Hasso Plattner Institute, University of Potsdam
Information Systems Group

Reader Comment Analysis on Online News Platforms

A dissertation submitted in partial fulfillment of
the requirements for the degree of

Dr. rer. nat.

Computer Science, Information Systems

to the Digital Engineering Faculty at the
Hasso Plattner Institute, University of Potsdam

by
Julian Risch

Potsdam, October 2020

Reviewers

Prof. Dr. Felix Naumann
Hasso Plattner Institute, University of Potsdam

Prof. Dr. Michael Granitzer
University of Passau

Prof. Vivien Petras, PhD
Humboldt University Berlin

Published online on the
Publication Server of the University of Potsdam:
https://doi.org/10.25932/publishup-48922
https://nbn-resolving.org/urn:nbn:de:kobv:517-opus4-489222

Abstract

Comment sections of online news platforms are an essential space to express opinions and discuss political topics. However, the misuse by spammers, haters, and trolls raises doubts about whether the benefits justify the costs of the time-consuming content moderation. As a consequence, many platforms limited or even shut down comment sections completely. In this thesis, we present deep learning approaches for comment classification, recommendation, and prediction to foster respectful and engaging online discussions. The main focus is on two kinds of comments: toxic comments, which make readers leave a discussion, and engaging comments, which make readers join a discussion. First, we discourage and remove toxic comments, e.g., insults or threats. To this end, we present a semi-automatic comment moderation process, which is based on fine-grained text classification models and supports moderators. Our experiments demonstrate that data augmentation, transfer learning, and ensemble learning allow training robust classifiers even on small datasets. To establish trust in the machine-learned models, we reveal which input features are decisive for their output with attribution-based explanation methods. Second, we encourage and highlight engaging comments, e.g., serious questions or factual statements. We automatically identify the most engaging comments, so that readers need not scroll through thousands of comments to find them. The model training process builds on upvotes and replies as a measure of reader engagement. We also identify comments that address the article authors or are otherwise relevant to them to support interactions between journalists and their readership. Taking into account the readers' interests, we further provide personalized recommendations of discussions that align with their favored topics or involve frequent co-commenters. Our models outperform multiple baselines and recent related work in experiments on comment datasets from different platforms.

Zusammenfassung

Kommentarspalten von Online-Nachrichtenplattformen sind ein essentieller Ort, um Meinungen zu äußern und politische Themen zu diskutieren. Der Missbrauch durch Trolle und Verbreiter von Hass und Spam lässt jedoch Zweifel aufkommen, ob der Nutzen die Kosten der zeitaufwendigen Kommentarmoderation rechtfertigt. Als Konsequenz daraus haben viele Plattformen ihre Kommentarspalten eingeschränkt oder sogar ganz abgeschaltet. In dieser Arbeit stellen wir Deep-Learning-Verfahren zur Klassifizierung, Empfehlung und Vorhersage von Kommentaren vor, um respektvolle und anregende Online-Diskussionen zu fördern. Das Hauptaugenmerk liegt dabei auf zwei Arten von Kommentaren: toxische Kommentare, die die Leser veranlassen, eine Diskussion zu verlassen, und anregende Kommentare, die die Leser veranlassen, sich an einer Diskussion zu beteiligen. Im ersten Schritt identifizieren und entfernen wir toxische Kommentare, z.B. Beleidigungen oder Drohungen. Zu diesem Zweck stellen wir einen halbautomatischen Moderationsprozess vor, der auf feingranularen Textklassifikationsmodellen basiert und Moderatoren unterstützt. Unsere Experimente zeigen, dass Datenanreicherung, Transfer- und Ensemble-Lernen das Trainieren robuster Klassifikatoren selbst auf kleinen Datensätzen ermöglichen. Um Vertrauen in die maschinell gelernten Modelle zu schaffen, zeigen wir mit attributionsbasierten Erklärungsmethoden auf, welche Teile der Eingabe für ihre Ausgabe entscheidend sind. Im zweiten Schritt ermutigen und markieren wir anregende Kommentare, z.B. ernsthafte Fragen oder sachliche Aussagen. Wir identifizieren automatisch die anregendsten Kommentare, so dass die Leser nicht durch Tausende von Kommentaren blättern müssen, um sie zu finden. Der Trainingsprozess der Modelle baut auf Upvotes und Kommentarantworten als Maß für die Aktivität der Leser auf. Wir identifizieren außerdem Kommentare, die sich an die Artikelautoren richten oder anderweitig für sie relevant sind, um die Interaktion zwischen Journalisten und ihrer Leserschaft zu unterstützen. Unter Berücksichtigung der Interessen der Leser bieten wir darüber hinaus personalisierte Diskussionsempfehlungen an, die sich an den von ihnen bevorzugten Themen oder häufigen Diskussionspartnern orientieren. In Experimenten mit Kommentardatensätzen von verschiedenen Plattformen übertreffen unsere Modelle mehrere grundlegende Vergleichsverfahren und aktuelle verwandte Arbeiten.

Acknowledgments

I would like to express my deepest appreciation to Felix Naumann and Ralf Krestel, who aroused my early interest in academic research through the *Beautiful Data* seminar in 2013 and the *Recommender Systems* seminar in 2014. Since then, they supported me throughout my Master's studies and finally my Ph.D. studies with valuable advice and constructive criticism. The Information Systems group is an ideal research environment and I would like to extend my gratitude to all my colleagues within this group.

Further, I am grateful to my mentor Patrick Baudisch for his insights into planning a career in academia. Starting with my Bachelor's project in 2013, he provided me with guidance and encouragement. The HPI Research School supported me not only by funding my position but also with the chance to present and discuss intermediate results of my research. I am most thankful for the opportunity to exchange ideas with researchers in Cape Town, Nanjing, and New York, and I very much enjoyed co-organizing the annual symposiums. Special thanks to Andreas Polze and Robert Hirschfeld but also to all the other professors and my fellow Ph.D. students for their feedback.

My gratitude also goes to the students at HPI with whom I had the pleasure to work with. First and foremost, I would like to recognize the help that I received from the student assistants Nicolas Alder, Samuele Garda, Lasse Kohlmeyer, Niklas Köhnicke, and Olena Vyshnevska. By co-teaching seminars and lectures, I worked with many more bright students, in particular, the Master's project students Carl Ambroselli, Yannick Bäumer, Andreas Burmeister, Jan Ladleif and Tim Naumann, and the Master's thesis students Carl Ambroselli, Johannes Filter, Samuele Garda, Philipp Hager, and Victor Künstler.

Many of my research projects would not have been possible and not half as much fun without the help from my collaborators from other universities: Betty van Aken and Alexander Löser from Beuth University of Applied Sciences, Anke Stoll and Marc Ziegele from Heinrich Heine University Düsseldorf, and Renukswamy Chikkamath and Robin Ruff from University of Passau. Last but not least, I would like to mention the support and domain knowledge from industry partners and their insights into journalism, aerospace engineering, the patent domain, and large-scale application scenarios for machine learning. I gratefully acknowledge the assistance of Andreas Loos (Zeit Online), Sarah Lück (Axel Springer), Klaus Paul (Rolls Royce), Christoph Hewel (Betten & Resch), Wolfgang Richter (Journalistenbüro Schnittstelle), and Timo Möller (deepset).

Preface

This thesis is based on multiple previously published peer-reviewed publications, which are listed in the following. All these publications were written by Risch under the supervision of Krestel, and we point out contributions by additional co-authors where applicable.

Chapter 3, in particular Section 3.3, is based on a journal article [153]. The contributions can be assigned as follows. Risch and Krestel drafted the concept, whereupon Risch designed and implemented the process and conducted the experiments. A group of ten students implemented the web crawlers and helped with the first phase of the data collection process under the supervision of Risch.

Chapter 4 is based on five publications. A combined overview that puts them into the context of related work has been published as a book chapter [154]. The publication underlying Section 4.1 is a joint work by van Aken, Risch, Löser, and Krestel [194]. Van Aken and Risch implemented and trained the models. The correlation analysis, model selection, and ensemble learning were done by Risch, and the error analysis was done by van Aken. Löser and Krestel supervised the work and revised the manuscript written by van Aken and Risch. Section 4.2 is published in a paper authored by Risch and Krestel, where Risch designed, implemented, and evaluated the approach [148]. The shared task submission that provides the basis for Section 4.3 was also designed, implemented, and evaluated by Risch, and the manuscript was written by Risch and revised by Krestel [155]. Section 4.4 combines the content of a journal article [162] and a paper [163] by Risch, Ruff, and Krestel. The author's roles can be assigned as follows: Ruff implemented the models based on related work selected by Risch and conducted the experiments under the supervision of Risch and Krestel in context of his Bachelor's thesis [170]. Risch wrote the journal article and paper, which were revised by Krestel. The contributions in Section 4.5 originate from a collaborative research project. Under the supervision of Risch and Krestel, Ambroselli, Bäumer, Burmeister, Ladleif, and Naumann implemented and evaluated multiple models. Loos from Zeit Online served as an industry expert and provided access to the dataset. The publication was written by Risch and revised by Krestel [150].

Chapter 5 is based on four publications. Section 5.1 builds on the Master's thesis by Ambroselli [6], which was co-supervised by Risch and Krestel, and continued in a subsequent paper [7]. Loos again provided advisory support to the project. The four authors conceptualized the idea and selected appropriate features in a joint effort. Ambroselli implemented the approach and conducted the experiments, which were designed by Risch and Krestel. Risch wrote the paper, which was revised by Krestel. Section 5.2 is based on the Master's thesis by Künstler [102], which was co-supervised by Risch and Krestel, and a subsequent paper, which is currently under review [161]. Risch selected the neural network architecture, which was implemented and adapted by Künstler under the close supervision of Risch. Künstler conducted the experiments, which were jointly designed by Künstler, Risch, and Krestel. The manuscript was written by Risch and revised by Krestel. Section 5.3 is published in a paper by Risch and Krestel [156]. Risch designed the taxonomy and designed, implemented, and evaluated the approach. The initial idea for the bias removal method was developed in collaboration with Filter, Hagmeister, and Kellermeier. A dataset paper by Risch and Krestel [157] is the basis of Section 5.4. Loos and Richter provided advice as domain experts throughout the project. Kohlmeyer and Köhnecke supported the data collection and enrichment under the close supervision of Risch. Risch, Alder, and Krestel annotated the data.

During my Ph.D. studies, I also worked on related research projects not discussed in this thesis. They include work on document representations that capture semantic similarities of texts despite different language use. My collaborators and I developed novel topic models, dense vector representations, and neural networks for texts from multiple different domains, such as patents, scientific papers, book synopses, technical documentation, or medical reports. Our respective publications in the area of digital libraries [147, 149, 158] and patent document classification [151, 152, 160] deal with large corpora and mine information from the documents to compare them. Therefore, they fall into the research field of comparative text mining. The connection to the work in this thesis is the common goal to develop novel document representations for a variety of applications, such as text classification or recommendation.

