0.5290
Neutral	Precision			0.6124	0.5478
	Recall			0.6155	0.5305
	F_1			**0.6073**	0.5089
Positive	Precision	0.1020		0.4850	0.3299
	Recall	0.3488		0.2422	0.1797
	F_1	0.1579		**0.2831**	0.1418
Off-topic	Precision	0.2472		0.4637	0.4344
	Recall	0.6086		0.2771	0.2169
	F_1	**0.3516**		0.3431	0.2603
Inappr	Precision	0.1340		0.4585	0.4531
	Recall	0.5776		0.2010	0.1358
	F_1	0.2175		**0.2720**	0.1939
Discrim	Precision	0.1207		0.3288	0.4881
	Recall	0.5922		0.1512	0.1080
	F_1	0.2005		**0.2037**	0.1596
Feedback	Precision	0.5311	0.85	0.8178	0.8424
	Recall	0.7356	0.83	0.6383	0.5362
	F_1	0.6168	**0.84**	0.7099	0.6443
Personal	Precision	0.6247		0.8787	0.7329
	Recall	0.8123		0.6271	0.6516
	F_1	0.7063		**0.7188**	0.6670
Argument	Precision	0.5457		0.7612	0.6854
	Recall	0.7652		0.5644	0.5592
	F_1	0.6371		**0.6430**	0.6026

7. Conclusions and Future Work

In this master's thesis, we investigated the importance of the conversational structure for automatically classifying news comments. We used the state-of-the-art methods for text classification which involve transfer learning with language models. Language models create powerful text representation as a by-product of being trained to predict the next word based on previous words. We introduced the preprocessing technique *Prepend Previous* to exploit the sequential structure of news comments. Prepending previous comments to each comment allows the language model to grasp the whole conversation around a news article. We demonstrate how Prepend Previous can be applied to ULMFIT, but it could be applied to any language-model-based classification technique.

We successfully conducted experiments on the English news comments dataset YNACC. With conversation-aware models, the performances for several categories increased over conversation-agnostic models. For instance, to detect whether a comment agrees or disagrees with another comment benefits from our technique. But also the results for other categories increased by the conversation-awareness: Off-topic, Controversial, Persuasive, or Sentiment. We conclude that conversation-aware models increase the classification performance for categories that require a deeper understanding of a comment's surrounding. With the right preprocessing steps, we showed that language models are able to accomplish this. We as well investigated whether adding information about the article improves the models even further. This is mostly not the case. More research is needed to build models that relate comments to their article. Our method has the disadvantage of duplicating information about the article in multiple training samples. This may lead to overfitting on, i.e., the headline of a news article. It can be overcome by introducing separate models for encoding information about the article and comments. In addition, we applied the approach to German news comments dataset OMPC. But more experiments are required to draw conclusions in regard to the conversation-awareness. However, we successfully applied ULMFIT to them and improve on previously reported results in six out of nine categories. We publish a German language model for further usage.

There are also some drawbacks of Prepend Previous. By prepending comments, the size of each sample grew immensely, in our experiments by a factor of 10. A language model by itself takes considerable resources to train and this modification increases them even further. So for future work, a mechanism to reduce training time should be explored. One could think of incorporating "skip connections" that let the model skip over previous comments if they are not important. It is rarely the case, that all previous comments are essential. This would also help to reduce the ecological footprint. One other direction is improving language models by adding

metadata. Text rarely appears in isolation so metadata such as timestamps should be incorporated. Additional performance improvements can be gained by using recent Transformer-based language models, i.e., the Transformer-Xl [9]. They outperform LSTM-based language models and should capture the long-term dependencies of news comments conversations better.

The lack of high-quality datasets of news comments was a major problem of this work. Datasets, especially annotated ones, are an integral part of machine learning. Without carefully created datasets, the model cannot learn the characteristics of the samples. For news comments datasets, the number of annotated comments is low (<10k) and the datasets have issues as noted in Section 4. For future work, a large corpus of annotated comments would help the research field immensely. The already annotated data could simplify the process of creating new training samples. First, one has to train a model on the available annotated data. Second, the model predicts the classes of unlabeled comments. Third, annotators need to verify the comments manually. The process is significantly shorter then planning and conducting annotations "from scratch". The German dataset OMPC is well suited for this approach since not all annotated comments are labeled for all categories.

For the future, the NLP community ought to critically reflect on what research problems it is working on. So far, the amount of research done on news comments is oversee-able. But doing research on news comments directly supports newspapers. Even in western societies, the free press is under economic and political pressure[1]. The media are often being named the fourth pillar of democracy. As the recent political development demonstrates, democracy is not self-evident and has to be protected against authoritarian forces. This can be done through, i.e., independent and high-quality journalism. NLP researches should support journalism in order to defend democracy and to continue to hold the powerful accountable.