Contents

1

Introduction

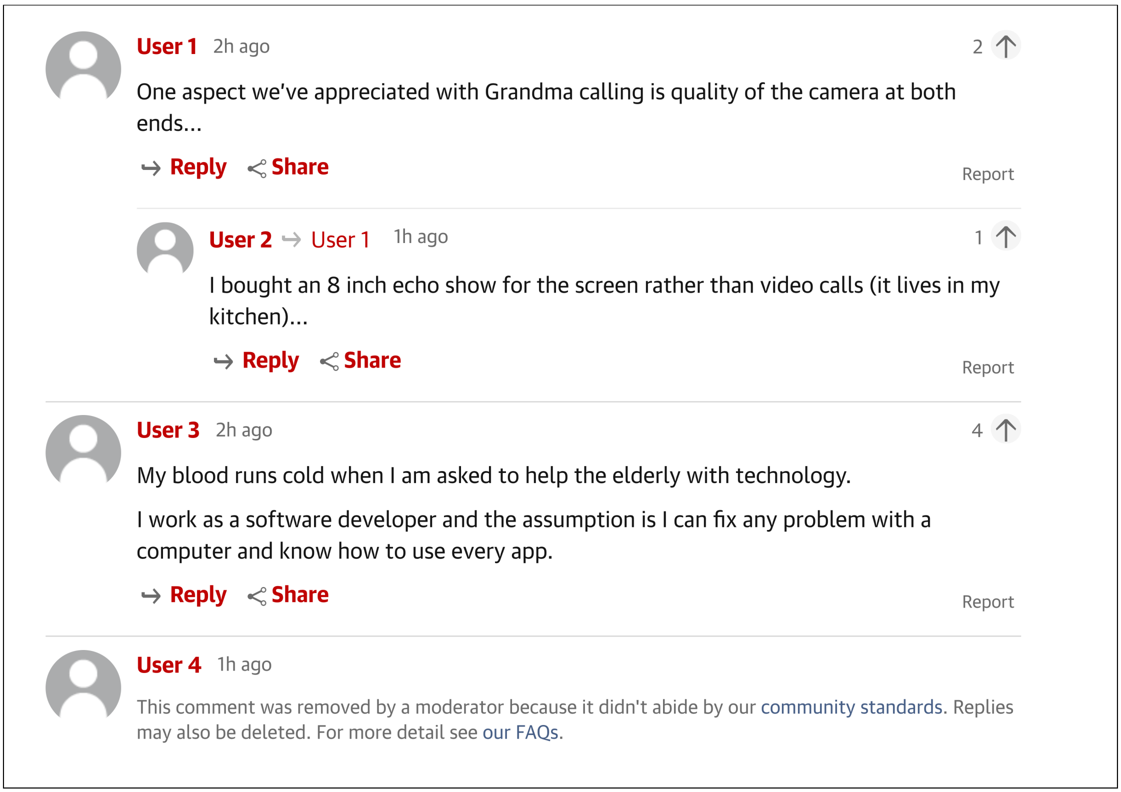
Social media platforms, such as Facebook, YouTube, Twitter, and Instagram, enable millions of users to share content publicly. Regardless of the content types, such as texts, photos, videos, and events, a crucial point of these platforms is that users can discuss content. The media business and journalists adapted to this development by introducing comment sections on their online news platforms. However, these comment sections brought up several industry-wide challenges.

1.1 Reader Discussions on Online News Platforms

Thirty years ago, newspapers received hand-written letters to the editor and selected maybe a handful for publication. This procedure was called reader engagement and was the only way for readers to interact with other readers or the newspaper via public discussion. With the rise of the World Wide Web, the establishment of online news platforms, and the appearance of comment sections, the situation has changed drastically. Nowadays, irrespective of who the readers are and what they think, they can freely share their opinion on news topics with a broad audience — if there is an open comment section. The motivation for platform providers to operate comment sections is the prospect of higher user loyalty, higher retention rates, and increased website traffic, which results in increased advertising revenue and subscription revenue.

Figure 1.1 shows the basic commenting features of an exemplary comment section. The reader discussion is shown below the text of the news article, where readers can post comments, cast upvotes, and reply to comments by others. The number of received upvotes and the publication time of each comment are displayed next to it. In contrast to social networks, readers cannot connect with or follow others, and there is no explicit social network graph. User profiles consist only of a user name and an optional profile image and are of minor significance on the news platforms. However, as on the websites of large social networks, there is an option to report a comment to the moderation team.

On online news platforms, moderators ensure compliance with the discussion rules. Depending on the platform, they read only those comments that are reported by readers, are posted by new users, discuss articles on particularly controversial topics, or — if time permits — all the comments. If they find a violation of the rules, they remove the



User 1 2h ago 2

One aspect we've appreciated with Grandma calling is quality of the camera at both ends...

User 2 → User 1 1h ago 1

I bought an 8 inch echo show for the screen rather than video calls (it lives in my kitchen)...

User 3 2h ago 4

My blood runs cold when I am asked to help the elderly with technology.

I work as a software developer and the assumption is I can fix any problem with a computer and know how to use every app.

User 4 1h ago

This comment was removed by a moderator because it didn't abide by our community standards. Replies may also be deleted. For more detail see our FAQs.

Figure 1.1: In comment sections, readers can post a comment, cast an upvote, and reply to comments by others (example from THE GUARDIAN).

comment in whole or in part. A note with a reference to the community standards is then inserted, as seen at the bottom of Figure 1.1. Repeated violations by the same user can result in a temporary or permanent ban. As a last resort, the reader discussion can be shut down completely. However, the opportunity to articulate opinions and ideas online is a valuable good. It is part of the freedom of expression, which is a declared universal human right:

> "You have the freedom to express yourself online and to access information and the opinions and expressions of others."
>
> Article 19 of the Universal Declaration of Human Rights

> "Congress shall make no law [. . .] abridging the freedom of speech, or of the press [. . .]"
>
> First Amendment of the United States Bill of Rights

> "Every person shall have the right freely to express and disseminate their opinions [. . .] and to inform themselves [. . .]"
>
> Article 5 of the Basic Law for the Federal Republic of Germany

With the media referred to as the fourth pillar of democracy, it ensures transparency of political processes and the three other pillars: Judiciary, Executive, and Legislature.[1] The media informs citizens of processes in their country and fosters a system of checks and balances. With more and more political campaigning or even agitation being distributed over the Internet, having serious and safe platforms to discuss political topics and news in general is increasingly critical. Posting comments in online discussions has become an important way to exercise one's right to freedom of expression on the Web.

Many readers take advantage of this opportunity, and reader discussions thereby enrich the platforms' content. For instance, a study found out that 78 percent of US Americans read comments on news, and 55 percent write them [185]. 19 percent of the commenters even spend more time with the comments than with the article. Diakopoulos and Naaman [54] analyzed the reasons why people read or write online news comments. Learning about other readers' views is the primary motivation for reading comments. On the writer's side, the strongest motive is expressing opinions and emotions, followed by the desire to provide information to others, such as answering questions, sharing experiences, or correcting errors. In rare cases, the motivation is to spread misinformation with the intent to see the reaction of the community. A survey among US American news commenters confirms these findings: the majority (56 percent) wants to express an emotion or opinion [185]. This reason is followed by wanting to add information (38 percent), to correct inaccuracies or misinformation (35 percent), or to take part in the debate (31 percent).[2]

1.2 Toxic Comments

On the flip side of all the benefits of online discussions, malicious users disrupt otherwise respectful discussions with their toxic comments. We define a toxic comment as a rude, disrespectful, or unreasonable comment that makes other users leave a discussion. It *poisons* an online discussion so that other users abandon it, and toxic comments can also cause real-life violence [83, 167].

1.2.1 Shades of Toxicity

Toxicity comes in many different forms and shapes, such as insults, threats, or identity hate. Insults contain rude or offensive statements that concern an individual or a group. In contrast to that, identity hate aims at members of groups defined by religion, sexual orientation, ethnicity, gender, or other social identifiers. Negative attributes are ascribed to them as if these attributes were universally valid. Examples of identity hate are racist, homophobic, and misogynistic comments. A common threat in online discussions is to have another user's account closed. The most severe threats announce or advocate for inflicting pain, injury, or damage on others or oneself. Spam messages and off-topic comments by trolls also meet the definition of toxicity because discussions filled with them would quickly become abandoned by users. However, the detection of spam and

[1] `www.voj.news/media-as-a-fourth-pillar-of-democracy/`

[2] Multiple reasons could be selected.

off-topic comments is neither the focus of toxic comment classification nor addressed in this thesis.

Some toxic comments might be legal expressions of opinions but still prohibited by the platform's terms of use or discussion guidelines. To exemplify reasons for comment removal, we summarize nine rules that comprise the discussion guidelines by the German news platform Zeit Online.[3] Most rules are not platform-specific but are rather part of the *Netiquette* — the etiquette on the Internet.

1. **Insults** are not allowed. Criticize the content of the article and not its author.
2. **Discrimination and defamation** are not allowed.
3. **Non-verifiable allegations and suspicions** that are not supported by any credible arguments or sources will be removed.
4. **Advertising and other commercial content** should not be part of comments.
5. **Personal data** of others may not be published.
6. **Copyright** must be respected. Never post more than short excerpts when quoting third party content.
7. **Quotations** must be labeled as such and must reference its source.
8. **Links** may be posted but may be removed if the linked content violates the rules.

1.2.2 Shut Down of Comment Sections

Comment moderation based on these rules is a time-consuming task. Moderators need to read through hundreds of comments per news article. Popular articles receive thousands of comments, with some of them even exceeding tens of thousands.[4] The number of received comments varies by topic and daytime so that the effort is unpredictable for the moderators. On some platforms, moderators work day and night shifts to cover 24/7.

News platforms around the world are overwhelmed by a large number of received comments and the high effort of moderation. As a consequence of toxic comments and the high costs of manual moderation, many platforms decided to limit operating hours or shut down their comment sections completely.[5] Although it is an industry-wide challenge, there seems to be no solution at hand. Consequently, toxic comments pose a direct

[3] www.zeit.de/administratives/2010-03/netiquette/seite-2

[4] www.theguardian.com/politics/live/2019/sep/26/boris-johnsons-brexit-rhetoric-condemned-as-mps-tell-of-death-threats-politics-live#comments

[5] www.vice.com/en_us/article/vvdjjy/were-getting-rid-of-comments-on-vice, www.theverge.com/2015/7/6/8901115/were-turning-comments-off-for-a-bit, www.libn.com/2018/02/28/newsday-shuts-down-online-comments, www.popsci.com/science/article/2013-09/why-were-shutting-our-comments/, www.abc.net.au/news/about/backstory/digital/2017-09-07/8878604, www.chicago.suntimes.com/news/2014/4/11/18580073, www.wired.com/2015/10/brief-history-of-the-demise-of-the-comments-timeline, www.gu.com/technology/2016/apr/12/the-dark-side-of-guardian-comments, www.nzz.ch/feuilleton/in-eigener-sache-warum-wir-unsere-kommentarspalte-umbauen-ld.143568, www.fuldaerzeitung.de/fulda/keine-kommentarfunktion-mehr-unserem-portal-13688577.html, www.dw.com/de/warum-wir-die-kommentarfunktion-abschalten/a-45017804, www.netzpolitik.org/2019/die-kommentare-sind-tot-lang-leben-eure-inhaltlichen-ergaenzungen

threat to the existence of comment sections. First, they lower the number of users who engage in discussions and, consequently, the number of visitors to the news platform or the time spent on it. As a result, an exchange of diverse opinions becomes impossible. With subscription models and ads as a way to earn money, a lower number of visitors means losing money. Second, legal reasons might require the platforms to deploy countermeasures against hate speech and to delete such content or not publish it at all. For example, in Germany, online social network providers are obliged by the Network Enforcement Act to check content reported by users, and "remove or block access to content that is manifestly unlawful within 24 hours of receiving the complaint".[6] While this law does not apply to "platforms offering journalistic or editorial content", it still serves as an accelerator for the development of comment moderation tools. This thesis supports news platforms in keeping their comment sections open by designing and implementing a semi-automatic comment moderation process.

1.3 Engaging Comments and Discussions

The ever-increasing number of comments not only poses a challenge for platform providers and moderators but also for readers. It is distracting and hinders engagement: no news consumer is able to read through all the comments. Overwhelmed by hundreds to thousands of comments, new users give up on joining the discussion. A current manual approach for coping with this information overload lets the editors highlight comments that are especially interesting from their point of view. These *editor's picks* are shown at the top of the reader discussion, where they are more visible to a broader audience. However, the manual effort to select these highlighted comments comes on top of the moderation of toxic comments and exceeds the platforms' resources.

1.3.1 Engaging Comments

Inspired by the idea of editor's picks, we define the concept of engaging comments. In contrast to toxic comments, which make readers leave a discussion, they make readers join a discussion. They encourage readers to actively contribute to the discussion. The engagement that a comment entails is measurable through the number of upvotes and replies it receives. Voting on a comment is a rather basic way to interact. It is faster and easier than posting a comment. Jokes or brief statements many readers agree with receive particularly large numbers of upvotes. In contrast, replying to another user's comment actually starts a conversation. Users reply to comments for different reasons. For example, they want to correct another user's error, give their personal view, or express consent or dissent. Consequently, comments that contain a serious question asking for factual information or opinions receive many replies. While the number of upvotes reveals a comment's popularity, the same does not apply to the number of replies.

The number of upvotes and replies that a comment receives depends not only on the comment text but also on several other influencing factors, such as its publication time or the article topic. It is difficult to interpret and analyze the reasons behind high or

[6] Article 1 (3) Network Enforcement Act: `germanlawarchive.iuscomp.org/?p=1245`

low numbers of upvotes and replies because of these factors. Therefore, one challenge is to mitigate the influence of these other factors and isolate the relationship between the comment text and the number of upvotes and replies. The removal of that bias would allow analyzing which comment texts trigger many reactions in form of upvotes and replies.

A special kind of comments is not engaging for the readers but for the journalists who wrote the news article. While most comments are for discussions amongst the readers, some comments address the journalists directly with a question or feedback. Supporting journalists at actively joining reader discussions could benefit both readers and journalists. Stroud et al. [185] found that the majority of readers want journalists to engage more in the comment sections. For example, 61 percent of the readers would like journalists to post comments to clarify factual questions. By joining the discussions, journalists could gain insights into their readership and, for example, learn about readers' wishes for future articles. In turn, readers might find the discussion more engaging and be encouraged to post questions to the journalists more often.

Note that the two classes of toxic and engaging comments are neither mutually exclusive nor collectively exhaustive. On the one hand, few comments fall into both classes at the same time. Such comments make some users leave the discussion while at the same time, they encourage other users to reply to the comment and express their dissent or insist on compliance with the rules. On the other hand, many comments are neither toxic nor engaging. They do not attract the readers' attention and cannot trigger other readers to leave or join the discussion.

1.3.2 Engaging Discussions

Reader engagement can not only be measured on the level of individual comments but also the level of entire discussions. To this end, we leave aside the number of upvotes and replies that a comment receives. Instead, the total number of comments in one discussion can serve as a measure of how engaging the discussion is. More interesting than the exact number is to know in advance which discussion will become most popular among all ongoing discussions on a platform. Readers looking to join a discussion might prefer one with many comments rather than with mostly inactive participants. Experienced moderators can estimate the reader engagement even before the discussion starts or at least in its early phase. However, their estimations are based on gut feeling rather than data analysis. Not only the news article topic but also a variety of other features influence the number of comments. Humans cannot take into account all the available data, which motivates the use of machine learning applications to model reader engagement.

Studying the popularity of discussions in the community as a whole neglects the individuality of the readers. Readers have different interests, and they favor different discussions to post their comments. Therefore, personalized discussion recommendations are interesting to encourage individual readers to join a particular discussion. These recommendations can not only lead to higher user retention and increased website traffic; they can also make the group of discussion participants more diverse by encouraging readers who would otherwise stay passive. A reader might join a particular discussion because of its general topic or individual comments. Another reason might be the other discussion participants: pairs of readers might often co-occur in discussions because of rivalry, friendship, or shared interests. The concept of engaging discussions thus considers more features than only the comment texts.

1.4 Task Descriptions

This thesis pursues the research question of how to foster respectful and engaging reader discussions on online news platforms with machine learning methods. To this end, we consider readers, commenters, moderators, journalists, and platform providers as stakeholders and focus on two kinds of comments: toxic comments and engaging comments. During the course of this thesis, we address and solve the following tasks centered around them.

Toxic Comment Classification. Manually reading through entire reader discussions to detect toxic comments is time-consuming and becomes infeasible with the increasing amount of comments. To support the moderators, it is necessary to automate parts of this task and implement a semi-automatic moderation process. A machine-learned model needs to automatically identify potentially toxic comments and draw the moderators' attention to them. Thereby, only a subset of the comments remains to be checked manually, and the time and money spent on moderation are drastically reduced. To increase reader acceptance, the model needs to be integrated in a way that allows moderators to continuously monitor its performance and correct any of its errors.

Explanation of Classification Results. The moderators working with the machine-learned model need to understand why a comment is automatically classified as toxic. Afterward, they need to provide reasons for their interventions to the readers to ensure a fair and transparent process. A fine-grained classification of toxic comments by the type of rule violation, e.g., posting an insult or a threat, is required. Further, explanation methods are needed that identify the toxic words or phrases within a comment. Researchers could also benefit from a solution to this task because they could rely on explanations to find weaknesses of the model and come up with improvements.

Reader Engagement Prediction. We define the task of identifying news articles that will receive a large number of comments. They are promising starting points for readers who expect vivid interaction. More importantly, the predictions support platform providers at estimating the moderation effort and planning their resources. We distinguish two different scenarios for the prediction task based on the available input features. First, the pre-publication scenario considers only those features that are available before the article is published, and the discussion is started. Second, the post-publication scenario adds the first few comments as features.

Personalized Discussion Recommendation. When readers access a news platform's website, they can choose from a variety of recent articles and ongoing discussions. One task to increase engagement is to recommend discussions to individual readers, either to reply or to simply read them. To be able to personalize these recommendations, the interests of each reader need to be modeled. A model could draw on a large amount of training data available in the form of all previous discussions as there are no manual labels required.

Engaging Comment Classification. Since online news platforms display reader comments in one long list per news article, the discussions can become unclear and difficult to follow with an increasing number of comments. Manually selected editor's picks are not an efficient solution to this problem because of the extra effort for moderators. Thus, there should be an automated approach that highlights the most engaging comments so that readers can quickly start a conversation.

Including Journalists in Reader Discussions. Although primarily meant as forums where readers discuss amongst each other, comment sections can also spark a dialog with the journalists who authored the article. A small but important fraction of comments address the journalists directly, e.g., with questions, recommendations for future topics, thanks and appreciation, or article corrections. However, the sheer number of comments makes it infeasible for journalists to closely follow discussions around their articles. An automated approach that notifies them of relevant comments is needed to encourage more interactions with their readership.

Learning from Limited Data. Deep learning on comment datasets comes with the challenge of having only a few thousand labeled samples available. This limitation makes the training of neural networks, which typically have millions of parameters, particularly difficult and leads to underfitting or overfitting the data. Underfitting means that the model learns too few patterns and, therefore, cannot provide accurate predictions. Overfitting means that the model learns too many patterns, including those that only hold for the training samples but do not generalize to other data. We must prevent both issues to train a model that generalizes well despite the limited training data.

Facilitating Dataset Reproducibility. A challenge for research on reader comments is that the data is typically restricted by copyright protection or privacy regulations, which hinder their distribution. However, accessible and reusable datasets are a necessity to accomplish reproducible research. On the one hand, platform providers and users should retain their rights. On the other hand, reproducible research on their data should be practically feasible. To this end, an approach is needed that allows scientists to work on the same dataset as their peers without having to share it directly.

1.5 Thesis Structure and Contributions

In the following, we outline the structure of the remainder of this thesis and the contributions made in the individual chapters:

Chapter 2 – Related Work. We provide an overview of related work by comparing all available research datasets of toxic comments and the neural network architectures used for their classification. This comparison reveals a discrepancy between the small size of these datasets and the great complexity of current neural networks, which motivates our work. In the emerging research area of reader engagement in online news discussions, there are no existing publications that address engaging comments, which is due to the novelty of this concept. Therefore, we describe the connections of this thesis to more distantly related work on recommendations of comments and discussions.

Chapter 3 – Novel Comment Datasets and Reproducibility. In this chapter, we introduce novel comment datasets, which we collected in the course of our research, and a technique to measure and facilitate dataset reproducibility. We made the implementation of this technique, those comment datasets that we are allowed to publish, pre-trained word embeddings, and the neural network architectures of our comment classification models all publicly available for research purposes.[7] The datasets' large size, the included labels, and additional metadata enable analyses and experiments that have not been possible before.

Chapter 4 – Classifying Toxic Comments. The first of two main contributions of this thesis is presented in Chapter 4. It consists of data augmentation, transfer learning, and ensembling methods for training specially tailored neural network models on small datasets. We design, implement, and evaluate these models to address the task of toxic comment classification. Top-ranking results in shared task competitions demonstrate their strong performance compared to various approaches from related work. Further, we compare four different approaches to make machine-learned models explainable and thoroughly investigate the practical application of automated comment classification in the context of manual moderation processes. Our collaboration with online news platforms led to the successful integration of machine-learned models into their working routine.

Chapter 5 – Recommending Engaging Comments and Discussions. The second main contribution is the conceptualization of engaging comments and their automatic classification in Chapter 5. To this end, we measure how engaging a comment is by the number of received upvotes and replies. Until now, it was infeasible to use this number as a feature because of an inherent bias. After removing this bias with a newly developed technique, we train a classifier to identify the most engaging comments solely based on their text. Furthermore, journalists form a special user group, for whom we identify relevant comments based on their previous interactions with their readership. We extend our contribution by predicting the most engaging discussions and providing personalized discussion recommendations based on content-based and co-commenter-based embeddings. Thereby, we open up new application possibilities and improve the state of the art in news discussion recommendation.

Chapter 6 – Conclusion. The final chapter concludes this thesis by summing up the results and describing research directions for future work.

[7] www.hpi.de/naumann/projects/repeatability/text-mining.html

2

Related Work

We distinguish two principal directions of related work on comment analysis, which either deal with detecting toxic or engaging comments. The first direction comprises many publications on feature-based and deep-learning-based classifiers, while the second direction is covered by only a few, more distantly related publications. We use *toxic* and *engaging* as collective terms, although they have no standard definitions, and related work might use different terminology.

2.1 Toxic Comments

The supervised classification of toxic comments has been approached with manual feature engineering [31, 50, 90, 113, 124, 165, 172, 204] or (deep) neural networks [16, 64, 128, 129, 137, 199]. While the former combine manually selected features into input vectors and directly use them for classification, the latter learn abstract features from the input automatically. Both methods have their strengths and weaknesses, and what is best depends on the application scenario. On the one hand, neural network approaches appear to be more effective for automatically extracting features from large datasets [219], but suffer from their complexity (many parameters) when it comes to low available resources (training data, training time, inference time, memory). On the other hand, feature-based approaches preserve some explainability through their input features but show inferior classification accuracy in many settings [101, 186, 215]. In the following, we first present the underlying training datasets and then the classification models. Related work on the special topic of explainability is deferred to Section 4.4.

2.1.1 Datasets

All the classification models presented in this chapter rely on manually labeled training data. Two of the main limitations for research progress in the field of toxic comment classification are different labeling schemes and a low amount of available accurately labeled data [91]. In a rather costly process, human annotators check for each and every comment, whether it fits into one of the pre-defined toxicity classes. Because of the inherent ambiguity of natural language, annotators might not always agree on the label. Further, a comment might be perceived toxic in one context but not in another.

Waseem [204] compared annotations by laypeople recruited through a crowd-sourcing platform and experts with theoretical and applied background knowledge. They found that models trained on experts' annotations significantly outperform models trained on laypeople's annotations.

However, researchers have not reached a consensus about what constitutes harassment online, and the lack of a precise definition complicates annotation [77]. There is no common task definition and no labeled standard dataset for comparative evaluation [175]. Instead, different shared tasks used varying terminology: hate speech, toxic comments, offensive language, abusive language, aggression, or misogyny identification. For example, one task dealt with hate speech against immigrants and women [20], other tasks with hate speech in general [29, 110], misogyny [61], offensive language [186, 215], or aggression [22, 99]. Similarly, related work apart from shared-tasks dealt with the detection of toxicity [67, 209], hate speech [16, 32, 50, 64, 66, 70, 168, 175, 199, 203, 204], harassment [72, 212], abusive language [113, 128], cyberbullying [48, 55, 196, 222] and offensive language [39, 210]. Each group of researchers used slightly different task definitions, but their applied methods were similar. Waseem et al. [206] provided an overview of the different tasks. They designed a two-dimensional scheme of abusive language with two dimensions *generalized/directed* and *explicit/implicit. Directed* means a comment addresses an individual, while *generalized* means it addresses a group. *Explicit* means, for example, outspoken name-calling, while *implicit* means, for example, sarcasm or other ways of obfuscation, which was also discussed by Struß et al. [186] and Caselli et al. [34]. Other terms for this dimension are *overtly abusive* and *covertly abusive* [99, 101].