[1] https://www.theguardian.com/uk-news/2019/apr/11/julian-assange-arrested-at-ecuadorian-embassy-wikileaks

Bibliography

[1] Akbik, A., Blythe, D., and Vollgraf, R. Contextual String Embeddings for Sequence Labeling. In *Proceedings of the 27th International Conference on Computational Linguistics* (2018), pp. 1638–1649.

[2] Bengio, Y., Ducharme, R., Vincent, P., and Janvin, C. A Neural Probabilistic Language Model. *The Journal of Machine Learning Research 3* (2003), 1137–1155.

[3] Bojanowski, P., Grave, E., Joulin, A., and Mikolov, T. Enriching Word Vectors with Subword Information. *Transactions of the Association for Computational Linguistics 5* (2017), 135–146.

[4] Cer, D., Yang, Y., Kong, S.-y., Hua, N., Limtiaco, N., St. John, R., Constant, N., Guajardo-Cespedes, M., Yuan, S., Tar, C., Strope, B., and Kurzweil, R. Universal Sentence Encoder for English. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations* (2018), pp. 169–174.

[5] Chen, D., Ma, S., Yang, P., and Sun, X. Identifying High-Quality Chinese News Comments Based on Multi-Target Text Matching Model. *arXiv pre-print* (2018).

[6] Cohen, J. A Coefficient of Agreement for Nominal Scales. *Educational and Psychological Measurement 20*, 1 (1960), 37–46.

[7] Collobert, R., and Weston, J. A Unified Architecture for Natural Language Processing: Deep Neural Networks with Multitask Learning. In *Proceedings of the 25th International Conference on Machine Learning* (2008), pp. 160–167.

[8] Czapla, P., Howard, J., and Kardas, M. Universal Language Model Fine-Tuning with Subword Tokenization for Polish. *arXiv pre-print* (2018).

[9] Dai, Z., Yang, Z., Yang, Y., Carbonell, J. G., Le, Q. V., and Salakhutdinov, R. Transformer-XL: Attentive Language Models Beyond a Fixed-Length Context. *arXiv pre-print* (2019).

[10] Davidson, T., Warmsley, D., Macy, M., and Weber, I. Automated Hate Speech Detection and the Problem of Offensive Language. In *Proceedings of the 11th International AAAI Conference on Web and Social Media* (2017), pp. 512–515.

[11] Deng, J., Dong, W., Socher, R., Li, L., and and. ImageNet: a Large-Scale Hierarchical Image Database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition* (2009), pp. 248–255.

[12] Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *arXiv pre-print* (2018).

[13] Diakopoulos, N. Picking the NYT Picks: Editorial criteria and automation in the curation of online news comments. *International Symposium on Online Journalism 6*, 1 (2015), 147–166.

[14] Diakopoulos, N., and Naaman, M. Towards Quality Discourse in Online News Comments. In *Proceedings of the ACM 2011 Conference on Computer Supported Cooperative Work* (2011), pp. 133–142.

[15] Diakopoulos, N. A. The Editor's Eye: Curation and Comment Relevance on the New York Times. In *Proceedings of the 18th ACM Conference on Computer Supported Cooperative Work* (2015), pp. 1153–1157.

[16] Ferri, C., Hernández-Orallo, J., and Modroiu, R. An Experimental Comparison of Performance Measures for Classification. *Pattern Recognition Letters 30*, 1 (2009), 27–38.

[17] Gao, L., and Huang, R. Detecting Online Hate Speech Using Context Aware Models. In *Proceedings of the International Conference Recent Advances in Natural Language Processing* (2017), pp. 260–266.

[18] Ghosh, D., Richard Fabbri, A., and Muresan, S. The Role of Conversation Context for Sarcasm Detection in Online Interactions. In *Proceedings of the 18th Annual SIGdial Meeting on Discourse and Dialogue* (2017), pp. 186–196.

[19] Goldberg, Y., and Hirst, G. *Neural Network Methods in Natural Language Processing*. Morgan & Claypool Publishers, 2017.

[20] Gómez, V., Kaltenbrunner, A., and López, V. Statistical Analysis of the Social Network and Discussion Threads in Slashdot. In *Proceedings of the 17th International Conference on World Wide Web* (2008), pp. 645–654.

[21] Grave, E., Bojanowski, P., Gupta, P., Joulin, A., and Mikolov, T. Learning Word Vectors for 157 Languages. In *Proceedings of the International Conference on Language Resources and Evaluation* (2018).

[22] Häring, M., Loosen, W., and Maalej, W. Who is Addressed in this Comment?: Automatically Classifying Meta-Comments in News Comments. In *Proceedings of the ACM on Human-Computer Interaction - CSCW* (2018), pp. 67:1–67:20.

[23] Heinzerling, B., and Strube, M. BPEmb: Tokenization-free Pre-trained Subword Embeddings in 275 Languages. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation* (2018).