Only a few comment datasets are publicly available for research purposes. The majority of them originate from shared tasks on toxic comment classification, and together, they cover a diverse set of languages. These are, besides English, Arabic [216], Bangla [22], Danish [216], German [110, 186, 207], Greek [216], Hindi [22, 99, 110], Italian [29], Polish [138], Spanish [61], and Turkish [216]. The main advantage of shared task datasets is comparability: various approaches are evaluated on the same data. The largest shared task concerning the number of participants and data samples so far was the Kaggle challenge on toxic comment classification.[1] The dataset comprises 150,000 English, hand-labeled user comments from Wikipedia discussion pages, and we use it in our comparative study in Section 4.1. More than 4,500 participating teams automatically classified Wikipedia talk page comments with regard to six non-exclusive labels: toxic, severe toxic, obscene, threat, insult, identity hate.

Table 2.1 lists datasets used in related work. It reveals that Twitter is the primary data source and that there is no common set of class labels. The median size is only 12,000 labeled samples. Ranging from 1,100 samples [96], 1,500 samples [66], and 2,500 samples [220] in the three smallest datasets to 950,000 samples [56], 1,500,000 samples [129], and 24,600,000 samples [109] in the three largest datasets, they span a broad range of sizes. Two of these datasets exemplify the challenges of collecting and distributing toxic comment data: Zhang et al. [220] collected their dataset via the Twitter API by filtering for a list of keywords, e.g., *muslim, refugee, terrorist,* and *attack* or hashtags, such as *#banislam, #refugeesnotwelcome,* and *#DeportallMuslims.* This step introduces a strong bias because all hateful tweets in the created dataset contain at least one of the keywords or hashtags. Thus, the data is not a representative sample of all hateful tweets

[1] www.kaggle.com/c/jigsaw-toxic-comment-classification-challenge

Table 2.1: Toxic comment datasets (sorted by year of publication).

Study	Size	Source	Lang.	Classes
Kwok et al. [103]	24.6k	Twitter	en	racism
Djuric et al. [56]	950k	news	en	hate
Waseem [204]	6.9k	Twitter	en	racism,sexism
Waseem et al. [205]	16.9k	Twitter	en	racism,sexism
Badjatiya et al. [16]	16k	Twitter	en	racism,sexism
Davidson et al. [50]	25k	Twitter	en	hate,offense
Gao et al. [66]	1.5k	news	en	hate
Jha et al. [88]	10k	Twitter	en	benevolent/hostile sexism
Mubarak et al. [119]	32k	news	ar	reject
Pavlopoulos et al. [130]	1.5m	news	el	reject
Schabus et al. [174]	12k	news	de	seven classes[a]
Vigna et al. [199]	6.5k	Facebook	it	strong hate,weak hate
Wulczyn et al. [209]	116k	Wikipedia	en	attack
Albadi et al. [3]	6.1k	Twitter	ar	hate
Álvarez-C. et al. [5]	10k	Twitter	es	aggressive
Bosco et al. [29]	17.6k	Facebook	it	strong/weak hate
Bosco et al. [29]	6.9k	Twitter	it	five classes[b]
Fersini et al. [61]	8.1k	Twitter	en,es	six classes[c]
Founta et al. [63]	80k	Twitter	en	six classes[d]
de Gibert et al. [51]	10.6k	Forum	en	hate
Kumar et al. [99]	15k	Facebook	en,hi	aggression,covert,overt
Ljubešić et al. [109]	24.6m	news	hr,sl	reject
Sanguinetti et al. [173]	6.9k	Twitter	it	five classes[b]
Wiegand et al. [207]	8.5k	Twitter	de	abuse,insult,profanity
Zhang et al. [220]	2.5k	Twitter	en	hate
Basile et al. [20]	19.6k	Twitter	en,es	aggression,hate,target
Fortuna et al. [62]	5.7k	Twitter	pt	hate,target
Ibrohim and Budi [84]	13.2k	Twitter	id	abuse,strong/weak hate,target
Kolhatkar et al. [96]	1.1k	news	en	very toxic,toxic,mildly toxic
Mandl et al. [110]	17.7k	Twitter	de,en,hi	hate,offense,profanity,target
Mulki et al. [120]	5.8k	Twitter	ar	abuse,hate
Ousidhoum et al. [125]	13k	Twitter	ar,en,fr	group,hostility,sentiment,target
Ptaszynski et al. [138]	11k	Twitter	pl	cyberbullying,hate
Qian et al. [140]	56.1k	misc	en	hate
Struß et al. [186]	8.5k	Twitter	de	abuse,insult,profanity,explicitness
Zampieri et al. [214]	14.1k	Twitter	misc	offense,target
Bhattacharya et al. [22]	20k	YouTube	bn,en,hi	aggression,sexism,covert,overt
Caselli et al. [34]	14.1k	Twitter	en	abuse,explicitness
Çöltekin [46]	36k	Twitter	tr	offense,target
Pitenis et al. [135]	4.8k	Twitter	el	offense
Sigurbergsson et al. [179]	3.6k	misc	da	offense,target

[a] argument,discrimination,feedback,inappropriate,sentiment,personal,off-topic

[b] aggression,hate,irony,offense,stereotype

[c] derailment,discredit,harassment,misogyny,stereotype,target

[d] abuse,aggression,cyberbullying,hate,offense,spam

on Twitter, and models trained on that data might overfit to the list of keywords and hashtags. However, the advantage of this step is that it reduces the annotation effort: fewer annotations are required to create a larger set of hateful tweets. As per Twitter's content redistribution policy,[2] the tweets itself were not released by the researchers but only the tweet ids. These ids allow re-collecting the dataset via the Twitter API.

Ljubešić et al. [109] and Pavlopoulos et al. [129] collaborated with news platforms and publicly released their datasets of reader comments. However, to prevent the distribution of unredacted toxic content, they applied undisclosed but straightforward monoalphabetic substitution ciphers to the data. Consequently, their datasets cannot readily be used without proper decoding. Both examples show that it is challenging to reproduce experiments on comment data. Comments are publicly available on news platforms, but typically, researchers refrain from distributing datasets that they collected or annotated. This restraint is due to the commenters and the platform providers holding rights to the data but also due to the toxic and sometimes illegal content. Section 3.3 is dedicated to the issue of reproducibility, and we discuss related work on this matter in that section.

A challenge that is not visible in Table 2.1 is the inherent class imbalance of many datasets. For example, the class distribution of the dataset by Wulczyn et al. [209] exhibits a bias towards "clean" comments (201,081 clean; 21,384 attack), whereas the dataset by Davidson et al. [50] exhibits a bias towards "offensive" comments (19,190 offensive; 4,163 clean). The latter class distribution is not representative of the underlying data in general. It is due to biased sampling, similar to the issues that we described for the dataset by Zhang et al. [220]. In fact, most comment platforms contain only a tiny percentage of toxic comments. Since research datasets are collected with a focus on toxic comments, they can be biased in a significant way. This focused data collection creates non-realistic evaluation scenarios and needs to be taken into account when deploying models trained on these datasets in real-world scenarios.

We limited Table 2.1 to publicly available, manually labeled toxic comment datasets that were presented in peer-reviewed publications. Consequently, we excluded three datasets that can only be obtained for research purposes by contacting the authors [72, 144, 193]. For four other datasets, it is unclear whether they can be obtained [8, 91, 196, 223]. Further, two datasets contain only machine-labeled samples and were therefore excluded [59, 166]. Last but not least, we excluded datasets that consider users instead of comments as the level of annotation [38, 145], focus on counter-narratives rather than on toxic comments [42], or study a different type of conversation, e.g., WhatsApp chats, where the participants presumably know each other in person [184]. An actively maintained list of toxicity datasets is also available on GitHub.[3]

2.1.2 Feature-Based Classification

The first feature-based approach for the classification of abusive messages on the Web used a decision tree [182]. It was based on syntactic and semantic text features and a hand-written set of rules. Since then, the set of features has remained similar and is homogeneous across many publications. The survey by Schmidt and Wiegand [175] pointed out that surface-level features, such as bag-of-words, have strong predictive power for

[2] www.developer.twitter.com/en/developer-terms/agreement-and-policy

[3] www.github.com/leondz/hatespeechdata

hate speech detection, although they ignore sentence syntax and word order. Word n-grams overcome the latter issue and there are many reports on their good performance [16, 50, 124, 175, 203]. Davidson et al. [50] compared several different classifiers in combination with these features. Among logistic regression, naive Bayes, decision trees, random forests, and support vector machines (SVMs), the authors concluded that logistic regression and SVMs perform best. This finding motivates us to study a logistic regression model in Section 4.5.

The paper by Nobata et al. [124] explored a variety of features, such as word n-grams, character n-grams, linguistic features (length of comment or average word length), syntactic features (part-of-speech tags), and word embeddings. The authors identified intentional obfuscation and the lack of fluency and grammatical correctness as a major challenge, which explains why character n-grams are the best-performing single feature. In contrast to character n-grams, word-based features are prone to out-of-vocabulary issues. They cannot represent words that occur only in the test dataset but not in the training dataset. Toxic comments often use obfuscation, for example, "Son of a B****", "***k them!!!!" but also neologisms and misspelled words, which are common in online discussions. Fast-paced interaction, small virtual keyboards on smartphones, and the lack of editing/correction tools reinforce this problem. However, manually crafted dictionaries also have their applications. Warner and Hirschberg [203] observed that hate speech aimed against specific groups often exhibits stereotypical words. Therefore, they created individual language models for each attacked group and determined word-based features separately. Similarly, Razavi et al. [142] followed a dictionary-based approach. They combined several classifiers with a dictionary of abusive and insulting phrases. According to Schmidt and Wiegand [175], words representing positive or negative sentiment or politeness, e.g., "no thanks", "please", and "would you", are promising features to distinguish toxic from non-toxic content. Polarity classifiers for short texts, such as SentiStrength [190] or VADER [69], can extract those phrases, which is why we use a polarity classifier in Section 4.2. Instead of relying on separate (sets of) words, there have been approaches that applied subjectivity detection [70] or word sense disambiguation [203] to detect toxicity while capturing the semantics of the full sentences.

Yin et al. [212] introduced contextual features that comprise a user's previous and succeeding posts. Their approach requires the availability of a user history. However, almost all publicly available datasets lack any user information, and so it is unknown, which comments were written by the same user. User-based features have been neglected in academic research on comment analysis but are nevertheless relevant for industry applications. Our applied research study in Section 4.5 considers user-based features and demonstrates that they improve classification accuracy.

2.1.3 Deep-Learning-Based Classification

With increasing amounts of labeled training data and the success of deep learning approaches at other natural language processing tasks, toxic comment classification has been approached with deep learning methods. Word embeddings are the basis of these approaches. They transform each word into a vector of typically 50 to 300 floating-point numbers, which then serve as the input layer. As opposed to sparse, one-hot encoded vectors, these dense vectors can capture and represent word similarity by the vectors' cosine similarity. Beyond simple distance measurements, arithmetics with words can be performed as presented with the Word2Vec model [114]. The similar approaches GloVe [132] and FastText [28] provide alternative ways to calculate word embeddings. FastText is particularly suited for toxic comments because it overcomes out-of-vocabulary issues. In contrast to Word2Vec and GloVe, it uses known subwords of unknown words to come up with a proper representation in the form of subword embeddings. The new ability to cope with unknown words is why previous findings [124] on the inferiority of word embeddings compared to word n-grams have become outdated.

Similar to other text classification tasks, neural networks for toxic comment classification use recurrent neural network layers (RNN), such as long short-term memory layers (LSTM) [68, 80] and gated recurrent unit layers (GRU) [41], or convolutional neural network layers (CNN) [106]. An extension to the standard versions of RNN layers are bi-directional layers, which process the sequence of words in correct and reverse order. All recurrent layers, regardless of whether it is a simple RNN, LSTM, or GRU layer, can either return only the output of the last cell or the sequence of the outputs of all cells. If the last output is returned, it serves as a combined representation of the input words. However, the outputs of all cells can be used as an alternative. So-called pooling layers can combine this sequence of outputs. Pooling in neural networks is typically used to reduce an input with many values to an output of fewer values. In neural networks for computer vision, pooling is widespread because it makes the output shift-invariant. Pooling on the word level can make neural networks in natural language processing order-invariant so that a word's exact position in a sequence of words is irrelevant. For toxic comment classification, both average-pooling and max-pooling are common with a focus on the latter. An intuitive explanation for the use of max-pooling over average-pooling is the following. If a small part of a comment is toxic, max-pooling will focus on that part, which will finally result in classifying the comment as toxic. In contrast, with average-pooling, the larger non-toxic part overrules the small toxic part of the comment, and thus the comment is finally classified as non-toxic. The definition of toxicity classes typically assumes that there is no way to make up for a toxic part of a comment by appeasing with other statements. Therefore, max-pooling is more suited than average-pooling for toxic comment classification.

An alternative to pooling after the recurrent layer is an attention layer. Graves [75] originally introduced the attention mechanism for neural networks with an application to hand-writing synthesis. It was quickly followed by an application to image classification [116] and neural machine translation to align words in translations [41]. Further, it has been successfully applied to toxic comment classification [129]. The attention mechanism is basically a weighted combination of all outputs from the preceding recurrent layer. The model can thereby put more emphasis on selected words (or outputs of the recurrent layer) that are decisive for the classification. We investigate the explanatory power of attention mechanisms in Section 4.4 and demonstrate their use as spotlights that highlight toxic words in a semi-automatic moderation process. Finally, a dense layer handles the classification output. Multi-label classification uses a sigmoid activation, and multi-class classification uses a softmax activation in the dense layer.

While RNNs are tailored to sequential processing of the input, CNNs focus more on proximity and divide the input into small sets of neighbored words or characters. The convolution operation applies filter kernels to process each of these sets. These kernels define how the input values in each set are weighted and comprise the trainable parameters of the convolutional layer. Similar to RNNs, pooling layers and dense layers constitute the rest of a CNN, and their final layer is also a dense layer with either sigmoid or softmax activation. The advantage of CNNs with character-level input is that they can deal with obfuscated words and do not rely on a pre-defined vocabulary of words. However, long-range dependencies can hardly be modeled by CNNs with standard filter kernel sizes.

In 2018, a task-agnostic language representation model called bidirectional encoder representations from transformers (BERT) was introduced by Devlin et al. [52] and has since then become popular for a broad range of natural language processing tasks. It consists of multiple layers of bidirectional transformers, which are an attention mechanism developed by Vaswani et al. [198]. Its language model component learns how words are generally organized and combined instead of learning which particular words occur in toxic comments. To this end, BERT comprises a masked language model that allows learning context both to the right and to the left of words, which previous models were not designed to accomplish. After being pre-trained on a large, general corpus, such as Wikipedia, it can not only be fine-tuned for text classification but also for many other tasks, such as named entity recognition, question answering, and text summarization. Thanks to the pre-training, a relatively small amount of a few thousand task-specific training samples are sufficient to fine-tune the model. The fine-tuning adds a task-specific layer. For a classification task, it is a dense layer with a number of neurons that matches the number of output classes and sigmoid activation (multi-label classification) or softmax activation (multi-class classification). BERT has been applied to toxic comment classification for the first time in the context of a shared task organized by Zampieri et al. [215]. Eight teams, including the winning team, used fine-tuned BERT models, and their system description papers were published at the same time. Their approaches differ only slightly in terms of the applied pre-processing (maximum sentence length, cased or uncased, hashtag segmentation, emoji substitution) or the parameter settings for the training process (learning rate, number of epochs). All teams among the top five used BERT, which underlines its strong classification performance.[4] The 2020 edition of the shared task comprised five different languages and thereby drew attention to multilingual BERT models. They are trained to generalize across languages by learning representations in a shared embedding space — even for languages with little vocabulary overlap [134]. The pre-training of the models can be continued on monolingual corpora to expose them to the target language or the target domain, which prepares for the fine-tuning step and further improves classification performance [44].

Table 2.2 provides an overview of neural network architectures and embeddings. For example, for the particular task of hate speech classification (three classes: sexist, racist, or neither), Badjatiya et al. [16] identified a combination of LSTM and gradient boosted decision trees as the best model. Their neural network approaches outperformed various baseline methods (tf-idf or bag-of-words and SVM classifier; character n-gram and logistic regression). Comparing CNNs and RNNs, there is no clear winner in Table 2.2. Both network architectures are of comparable popularity because they achieve comparable performance. However, the training of CNNs is, in general, faster than the training of RNNs because it can be better parallelized. Djuric et al. [56] used comment embeddings

[4] 104 teams participated in the shared task.

Table 2.2: Deep neural network architectures.

Study	Model	Embeddings
Djuric et al. 2015 [56]	–	paragraph2vec
Jha et al. 2017 [88]	FastText[a]	FastText
Park et al. 2017 [128]	CNN	Word2Vec
Badjatiya et al. 2017 [16]	CNN/LSTM/FastText[a]	GloVe, FastText
Schabus et al. 2017 [174]	LSTM	Word2Vec
Vigna et al. 2017 [199]	LSTM	Word2Vec
Pavlopoulos et al. 2017 [129]	CNN/GRU/RNN+Att	Word2Vec
Pavlopoulos et al. 2017 [130]	GRU	Word2Vec
Gambäck et al. 2017 [64]	CNN	Word2Vec
Zhang et al. 2018 [220]	CNN+GRU	Word2Vec
Pitsilis et al. 2018 [136]	LSTM	–
Mitrović et al. 2019 [115]	CNN+GRU	Word2Vec
Zampieri et al. 2019 [215][b]	BERT	–

[a] Joulin et al. [89] provided a stand-alone classifier based on FastText embeddings, which uses the same name.

[b] Multiple research groups first applied BERT to toxic comment classification simultaneously in the context of a shared task by Zampieri et al. [215].

based on paragraph2vec [114] and refrained from CNNs and RNNs. There are already too many BERT-based approaches to list all of them in Table 2.2. The overviews of two shared tasks [186, 215] provide brief summaries of the first English-language, respectively German-language, approaches. Two BERT-based submissions to shared tasks are the basis of Section 4.3 in this thesis, where we present an ensembling approach to train more robust models. A GRU-based neural network architecture that comprises few trainable parameters and thereby copes with small training datasets without suffering from overfitting issues is presented in Section 4.2. Three evaluation datasets used in Chapter 4 stem from shared tasks, and thus, our models can be compared to many related work approaches.

2.2 Engaging Comments and Discussions

The concept of toxic comments has already been established in research with the Kaggle challenge on toxic comment classification.[5] In contrast to that, the concept of engaging comments is new and defined by us as the desirable opposite. In that sense, highlighting engaging comments can be seen as the complementary task to deleting toxic comments. The terminology used in related work varies from "engaging, respectful, and informative conversations" [121] through "desirable content" [174] to "interesting or thoughtful comments" [53], "high-quality comments" [127], and "constructive comments" [95, 96]. In the following, we summarize approaches that study reader engagement in online discus-

[5] www.kaggle.com/c/jigsaw-toxic-comment-classification-challenge

sions. We cover prediction, classification, and recommendation approaches, pursuing the goal of fostering more interaction between readers and detecting valuable contributions.

2.2.1 Predicting Reader Engagement

Related work on predicting the number of comments that a news article receives as a form of reader engagement can be classified into two categories: pre-publication and post-publication scenarios. By the nature of news articles, the attention span after article publication is short, and in practice, post-publication prediction is valuable only within a short time frame. Tsagkias et al. [191] used random forests for classification rather than regression. First, they classified whether an article will receive any comments at all. Second, they classified articles as receiving a high or low amount of comments. The authors found that the second task is much more challenging and that predicting the actual number of comments is practically infeasible. Bandari et al. [17] concluded the same, analyzing Twitter activity as a popularity indicator for news: Popularity prediction as a regression task results in large errors. Therefore, the authors predicted classes of popularity by binning the absolute numbers (1-20, 20-100, 100-2400 received tweets). However, predicting the number of received tweets includes modeling both the readers and the platform, which is problematic. It is part of a platform's business secrets how content is internally ranked and distributed to readers, making it hard to distinguish cause and effect from the outside. Aiming to support the moderation teams, we even see no benefit in predicting the exact number of comments. Instead, in Section 5.1, we predict which articles belong to the weekly top ten percent of articles with the highest reader engagement, which is a task defined by Tsagkias et al. [191].

In a post-publication scenario, Tsagkias et al. [192] considered the comments received within the first ten hours after article publication. Based on this feature, they proposed a linear model to predict the final number of comments. Comparing comment behavior at eight online news platforms, they observed seasonal trends. Tatar et al. [188] considered the shorter time frame of five hours after article publication to predict article popularity. They also used a linear model and found that neither adding publication time and article category to the feature set nor extending the dataset from three months to two years improves prediction results. Their survey on popularity prediction for web content summarizes features with good predictive capabilities and lists application fields for popularity prediction [189]. Rizos et al. [164] focused on user comments to predict a discussion's controversiality. They extracted a comment tree and a user graph from the discussion and investigated, for example, comment count, number of users, and vote score. The demonstrated improvement of predictions with these limited, focused features motivates us to further explore content-based features of comments in Section 5.1.

2.2.2 Classifying Engaging Comments

Similar to the task of toxic comment classification, state-of-the-art approaches for classifying engaging comments use supervised machine learning and require labeled training data, e.g., 2,300 annotated conversations [122] or 30,000 annotated comments [95]. We refrain from costly annotation efforts in Section 5.3 and instead draw on information inherent to the data: upvotes and replies by users.

Napoles et al. [122] laid the groundwork by training a logistic regression classifier to detect “engaging, respectful, and informative conversations” on the Yahoo News platform. Inspired by that, Kolhatkar and Taboada [95] used editor’s picks from the New York Times as positive training samples to learn to identify constructive comments. These picks are a selection of comments judged as interesting or thoughtful by news editors. Negative training samples were taken from conversations that were annotated as non-constructive in previous work [122]. The authors used the data to train a bidirectional LSTM model. However, using positive and negative training samples from two different platforms is a potential source of error. Some samples might contain features that identify the source platform, e.g., by mentioning the platform’s name in the comment text. Inadvertently learning these features circumvents the actual classification task. In Section 5.3, we focus on one platform at a time and carefully sample a non-biased subset from the data. Further, while Kolhatkar and Taboada [95] aimed to engage more readers, we aim to engage also more authors of news articles to join a conversation in Section 5.4.

Except for a handful of publications, most related work refrains from using the number of upvotes as a feature because of the many different factors influencing them, such as a comment’s position. At least, the interplay of a post’s title, text, and publication time to predict user votes on Reddit and YouTube has been subject to research [104, 178]. Chronological ranking in discussion threads is an essential difference in news comments compared to, for example, posts on Twitter or Facebook that can stand alone without a conversational context. Berry and Taylor [21] studied the ranking of posts on public Facebook pages. They compared chronological ranking to ranking via social feedback and found that the latter positively affects response quality. This insight motivates our research on ranking criteria for online comments aside from chronological ranking in Section 5.3. Jaech et al. [85] evaluated the ranking of comments based on their “karma”, which they defined as the number of upvotes minus the number of downvotes a comment received. While they worked on the platform Reddit, which is not exactly a news platform, the task is still similar to ours. Informativeness, relevance, and user reputation were identified to be essential features for finding high-karma comments. However, the importance of these features depends on the community. A combination of received upvotes and downvotes was also used to indicate a comment’s success and popularity in the community [104]. While studying crowd-sourced models for moderation, Lampe and Resnick [105] found that users generally agree on what comments are of high or low quality. Hence, the workload of content moderation can be distributed among many users. However, users pay more attention to the earliest comments and top-level comments in a conversation than to responses. This finding motivates us to identify and remove the position bias in our dataset in Section 5.3. Recent research that focused on how the engagement varies by topic and news platform [4] further motivates us to remove also the article bias.