[24] Hochreiter, S., and Schmidhuber, J. Long Short-Term Memory. *Neural Computation 9*, 8 (1997), 1735–1780.

[25] Howard, J., and Ruder, S. Universal Language Model Fine-tuning for Text Classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)* (2018), pp. 328–339.

[26] Joulin, A., Grave, E., Bojanowski, P., and Mikolov, T. 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* (2017), pp. 427–431.

[27] Kingma, D. P., and Ba, J. Adam: A Method for Stochastic Optimization. In *International Conference on Learning Representations* (2015).

[28] Kochkina, E., Liakata, M., and Augenstein, I. Turing at SemEval-2017 Task 8: Sequential Approach to Rumour Stance Classification with Branch-LSTM. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)* (2017), pp. 475–480.

[29] Kolhatkar, V., and Taboada, M. Constructive Language in News Comments. In *Proceedings of the First Workshop on Abusive Language Online* (2017), pp. 11–17.

[30] Kolhatkar, V., and Taboada, M. Using New York Times Picks to Identify Constructive Comments. In *Proceedings of the 2017 EMNLP Workshop on Natural Language Processing meets Journalism* (2017), pp. 100–105.

[31] Kolhatkar, V., Wu, H., Cavasso, L., Francis, E., Shukla, K., and Taboada, M. The SFU Opinion and Comments Corpus: A Corpus for the Analysis of Online News Comments. *pre-print* (2018).

[32] Krizhevsky, A., Sutskever, I., and Hinton, G. E. ImageNet Classification with Deep Convolutional Neural Networks. In *Advances in Neural Information Processing Systems 25*. 2012, pp. 1097–1105.

[33] Ksiazek, T. B., and Springer, N. User Comments in Digital Journalism: Current research and future directions. In *The Routledge Handbook of Developments in Digital Journalism Studies*, 1 ed. Routledge, 2018, pp. 475–486.

[34] Lampe, C., and Resnick, P. Slash(Dot) and Burn: Distributed Moderation in a Large Online Conversation Space. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (2004), pp. 543–550.

[35] Loosen, W., Häring, M., Kurtanovic, Z., Merten, L., Reimer, J., Roessel, L. v., and Maalej, W. Making Sense of User Comments. Identifying Journalists' Requirements for a Software Framework. *Studies in Communication | Media 6*, 4 (2017), 333–364.

[36] Loshchilov, I., and Hutter, F. Decoupled Weight Decay Regularization. In *International Conference on Learning Representations* (2019).

[37] Maas, A. L., Daly, R. E., Pham, P. T., Huang, D., Ng, A. Y., and Potts, C. Learning Word Vectors for Sentiment Snalysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1* (2011), pp. 142–150.

[38] McCann, B., Bradbury, J., Xiong, C., and Socher, R. Learned in Translation: Contextualized Word Vectors. In *Advances in Neural Information Processing Systems* (2017), pp. 6294–6305.

[39] Merity, S., Keskar, N. S., and Socher, R. Regularizing and Optimizing LSTM Language Models. *arXiv pre-print* (2017).

[40] Merity, S., Xiong, C., Bradbury, J., and Socher, R. Pointer Sentinel Mixture Models. In *International Conference on Learning Representations* (2017).

[41] Mikolov, T., Chen, K., Corrado, G., and Dean, J. Efficient Estimation of Word Representations in Vector Space. *arXiv pre-print* (2013).

[42] Mikolov, T., Karafiát, M., Burget, L., Cernocký, J., and Khudanpur, S. Recurrent Neural Network Based Language Model. In *INTERSPEECH* (2010), ISCA, pp. 1045–1048.

[43] Mikolov, T., Sutskever, I., Chen, K., Corrado, G., and Dean, J. Distributed Representations of Words and Phrases and Their Compositionality. In *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2* (2013), pp. 3111–3119.

[44] Miura, Y., Kano, R., Taniguchi, M., Taniguchi, T., Misawa, S., and Ohkuma, T. Integrating Tree Structures and Graph Structures with Neural Networks to Classify Discussion Discourse Acts. In *Proceedings of the 27th International Conference on Computational Linguistics* (2018), pp. 3806–3818.

[45] Napoles, C., Pappu, A., and Tetreault, J. R. Automatically Identifying Good Conversations Online (Yes, They Do Exist!). In *Proceedings of the Eleventh International AAAI Conference on Web and Social Media (ICWSM 2017)* (2017), pp. 628–631.

[46] Napoles, C., Tetreault, J., Pappu, A., Rosato, E., and Provenzale, B. Finding Good Conversations Online: The Yahoo News Annotated Comments Corpus. In *Proceedings of The 11th Linguistic Annotation Workshop* (2017), pp. 13–23.