More distantly related to our work, there is research on the dynamics of re-tweets [94, 218] and on conversation modeling in general [15, 73, 98, 112, 202]. However, the motivation behind re-tweeting is to spread information in a social network, and in this regard, it differs from replies in news discussions. Conversation modeling is similar to our research in a few of the considered tasks, e.g., predicting the number of comments, but it leverages generative models to solve them. Aragón et al. [9] provided an overview of seven generative models of online discussions, which comprise the state-of-the-art in that field. These models predict, for example, the time between two comments of the same user [202], the number of distinct users in a discussion [98], reply cascades (number of consecutive replies in the same subthread) [73, 112], or user re-entry (which commenters will contribute another comment to the same discussion) [15].

2.2.3 Recommending Discussions and Comments

In contrast to news article recommendation [65, 213], where the goal is typically to find interesting articles for the user to read, discussion recommendation aims to find engaging reader discussions for the user to contribute to. While the problem setting is similar, there are three major differences: (1) Users need to authenticate themselves on the news platform to author comments, which allows recommender systems to create user profiles [60]. (2) The data available to make informed recommendations are much richer. Besides the article itself, there are previous comments, along with information about their authors. (3) This extended context leads to a more subtle difference: the reason *why* someone posts a comment. While a recommendation for reading an article can mostly be based on the user's topical interest, the reason for commenting could be the article, other comments, or the fact that one or more particular users have commented. In our approach in Section 5.2, we actually refrain from using the article text as a source of information due to the weak signal in comparison to the comments from other users. Topical interest, as derived from the article itself, is too coarse-grained and shifts over time [141].

There is related work on recommending either single comments, which we focus on in Section 5.3, or entire reader discussions, which we focus on in Section 5.2. Both tasks have the common goal to foster user engagement. Comment recommendations are either personalized [1] or based on a community's preferences [82]. These recommended comments can be integrated into online platforms by adjusting the standard chronological ranking of comments by their estimated relevance to users. Further, a threaded (hierarchical) presentation of comments increases reciprocity compared to a linear presentation: users more often reply back when another user replies to their earlier comment [10].

In the area of recommending discussions, previous approaches typically combine collaborative filtering and content-based recommenders, thus exploiting the available data: co-commenting patterns and article content. In contrast to these approaches, we make use of the comment text instead of the article text for assessing relevance to a user in Section 5.2. Most work along these lines employed topic modeling to model users and content. Bansal et al. [18] combined collaborative topic modeling [201] with matrix factorization [108] to identify comment-worthy articles. Because of the time-consuming Gibbs sampling and the constraints on vocabulary size, the model is tailored to data of smaller size and lower topical diversity. This limitation makes it suitable for specialized blogs with a few thousand users but renders the large-scale deployment for news platforms with more than one hundred thousand users infeasible. Another approach [13] combined collaborative filtering and topic modeling in a learning-to-rank setting. The approach by Shmueli et al. [177] combines memory-based collaborative filtering (CF) and latent factor models for both tag-based and co-commenting patterns but ignores comment content. Their evaluation revealed a challenge of static train-test data splits, also identified by Aharon et al. [2]: These splits do not take into account that the comments in a discussion are added gradually rather than all at once. Further, if users did not join a particular discussion, it might just be that they were inactive during that time. Notably, it cannot be inferred that the discussion topic is irrelevant to them. In Section 5.2, we build on these findings and design a more realistic evaluation scenario. We model *when* each user was active and which comments were published at that time.

Related work on recommendations focused more on review platforms rather than on online discussions. For example, Zheng et al. [221] worked on review rating prediction and developed a model called deep cooperative neural networks (DeepCoNN), which consists of two networks that are joined by a final shared layer. The first network models user behavior using the texts of a particular user's reviews, while the second network models item properties using the texts of all reviews on a particular item. The final layer learns the interaction of users and items to predict review ratings. Both networks are CNNs that get a sequence of words as input and provide latent features of that text as output. The concatenation of the resulting user embedding and item embedding is used as input to a factorization machine [143] to predict review ratings. Seo et al. [176] also joined two networks to learn user and item representations to predict review ratings. However, instead of putting word embeddings directly into a CNN, they introduced an attention layer to learn the importance of words locally and globally. The learned item embeddings were evaluated using an SVM for multi-class classification of items into categories. For both approaches, the main difference to our recommender system in Section 5.2 is that they are solely content-based and that they are tailored to the domain of product reviews. We adapt the architecture to the domain of online discussions and combine it with an additional neural network for collaborative filtering, which captures user co-occurrence patterns.

3

Novel Comment Datasets and Reproducibility

In this chapter, we introduce several novel comment datasets. The datasets were either provided to us by the collaborating news platforms in the form of a database dump, we accessed them through the platforms' web APIs, or we collected publicly available data from the platforms' websites. In the following, we describe the data collection processes and give an overview of the data. We are aware of the responsibility we bear in processing comments by readers who do not know that their comments are a subject of research. Therefore, we follow the guidelines for responsible big data research by Zook et al. [224] and address ethical considerations in the final section of this chapter, Section 3.3. If readers remove their comments from the news platforms so that they are no longer publicly available, our process ensures that they are also no longer used in our experiments. Further, we strictly limit the use of metadata that potentially identifies individuals, such as user ids or user names.

3.1 Zeit Online

Thanks to a collaboration with the German news provider Zeit Online, we have access to two large datasets of reader comments from their platform.[1] The first dataset contains labels by moderators and therefore serves as training data for toxic comment classifiers in Chapter 4. The second dataset is unlabeled but twice as large and also contains news articles. It is used for reader engagement prediction in Chapter 5.

3.1.1 Comment Labels by Moderators

At Zeit Online, a team of moderators enforces the discussion rules and decides which comments to remove from the platform. These comments become invisible to the readers, but they are not deleted from the platform's database. Thus, every decision made by a moderator corresponds to one more data sample labeled by an expert. Our labeled dataset consists of all comments from Zeit Online between January 1st, 2016 and March

[1] www.zeit.de

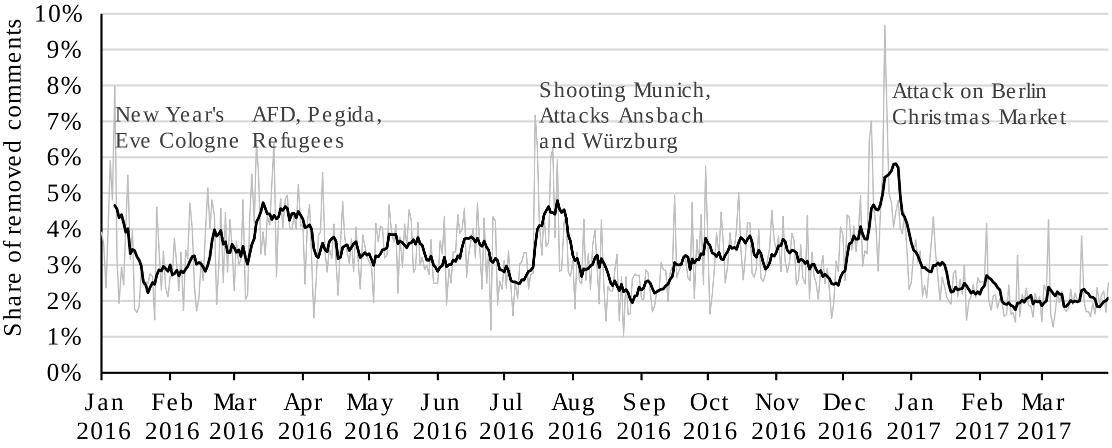


Figure 3.1: At Zeit Online, the rate of removed comments (light gray) aggregated with a seven-day moving average (black) peaks at the date of specific news events.

31st, 2017. In total, there are about 3 million comments by 60,000 readers associated with 26,000 articles. Besides the comment text, the dataset contains the user id, the timestamp, the article URL, an optional reference to the parent comment (for replies only), and the moderation label. The labels of 100,000 comments reveal that they have been removed by a moderator. The high value of this dataset is underlined by the time that has been invested in moderation. If one person needed to label all 3 million comments, spent only 10 seconds each, and worked 24 hours, the task would take almost a year — 347 days, to be precise. Unfortunately, even for research purposes, we are not allowed to publish the comments labeled as toxic and removed from the platform, making them the only data used in this thesis that cannot be shared.

Figure 3.1 visualizes that the amount of removed comments varies over time, primarily due to exceptional or unforeseen high-impact events. The rate of removed comments varies between roughly 2 percent and 10 percent. We indicate events possibly causing the temporary changes with labels in the figure. For example, terror attacks typically result in emotional and controversial debates, prone to include toxic remarks. It is worth noting, though, that the general increase in comments following those events might change the way moderators work. Under stress, moderators might choose to follow stricter moderation policies. This decision – consciously or unconsciously – might result in a higher rate of flagged comments, even though they do not seem to be objectively worse than similar comments at a different point in time.

Figure 3.1 also shows that events that gain the most attention are related to social, political, or security issues. This circumstance is also mirrored in the news sections with the highest rate of removed comments: the sections related to society and politics have a rate of 4.1 percent and 3.4 percent, respectively, while also containing the majority of all comments (70 percent). In contrast, the average rate across all other sections is only 2.1 percent, being the lowest in the business section (1.7 percent). Furthermore, comments are not distributed uniformly among the readers: about 6 percent of the readers posted more than 200 comments, while roughly 48 percent of the readers posted only once or twice. To our surprise, there is one reader who posted 11,082 comments — within only 15 months. 120 of these comments were flagged as inappropriate and thus removed.

In addition to the platform's guidelines, as described in Section 1.2, we observed that moderators removed links to foreign-language content. As not all readers can be expected to understand foreign languages, such content hinders them from joining the discussion. Furthermore, moderators remove duplicate comments from the platform, which they do by flagging a duplicate just in the same way as they would do for insults or hate speech. Duplicate detection and hate speech detection are quite different tasks that ask for different approaches. For this reason, we filter exact duplicates in a pre-processing step and resolve data inconsistencies with data cleansing techniques. We aim to detect toxic comments that have been moderated because of insults, discrimination, and defamation, but also unverifiable suspicions, which do not rely on plausible arguments or credible sources. These comments hinder a respectful discussion directly. We do not focus on comments flagged due to copyright infringements, web links to inappropriate content, or personally identifiable information.

3.1.2 Engagement Prediction

Our second dataset from Zeit Online is unlabeled and contains only those comments that passed the moderation. However, it consists not only of 7 million comments but also of the corresponding 130,000 online news articles published between 2009 and 2017. Out of 174,699 users in total, 60 percent posted more than one comment, 23 percent more than 10 comments, and 7 percent more than 100 comments. For both, articles and comments, extensive metadata is available, such as author list, department, publication date, and tags (for articles), and user name, parent comment (if posted in response), and number of upvotes (for comments). Unsurprisingly, the dataset follows a popularity growth with an increasing number of articles and comments over time. While Zeit Online published roughly 1,300 articles per month in 2010, and each article received roughly 20 comments on average, they nowadays publish roughly 1,500 articles per month, each receiving 110 comments on average. As the dataset's articles and comments cover a time span of several years and many different departments, they deal with a broad range of topics. More than 50 percent of the comments were posted in response to articles in the politics department. On average, an article received 90 percent of its comments within 48 hours.