[47] Nobata, C., Tetreault, J., Thomas, A., Mehdad, Y., and Chang, Y. Abusive Language Detection in Online User Content. In *Proceedings of the 25th International Conference on World Wide Web* (2016), pp. 145–153.

[48] Noci, J. D., Domingo, D., Masip, P., Micó, J., and Ruiz, C. Comments in News, Democracy booster or Journalistic Nightmare: Assessing the Quality and Dynamics of Citizen Debates in Catalan Online Newspapers. *International Symposium on Online Journalism 2*, 1 (2012), 46–64.

[49] Oquab, M., Bottou, L., Laptev, I., and Sivic, J. Learning and Transferring Mid-Level Image Representations using Convolutional Neural Networks. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2014), pp. 1717–1724.

[50] Pan, S. J., and Yang, Q. A Survey on Transfer Learning. *IEEE Transactions on Knowledge and Data Engineering 22*, 10 (2010), 1345–1359.

[51] Park, D., Sachar, S., Diakopoulos, N., and Elmqvist, N. Supporting Comment Moderators in Identifying High Quality Online News Comments. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems* (2016), pp. 1114–1125.

[52] Peters, M., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., and Zettlemoyer, L. Deep Contextualized Word Representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)* (2018), pp. 2227–2237.

[53] Peters, M., Neumann, M., Zettlemoyer, L., and Yih, W.-t. Dissecting Contextual Word Embeddings: Architecture and Representation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing* (2018), pp. 1499–1509.

[54] Qin, L., Liu, L., Bi, W., Wang, Y., Liu, X., Hu, Z., Zhao, H., and Shi, S. Automatic Article Commenting: the Task and Dataset. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)* (2018), pp. 151–156.

[55] Radford, A., Narasimhan, K., Salimans, T., and Sutskever, I. Improving Language Understanding by Generative Pre-Training. *pre-print* (2018).

[56] Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., and Sutskever, I. Language Models are Unsupervised Multitask Learners. *pre-print* (2019).

[57] Risch, J., and Krestel, R. Delete or not Delete? Semi-Automatic Comment Moderation for the Newsroom. In *Proceedings of the First Workshop on Trolling, Aggression and Cyberbullying (TRAC-2018)* (2018), pp. 166–176.

[58] Rizos, G., Papadopoulos, S., and Kompatsiaris, Y. Predicting News Popularity by Mining Online Discussions. In *Proceedings of the 25th International Conference Companion on World Wide Web* (2016), pp. 737–742.

[59] Rother, K., and Rettberg, A. ULMFiT at GermEval-2018: A Deep Neural Language Model for the Classification of Hate Speech in German Tweets. In *Proceedings of GermEval 2018 (co-located with KONVENS)* (2018), pp. 113–119.

[60] Rücklé, A., Eger, S., Peyrard, M., and Gurevych, I. Concatenated Power Mean Embeddings as Universal Cross-Lingual Sentence Representations. *arXiv pre-print* (2018).

[61] Salton, G., and McGill, M. J. *Introduction to Modern Information Retrieval.* McGraw-Hill, Inc., 1986.

[62] Schabus, D., and Skowron, M. Academic-Industrial Perspective on the Development and Deployment of a Moderation System for a Newspaper Website. In *Proceedings of the 11th International Conference on Language Resources and Evaluation (LREC)* (2018), pp. 1602–1605.

[63] Schabus, D., Skowron, M., and Trapp, M. One million posts: A data set of german online discussions. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)* (2017), pp. 1241–1244.

[64] Schmidt, A., and Wiegand, M. A Survey on Hate Speech Detection using Natural Language Processing. In *Proceedings of the Fifth International Workshop on Natural Language Processing for Social Media* (2017), pp. 1–10.

[65] Sennrich, R., Haddow, B., and Birch, A. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)* (2016), pp. 1715–1725.

[66] Smith, L. N. Cyclical Learning Rates for Training Neural Networks. In *2017 IEEE Winter Conference on Applications of Computer Vision (WACV)* (2017), IEEE, pp. 464–472.

[67] Sokolova, M., and Lapalme, G. A Systematic Analysis of Performance Measures for Classification Tasks. *Information Processing & Management 45*, 4 (2009), 427–437.

[68] Sundermeyer, M., Schlüter, R., and Ney, H. LSTM Neural Networks for Language Modeling. In *INTERSPEECH* (2012), pp. 194–197.

[69] Szabo, G., and Huberman, B. A. Predicting the Popularity of Online Content. *Communications of the ACM 53*, 8 (2010), 80–88.

[70] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L. u., and Polosukhin, I. Attention is All you Need. In *Advances in Neural Information Processing Systems 30*. 2017, pp. 5998–6008.

[71] Witten, I. H., and Frank, E. *Data Mining: Practical Machine Learning Tools and Techniques*. Morgan Kaufmann, 2011.

[72] Yosinski, J., Clune, J., Bengio, Y., and Lipson, H. How Transferable Are Features in Deep Neural Networks? In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2* (2014), pp. 3320–3328.

[73] Zayats, V., and Ostendorf, M. Conversation Modeling on Reddit Using a Graph-Structured LSTM. *Transactions of the Association for Computational Linguistics 6* (2018), 121–132.

[74] Zhang, A. X., Culbertson, B., and Paritosh, P. Characterizing Online Discussion Using Coarse Discourse Sequences. In *Proceedings of the 11th International AAAI Conference on Weblogs and Social Media* (2017), pp. 357–367.

[75] Zheng, L., Noroozi, V., and Yu, P. S. Joint Deep Modeling of Users and Items Using Reviews for Recommendation. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining* (2017), pp. 425–434.

[76] Zook, M., Barocas, S., danah boyd, Crawford, K., Keller, E., Gangadharan, S. P., Goodman, A., Hollander, R., König, B., Metcalf, J., Narayanan, A., Nelson, A., and Pasquale, F. Ten simple rules for responsible big data research. *PLoS Computational Biology 13*, 3 (2017), 1–10.

[77] Zubiaga, A., Kochkina, E., Liakata, M., Procter, R., and Lukasik, M. Stance Classification in Rumours as a Sequential Task Exploiting the Tree Structure of Social Media Conversations. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers* (2016), pp. 2438–2448.

[78] Zubiaga, A., Kochkina, E., Liakata, M., Procter, R., Lukasik, M., Bontcheva, K., Cohn, T., and Augenstein, I. Discourse-Aware Rumour Stance Classification in Social Media Using Sequential Classifiers. *Information Processing & Management 54*, 2 (2018), 273–290.

A. Complete Experiment Results on YNACC

Table A.1.: Naive Bayes as baseline on YNACC without reply feature.

	Validation			Test		
Category	$F1_{micro}$	$F1_{macro}$	Kappa	$F1_{micro}$	$F1_{macro}$	Kappa
Persuasive	0.814 751	0.544 689	0.097 834	0.824 593	0.615 049	0.240 180
Audience	0.686 106	0.586 626	0.207 121	0.719 711	0.602 820	0.218 250
Agreement	0.885 077	0.497 575	0.046 944	0.851 718	0.504 090	0.070 055
Informative	0.818 182	0.558 838	0.136 996	0.811 935	0.534 240	0.072 118
Mean	0.813 036	0.640 313	0.296 432	0.835 443	0.651 032	0.302 184
Controversial	0.663 808	0.639 633	0.286 325	0.603 978	0.603 895	0.245 143
Disagreement	0.634 648	0.626 521	0.254 677	0.640 145	0.638 057	0.289 263
Off-topic	0.682 676	0.543 384	0.127 493	0.694 394	0.559 525	0.138 383
Sentiment	0.548 885	0.248 516	0.099 765	0.594 937	0.270 978	0.097 814

Table A.2.: Naive Bayes as baseline on YNACC with reply feature.

	Validation			Test		
Category	$F1_{micro}$	$F1_{macro}$	Kappa	$F1_{micro}$	$F1_{macro}$	Kappa
Persuasive	0.835 334	0.578 757	0.168 553	0.831 826	0.626 539	0.263 901
Audience	0.783 877	0.728 657	0.470 466	0.835 443	0.783 937	0.569 429
Agreement	0.885 077	0.497 575	0.046 944	0.851 718	0.504 090	0.070 055
Informative	0.826 758	0.553 758	0.135 205	0.822 785	0.541 971	0.089 972
Mean	0.811 321	0.641 908	0.297 929	0.835 443	0.654 870	0.309 783
Controversial	0.691 252	0.661 440	0.325 396	0.607 595	0.606 437	0.263 648
Disagreement	0.660 377	0.654 662	0.312 295	0.661 844	0.661 206	0.329 549
Off-topic	0.704 974	0.601 573	0.222 908	0.685 353	0.581 978	0.169 168
Sentiment	0.554 031	0.256 410	0.107 802	0.598 553	0.255 306	0.091 430

Table A.3.: Ridge Regression as baseline on YNACC without reply feature.

	Validation			Test		
Category	$F1_{micro}$	$F1_{macro}$	Kappa	$F1_{micro}$	$F1_{macro}$	Kappa
Persuasive	0.855 918	0.645 869	0.298 315	0.840 868	0.648 706	0.307 096
Audience	0.698 113	0.594 217	0.228 105	0.743 219	0.635 116	0.282 238
Agreement	0.879 931	0.494 976	0.035 817	0.837 251	0.476 810	0.010 576
Informative	0.818 182	0.528 404	0.087 152	0.831 826	0.510 689	0.038 297
Mean	0.813 036	0.630 087	0.280 125	0.824 593	0.610 476	0.221 906
Controversial	0.641 509	0.574 058	0.149 974	0.535 262	0.524 814	0.158 784
Disagreement	0.646 655	0.639 345	0.280 635	0.594 937	0.591 989	0.200 826
Off-topic	0.663 808	0.539 391	0.105 450	0.645 570	0.501 893	0.020 000
Sentiment	0.533 448	0.290 118	0.102 605	0.593 128	0.299 563	0.127 993

Table A.4.: Ridge Regression as baseline on YNACC with reply feature.