3.2 The Guardian

The website of The Guardian[2] reaches more than 35 million monthly readers making it the third most popular online newspaper in the UK.[3] We collected a dataset that comprises all 61 million reader comments published between 2006 and 2018. 1.2 million readers posted them in discussions associated with 600,000 news articles. Each comment in our dataset is represented with a comment id and its text. Further, the user id, timestamp, number of upvotes, and corresponding article URL are given. The latter includes the article's news section, such as politics, sports, or lifestyle. Those comments that are a reply to another comment reference their parent with its comment id. In contrast to our experiments on the Zeit Online datasets, we use only publicly available

[2] www.theguardian.com

[3] www.pamco.co.uk/pamco-data/latest-results

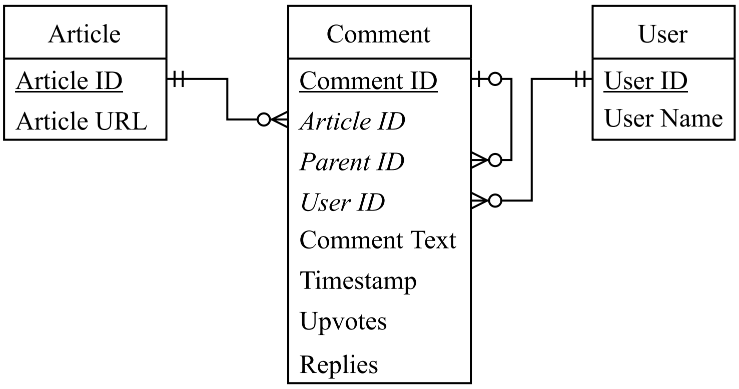


Figure 3.2: Our unified data model for online news discussions considers comments and their publicly available metadata, including references to users, news articles, and other comments if they were posted as a reply.

metadata from The Guardian. Thus, similar datasets could be collected from other online news platforms. Figure 3.2 shows our unified data model, which we also used for other comment datasets. The dataset from The Guardian contains only those comments that remain visible on the platform after moderation. Therefore, it is not suitable for training toxic comment classifiers. Instead, we use the data for experiments concerning reader engagement in Chapter 5. To this end, we distinguish three subsets of the data, which we introduce in the following.

3.2.1 Engaging Comments

We consider the number of upvotes and replies that a comment receives as a measure of reader engagement. Half of the comments (53 percent) are replies to another reader's comment and thus refer to this parent. Before November 2011, there was no option to post a reply in reference to another reader's comment on *TheGuardian.com*.[4] Therefore, we limit the dataset to the time after 2011 whenever we study the replies. We neglect that there was another change on the platform in 2012, when a single-level threaded design was introduced for the comment section.[5] Budak et al. [30] analyzed the impact of this change on readers and their discussions.

Upvotes cover the full time span from 2006 to 2018, so that there is no need to limit the dataset when we study them. There are 260 million upvotes in total. While we have no knowledge of when, why, and from whom a particular comment received upvotes, we do know its final number of upvotes. However, there is a bias in the upvotes and replies: the number of upvotes and replies that a comment receives depends on its position in the chronological ranking and the article's topic. We present a method to remove this bias and a subsequent analysis of reader engagement in Section 5.3.

[4] www.theguardian.com/help/insideguardian/2011/nov/03/responses-in-comments

[5] www.theguardian.com/commentisfree/2012/dec/03/threading-arrives-on-comment-is-free

Table 3.1: Statistics of the dataset of journalists' interactions with their readership.

Comments	56 631
Articles	4 563
Readers	18 084
Journalists	432
Reader Comments with/without Journalist Reply	18 877 / 18 877
Min/Median/Max Comment Length (Chars)	1 / 239 / 4 985
Journalist replies	18 877
Min/Median/Max Reply Length (Chars)	1 / 151 / 4 542

3.2.2 Engaging Discussions

The comments from The Guardian contain user ids, and thus we are able to reconstruct the history of a reader's comments. This data allows studying personalized recommendations of discussions to readers in an offline scenario. Our main experiment is based on a recommendation task for a hold-out set of comments, where we predict in which discussion a reader posts a comment. We select appropriate relevant and irrelevant discussions for each reader in the training, validation, and test set. *Appropriate* means that we consider only those discussions and only those comments that were available at the time when the reader visited the website. This selection and its motivation are described in more detail in Subsection 5.2.2. As a privacy-preserving step, the usage of user ids instead of user names introduces a pseudonymization. Still, the data contains information that potentially allows identifying individuals. The reported results are aggregated, and we are not interested in results for individual users.

3.2.3 Journalists' Interactions with Their Readership

The third subset of the data from The Guardian investigates the interactions of the journalists with their readership. It exhibits a balanced structure: half of the comments received a reply from the journalists, while the other half did not. From the time span between November 2011 and December 2018, we selected all 18,877 reader comments that received a reply from the journalist who authored the corresponding news article (positive samples) and a set of 18,877 reader comments that did not receive a reply from the journalist (negative samples). The dataset also contains the 18,877 replies from journalists. The negative samples are randomly sampled reader comments that did not receive a reply but were posted in a short time window before and after the journalist reply itself. This time window starts one hour before the journalist reply and ends one hour after it. Thereby, we can assume that the journalist was active on the platform and could have chosen to react to the negative sample. The journalists can be identified because the profile names of their official accounts match with the article author names. After matching the articles with the user id of their authors, the user names were discarded. Statistics of the dataset are presented in Table 3.1. Section 5.4 describes the data collection and the data enrichment process involving machine labeling and manual labeling in more detail. Further, it demonstrates that the data can readily be used for supervised machine learning due to its simple, balanced structure.

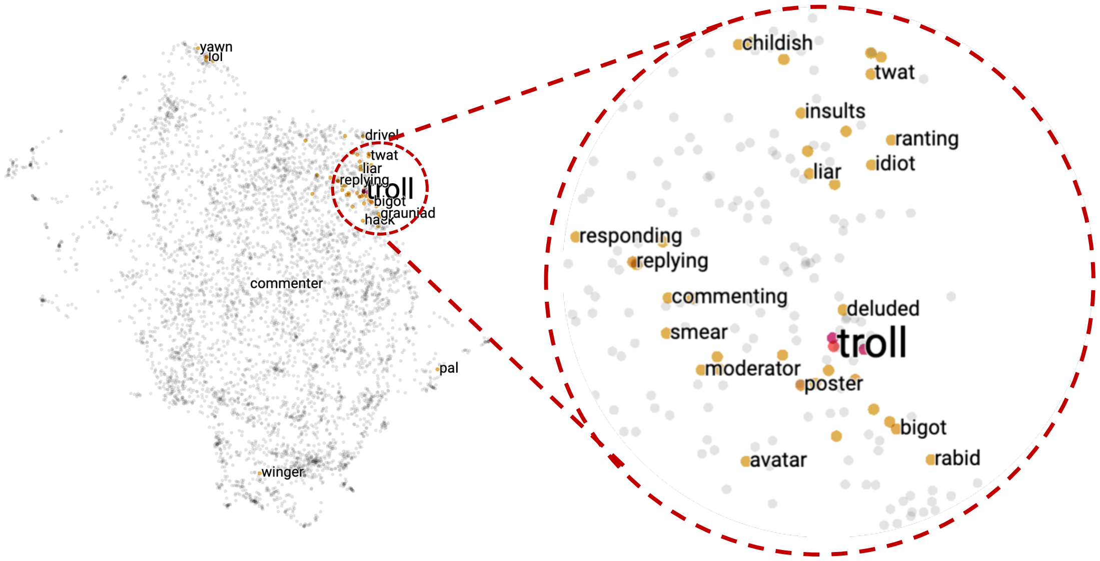


Figure 3.3: This two-dimensional projection of the domain-specific word embeddings trained on 61 million comments from The Guardian highlights the nearest neighbors of the word *troll.* In the high-dimensional space, *troll* is embedded close to *trolling, trolls, commenter, poster,* and several swear words.

3.2.4 Domain-Specific Word Embeddings

For Word2Vec, GloVe, and FastText, various pre-trained word embeddings are available, which have been trained on large corpora of Wikipedia pages, tweets, or websites. However, with enough data available, an alternative is to train embeddings from scratch on domain-specific data. We pre-train 300-dimensional word embeddings on the full dataset of all 61 million comments from The Guardian with the FastText method [28]. This corpus comprises 4.4 billion tokens, and its size is comparable to Wikipedia with 4 billion tokens. The full text is transformed to lowercase, and user mentions and URLs are replaced with special tokens. The embeddings are trained for five epochs using the skip-gram method and subwords of three to six characters. After pre-training, the weights of the word embedding layer remain fixed during the training of downstream neural models for the individual classification tasks. Figure 3.3 visualizes a two-dimensional projection of the embedding space with a focus on the nearest neighbors of the word *troll.* We enable further exploration of the embeddings through the interactive web application Embedding Projector by Google.[6] The interactive visualization and the embeddings are available on GitHub.[7]

3.2.5 Validation Dataset: Daily Mail

Similar to the dataset from The Guardian, we collected reader comments from Daily Mail for validation purposes.[8] With approximately 36 million monthly readers, it is the

[6] www.ai.googleblog.com/2016/12/open-sourcing-embedding-projector-tool.html

[7] www.github.com/julian-risch/CIKM2020#word-embeddings

[8] www.dailymail.co.uk

Table 3.2: Dataset statistics before selecting task-specific subsets.

				Comments per User		
News Platform	Comments	Articles	Users	Min	Median	Max
Zeit Online	6,831,741	134,039	174,698	1	2	23,290
The Guardian	61,491,775	626,396	1,247,647	1	2	64,031
Daily Mail	129,732,986	1,368,219	1,764,558	1	2	313,057

second most-read online newspaper after The Sun (40 million).[9] The dataset comprises 130 million comments posted by 1.8 million readers in discussions about 1.4 million news articles. It covers the time span between 2009 and 2018 and follows the same unified data model as visualized in Figure 3.2. This dataset aims to exemplify that our research is not specific to Zeit Online and The Guardian, but can also be applied to other platforms. Table 3.2 gives an overview of the datasets' sizes before selecting subsets for the individual tasks and experiments. The dataset of Zeit Online is considerably smaller than the other two datasets, and with an average number of 51 comments per news article, the reader discussions are less extensive (compared to 98 at The Guardian and 95 at Daily Mail). On all three online platforms, the majority of the readers posted only one or two comments. However, there are also outliers, power users, who posted thousands of comments.