	Validation			Test		
Category	$F1_{micro}$	$F1_{macro}$	Kappa	$F1_{micro}$	$F1_{macro}$	Kappa
Persuasive	0.876 501	0.655 369	0.326 855	0.849 910	0.666 696	0.343 052
Audience	0.809 605	0.753 573	0.524 194	0.877 034	0.829 945	0.663 691
Agreement	0.885 077	0.555 544	0.140 667	0.837 251	0.476 810	0.010 576
Informative	0.821 612	0.494 430	0.037 679	0.824 593	0.514 319	0.040 532
Mean	0.813 036	0.619 098	0.263 044	0.824 593	0.610 476	0.221 906
Controversial	0.703 259	0.590 425	0.212 205	0.518 987	0.498 028	0.147 317
Disagreement	0.701 544	0.687 640	0.375 473	0.636 528	0.633 996	0.282 731
Off-topic	0.668 954	0.625 791	0.252 480	0.643 761	0.536 526	0.076 356
Sentiment	0.536 878	0.303 776	0.133 629	0.602 170	0.322 052	0.146 048

Table A.5.: Fast Text Classifier as baseline on YNACC without reply feature.

	Validation			Test		
Category	$F1_{micro}$	$F1_{macro}$	Kappa	$F1_{micro}$	$F1_{macro}$	Kappa
Persuasive	0.871 355	0.681 743	0.369 766	0.835 443	0.620 928	0.256 424
Audience	0.741 824	0.677 795	0.369 063	0.715 064	0.617 762	0.239 639
Agreement	0.902 230	0.673 389	0.357 396	0.855 335	0.552 154	0.146 638
Informative	0.816 467	0.614 468	0.233 489	0.792 043	0.520 952	0.042 777
Mean	0.826 758	0.666 712	0.348 070	0.842 676	0.666 371	0.332 857
Controversial	0.687 822	0.644 750	0.289 539	0.585 895	0.585 114	0.219 624
Disagreement	0.703 259	0.682 818	0.368 071	0.609 403	0.602 476	0.233 610
Off-topic	0.687 822	0.608 060	0.222 199	0.616 637	0.530 380	0.060 759
Sentiment	0.564 014	0.381 621	0.228 002	0.574 545	0.369 061	0.200 720

Table A.6.: Fast Text Classifier as baseline on YNACC with reply feature.

	Validation			Test		
Category	$F1_{micro}$	$F1_{macro}$	Kappa	$F1_{micro}$	$F1_{macro}$	Kappa
Persuasive	0.874 786	0.694 227	0.393 790	0.828 210	0.604 265	0.223 739
Audience	0.876 588	0.829 753	0.662 748	0.871 143	0.824 526	0.651 679
Agreement	0.905 660	0.672 221	0.357 737	0.858 951	0.563 350	0.167 972
Informative	0.821 612	0.623 310	0.251 383	0.808 318	0.531 745	0.066 473
Mean	0.823 328	0.674 801	0.359 347	0.831 826	0.658 445	0.316 930
Controversial	0.704 974	0.663 350	0.326 708	0.594 937	0.594 107	0.237 204
Disagreement	0.737 564	0.724 273	0.448 930	0.669 078	0.667 158	0.346 407
Off-topic	0.680 961	0.609 778	0.222 421	0.638 336	0.544 496	0.089 727
Sentiment	0.577 855	0.358 970	0.232 011	0.596 364	0.378 841	0.223 737

Table A.7.: ULMFIT on YNACC without reply feature.

	Validation			Test		
Category	$F1_{micro}$	$F1_{macro}$	Kappa	$F1_{micro}$	$F1_{macro}$	Kappa
Persuasive	0.830 189	0.684 732	0.370 458	0.795 660	0.623 408	0.247 184
Audience	0.834 768	0.790 530	0.591 875	0.882 033	0.839 355	0.681 114
Agreement	0.912 521	0.696 059	0.404 447	0.867 993	0.609 240	0.249 186
Informative	0.847 341	0.679 324	0.362 434	0.808 318	0.525 283	0.054 274
Mean	0.840 480	0.752 181	0.504 474	0.801 085	0.690 035	0.393 532
Controversial	0.706 690	0.672 649	0.346 074	0.605 787	0.600 603	0.277 090
Disagreement	0.778 731	0.772 825	0.546 145	0.690 778	0.690 373	0.386 389
Off-topic	0.706 690	0.644 402	0.290 648	0.678 119	0.596 264	0.193 005
Sentiment	0.628 028	0.448 142	0.322 735	0.652 727	0.436 832	0.302 938

Table A.8.: ULMFIT on YNACC with reply feature.

	Validation			Test		
Category	$F1_{micro}$	$F1_{macro}$	Kappa	$F1_{micro}$	$F1_{macro}$	Kappa
Persuasive	0.830 189	0.684 732	0.370 458	0.795 660	0.623 408	0.247 184
Audience	0.834 768	0.790 530	0.591 875	0.882 033	0.839 355	0.681 114
Agreement	0.897 084	0.698 864	0.401 212	0.867 993	0.666 121	0.343 241
Informative	0.842 196	0.670 233	0.344 095	0.824 593	0.514 319	0.040 532
Mean	0.833 619	0.744 752	0.489 524	0.793 852	0.672 066	0.356 189
Controversial	0.680 961	0.665 067	0.343 118	0.598 553	0.597 594	0.245 084
Disagreement	0.778 731	0.772 825	0.546 145	0.690 778	0.690 373	0.386 389
Off-topic	0.698 113	0.643 038	0.286 434	0.661 844	0.599 624	0.200 367
Sentiment	0.628 028	0.448 142	0.322 735	0.652 727	0.436 832	0.302 938

Table A.9.: BERT on YNACC without reply feature.

	Validation			Test		
Category	$F1_{micro}$	$F1_{macro}$	Kappa	$F1_{micro}$	$F1_{macro}$	Kappa
Persuasive	0.864 494	0.721 146	0.442 375	0.837 251	0.674 553	0.351 970
Audience	0.795 181	0.765 730	0.532 298	0.785 844	0.735 182	0.470 374
Agreement	0.903 945	0.709 434	0.423 486	0.867 993	0.656 297	0.326 364
Informative	0.862 779	0.748 490	0.496 991	0.788 427	0.570 301	0.141 293
Mean	0.849 057	0.740 574	0.484 527	0.811 935	0.680 105	0.366 427
Controversial	0.735 849	0.700 215	0.400 502	0.598 553	0.594 543	0.259 741
Disagreement	0.739 280	0.729 836	0.459 698	0.667 269	0.664 636	0.343 815
Off-topic	0.737 564	0.663 162	0.334 490	0.683 544	0.593 739	0.189 279
Sentiment	0.633 218	0.471 124	0.377 149	0.609 091	0.446 530	0.313 745

Table A.10.: BERT on YNACC with reply feature.

	Validation			Test		
Category	$F1_{micro}$	$F1_{macro}$	Kappa	$F1_{micro}$	$F1_{macro}$	Kappa
Persuasive	0.862 779	0.721 825	0.443 662	0.835 443	0.672 609	0.347 869
Audience	0.831 325	0.803 391	0.608 534	0.834 846	0.796 176	0.592 366
Agreement	0.903 945	0.727 770	0.457 728	0.858 951	0.664 504	0.335 705
Informative	0.845 626	0.717 051	0.434 115	0.790 235	0.588 900	0.179 504
Mean	0.835 334	0.723 270	0.448 993	0.815 552	0.688 709	0.383 999
Controversial	0.758 148	0.722 872	0.445 746	0.602 170	0.596 597	0.271 462
Disagreement	0.777 015	0.770 116	0.540 433	0.694 394	0.693 492	0.394 861
Off-topic	0.740 995	0.676 693	0.357 747	0.685 353	0.593 382	0.189 081
Sentiment	0.638 408	0.450 552	0.374 485	0.625 455	0.448 756	0.330 252

Table A.11.: Prepend Previous: TX_1

	Validation			Test		
Category	$F1_{micro}$	$F1_{macro}$	Kappa	$F1_{micro}$	$F1_{macro}$	Kappa
Persuasive	0.847 341	0.703 593	0.407 351	0.808 318	0.628 021	0.257 925
Audience	0.836 489	0.796 326	0.600 936	0.878 403	0.842 171	0.684 936
Agreement	0.921 098	0.761 321	0.526 435	0.877 034	0.677 451	0.368 446
Informative	0.847 341	0.695 042	0.391 662	0.797 468	0.577 540	0.155 288
Mean	0.814 751	0.726 633	0.453 504	0.788 427	0.675 738	0.367 456
Controversial	0.728 988	0.679 546	0.360 168	0.587 703	0.579 978	0.250 134
Disagreement	0.794 168	0.786 134	0.572 267	0.716 094	0.715 256	0.437 830
Off-topic	0.754 717	0.722 000	0.444 536	0.638 336	0.592 760	0.193 548
Sentiment	0.647 059	0.408 630	0.361 416	0.645 455	0.440 146	0.333 992

Table A.12.: Prepend Previous: TX_2

	Validation			Test		
Category	$F1_{micro}$	$F1_{macro}$	Kappa	$F1_{micro}$	$F1_{macro}$	Kappa
Persuasive	0.845 626	0.674 018	0.348 442	0.822 785	0.645 624	0.294 367
Audience	0.827 883	0.777 251	0.569 100	0.880 218	0.835 621	0.674 093
Agreement	0.912 521	0.746 100	0.494 895	0.855 335	0.631 481	0.275 728
Informative	0.843 911	0.672 118	0.348 107	0.828 210	0.588 687	0.179 230
Mean	0.840 480	0.724 784	0.453 366	0.828 210	0.699 396	0.402 977
Controversial	0.725 557	0.672 315	0.346 431	0.560 579	0.549 162	0.207 844
Disagreement	0.782 161	0.773 814	0.547 629	0.717 902	0.716 492	0.442 483
Off-topic	0.763 293	0.712 525	0.426 655	0.640 145	0.587 228	0.179 140
Sentiment	0.615 917	0.420 182	0.321 711	0.645 455	0.440 688	0.314 688

Table A.13.: Prepend Previous: HL_1

	Validation			Test		
Category	$F1_{micro}$	$F1_{macro}$	Kappa	$F1_{micro}$	$F1_{macro}$	Kappa
Persuasive	0.847 341	0.688 988	0.378 010	0.815 552	0.645 513	0.292 466
Audience	0.839 931	0.794 148	0.600 888	0.894 737	0.854 020	0.711 080
Agreement	0.921 098	0.761 321	0.526 435	0.862 568	0.634 030	0.284 898
Informative	0.859 348	0.693 252	0.392 137	0.819 168	0.564 155	0.130 530
Mean	0.845 626	0.749 628	0.500 210	0.810 127	0.690 275	0.389 695
Controversial	0.727 273	0.684 804	0.369 713	0.575 045	0.566 864	0.227 640
Disagreement	0.792 453	0.783 897	0.567 807	0.701 627	0.699 549	0.411 197
Off-topic	0.758 148	0.702 578	0.407 781	0.614 828	0.559 491	0.124 486
Sentiment	0.603 806	0.409 586	0.318 831	0.623 636	0.427 908	0.314 908

Table A.14.: Prepend Previous: HL_2

	Validation			Test		
Category	$F1_{micro}$	$F1_{macro}$	Kappa	$F1_{micro}$	$F1_{macro}$	Kappa
Persuasive	0.871 355	0.720 973	0.442 745	0.817 360	0.657 186	0.315 082
Audience	0.826 162	0.797 065	0.596 042	0.816 697	0.781 933	0.564 768
Agreement	0.915 952	0.763 645	0.529 009	0.857 143	0.662 411	0.331 154
Informative	0.845 626	0.690 113	0.381 993	0.793 852	0.579 114	0.158 789
Mean	0.838 765	0.738 500	0.477 997	0.817 360	0.706 360	0.422 583
Controversial	0.728 988	0.686 329	0.372 794	0.584 087	0.580 519	0.230 932
Disagreement	0.785 592	0.777 068	0.554 137	0.710 669	0.709 070	0.428 431
Off-topic	0.770 154	0.740 527	0.481 743	0.629 295	0.579 677	0.166 182
Sentiment	0.610 727	0.411 240	0.328 518	0.620 000	0.433 459	0.303 249

Table A.15.: Prepend Previous: ART

	Validation			Test		
Category	$F1_{micro}$	$F1_{macro}$	Kappa	$F1_{micro}$	$F1_{macro}$	Kappa
Persuasive	0.811 321	0.651 210	0.303 646	0.801 085	0.656 188	0.312 516
Audience	0.833 046	0.785 294	0.583 722	0.889 292	0.848 466	0.699 438
Agreement	0.909 091	0.731 810	0.466 992	0.857 143	0.616 663	0.251 768
Informative	0.826 758	0.666 712	0.334 061	0.806 510	0.594 188	0.188 516
Mean	0.826 758	0.727 375	0.455 054	0.786 618	0.671 893	0.359 564
Controversial	0.739 280	0.689 720	0.380 905	0.529 837	0.506 196	0.170 972
Disagreement	0.763 293	0.756 554	0.513 485	0.710 669	0.710 668	0.423 267
Off-topic	0.753 002	0.711 553	0.423 137	0.578 662	0.554 810	0.144 730
Sentiment	0.608 997	0.421 227	0.324 075	0.594 545	0.447 881	0.322 712

Eigenständigkeitserklärung

Ich versichere hiermit, die vorliegende Arbeit selbst verfasst, Zitate gekennzeichnet und keine anderen als die offengelegten Quellen und Hilfsmittel benutzt zu haben.

Berlin, 15. April 2019 Johannes Filter