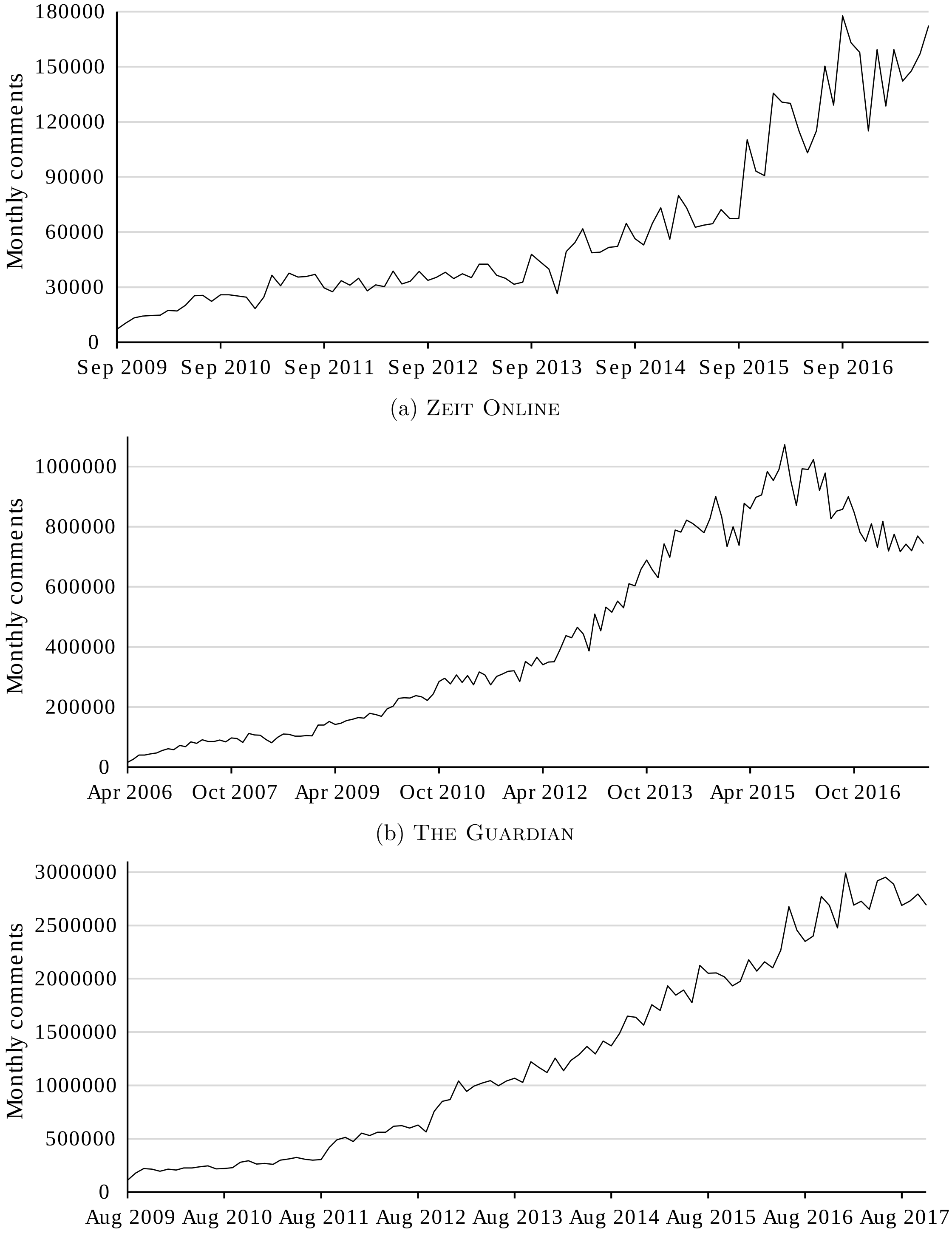
Figure 3.4 visualizes the number of monthly comments posted on the platforms of Zeit Online, The Guardian, and Daily Mail. During the last decade, this number increased for all platforms, but there was a noticeable decline at The Guardian in 2016. Interestingly, the number of commenters and article discussions exhibits the same decline. We assume that The Guardian decided to allow discussions only on selected articles starting from early 2016. At that time, the platform published an analysis of toxic comments and discussed possible changes to the comment section with the readers.[10,11]

Comparing the comment sections of Zeit Online, The Guardian, and Daily Mail, there is a significant difference in the functionality of the upvotes. Readers of Zeit Online and The Guardian need to be logged in to upvote, whereas readers of Daily Mail do not need to be logged in, and Daily Mail additionally allows downvotes. Only Zeit Online and The Guardian provide publicly available ids for individual comments on their online platforms but not Daily Mail. For the latter, sharing the comment ids with other researchers to allow replicating the data is impossible. The lack of ids thus poses an additional challenge for the reproducibility of experiments on the dataset from Daily Mail. We address reproducibility issues in the next section.

[9] www.pamco.co.uk/pamco-data/latest-results

[10] www.gu.com/commentisfree/2016/mar/27/readers-editor-on-closing-comments-below-line

[11] www.gu.com/technology/2016/apr/12/the-dark-side-of-guardian-comments



(a) Zeit Online

(b) The Guardian

(c) Daily Mail

Figure 3.4: The monthly number of comments overall increased in the last decade. However, starting from 2016, The Guardian allowed reader discussions only on selected articles, which led to a declining number of comments.

3.3 Measuring and Facilitating Dataset Reproducibility

This section is dedicated to the dataset aspect of experiment reproducibility, which we refer to as *dataset reproducibility*. Researchers rarely publish comment datasets, and comment analysis experiments are especially difficult to reproduce for several reasons:

1. Copyright-protection and privacy regulations hinder redistribution of the datasets even if it is only for research and not for commercial purposes;
2. Web pages containing comments are dynamic and thus, scraping at different points in time can lead to different results;
3. Web scrapers collect comments from different platforms in various formats, which need to be integrated and pre-processed in the same way.

Related work that conducts experiments on tweets makes the tweet ids publicly available and thereby supports the re-creation of the datasets via the Twitter API [50, 64, 128, 204, 205]. This process is called re-hydration. However, there is no equivalent for comment datasets from online news platforms. We deal with this issue by introducing a process to measure and facilitate dataset reproducibility and demonstrate its application. Our process consists of two components: a scraping component and a fingerprinting component. If researchers cannot publish their dataset, we suggest that they instead publish implementations of these two components: a process to re-create the data in the form of a scraping tool and a process to create and compare fingerprints of the data to check for any changes that occurred in the meantime. Our experiments demonstrate that comment datasets can be re-scraped with negligible changes to reproduce an experiment even after one year. Thus, we conclude that the readers' option to delete their comments and the researchers' desire for reproducibility are not necessarily mutually exclusive, but can exist in parallel.

3.3.1 Repeatability, Reproducibility, and Replicability

The Association for Computing Machinery (ACM) recently updated its definitions of repeatability, reproducibility, and replicability to align them with the terminology used by the National Information Standards Organization.[12] We use its definitions for this thesis:

Repeatability (same team, same setup): The original authors can re-run an experiment with the same setup and come to the same conclusions.

Reproducibility (different team, same setup): Different researchers than the original authors can re-run an experiment with the same setup using the authors' artifacts and come to the same conclusions.

Replicability (different team, different setup): Different researchers can independently re-run an experiment without using the authors' artifacts and come to the same conclusions.

[12] `https://www.acm.org/publications/badging-terms`

These definitions are not universally accepted. For example, according to Cohen et al. [43], replicability and repeatability interchangeably describe the ability to recreate an experiment exactly as reported. In contrast to that, reproducibility describes the ability to come to the same conclusions, findings, or values as reported even if a different method is used. Reproducibility considers not only algorithms and their implementation in code, but also theorems and their proofs, and datasets [169].

While repeatability, reproducibility, and replicability seem to be a foundation of science, it is by far not the standard in today's computer science research. Out of 601 papers from ACM conferences and journals, only one third provides source code [45]. Similarly, two independent studies on IEEE Transactions on Image Processing found that only one-third of the papers make datasets available online [97, 197]. The ACM, the information retrieval community, and the database community recently intensified their efforts to encourage reproducibility and replicability.[13] The first step to accomplish this ambitious goal is to ensure availability. All information necessary to re-create an experiment, which comprises software, datasets, experiment setups, and steps to render result graphs[14], needs to be published.

Currently, there are two ways to make data available in a way that enables re-running an experiment: the dataset itself can be provided or a way to generate the dataset can be described. Regarding the first, there are online data repositories specialized in storing research datasets, such as Mendeley.[15] However, a quick survey of such repositories shows that only a small minority of researchers use them. And again, legal restrictions might exclude this option completely. Regarding the second possibility, datasets are generated according to pre-defined probability distributions, e.g., benchmarks of database algorithms typically fall into this class. Popular benchmark data generators are dbtesma[16] and dbgen.[17] Several tools attempt to improve reproducibility by creating self-contained packages for experiments [40, 81, 87, 133]. Pedersen [131] suggested planning for software releases from the start of a research project. More open-source software in machine learning would allow researchers to build on each other's tools [181]. We transfer this idea to data and suggest to plan from the start for releasing datasets or a way to obtain them. Only if datasets can be accessed and modified, the research community can enrich existing datasets, connect them, and build them together. In line with this, Bogers et al. [27] envision a repository of interactive information retrieval resources to enable and promote their re-use. Vitek and Kalibera [200] go one step further and question any experiments on unpublished (proprietary) datasets. According to them, researchers can learn something from others' experiment results only if they can inspect and understand the dataset. And even if the data is available, Drummond [58] emphasizes that it is important to document also the way it has been collected. For example, this documentation helps to reveal sampling bias or, worse, purposive sampling.

Blockeel and Vanschoren [26] presented experiment databases and how to construct them. The idea is to store all information about experimental setups in one online repository, which can be queried by other researchers. The vision of a Linked Open Data graph

[13] www.acm.org/publications/policies/artifact-review-badging, www.ecir2021.eu/call-for-reproducibility-track, www.db-reproducibility.seas.harvard.edu, www.vldb-repro.com

[14] www.vldb-repro.com/#process

[15] https://data.mendeley.com

[16] www.sourceforge.net/projects/dbtesma

[17] www.github.com/electrum/tpch-dbgen

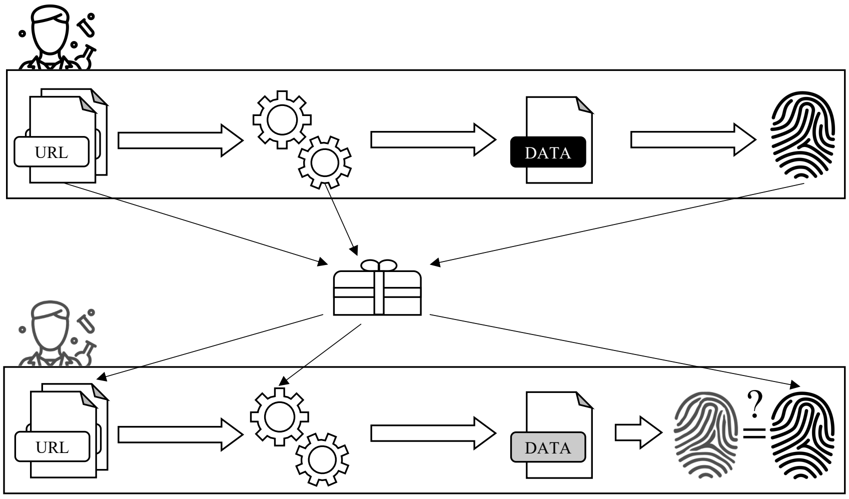


Figure 3.5: A researcher (top) defines a set of URLs to collect a dataset and calculates its fingerprints. URLs, a scraping tool, and fingerprints are provided to a second researcher (bottom), who re-scrapes the URLs and obtains a different version of the dataset. The fingerprints of these versions are then compared.

of related experiments goes into a similar direction [126]. As a first step towards this vision for the field of comment analysis, we monitor to what extent comment datasets on the Web remain unchanged. Blanco et al. [23] proposed a standardized evaluation framework for the semantic search domain. This framework comprises standard datasets, queries, and metrics. The authors find that even crowd-sourced relevance judgments are reproducible in their experiment. Godbole et al. [71] studied the re-usability of research on text mining, for example, on entity extraction. They focus on dictionary-based approaches and bring forward best practices to make dictionaries re-usable across datasets, such as a service-oriented modular approach. To the best of our knowledge, no previous work studies the reproducibility of experiments on comment data and its inherent dynamics.

3.3.2 Process Components

Figure 3.5 visualizes our proposed process, which consists of a scraping component and a fingerprinting component. Their combination allows re-scraping a dataset, estimating the extent of changes compared to the original version, and identifying a data subset that remained unchanged. Thereby, it can be estimated whether the re-scraped data is sufficient to re-run experiments in a comparable way. If so, the experiment can claim *dataset reproducibility*.

Scraping Component. The first component of the process addresses legal restrictions or ethical concerns that hinder the publication of comment datasets. If researchers published comments containing personal data, affected persons could not remove their records from the dataset, and neither could the original platform provider. Even if the users removed their data from the platform, it would remain in the published dataset. In contrast to that, our proposed approach ensures that the user and the platform provider can edit the data. At the extreme, the provider could prevent the usage of scrapers on the platform or take the data offline.

At first glance, this might seem like a major disadvantage for research. But this cost is necessary to allow users and platform providers to retain control of their data. In fact, for researchers, it comes with the advantage that datasets can be deleted locally after an experiment. There is no need to keep the data stored in a decentralized way. The only place to store the data is the original provider. We assume that typically only slight changes are made, and thus, *partial dataset reproducibility* is ensured. To this end, the scraping component implements a way to extract a dataset from the Web. To accomplish reproducibility, web content that needs to be obtained requires an identifier. On the Web, the most typical identifier is a Uniform Resource Identifier (URI), and if the location is specified, a Uniform Resource Locator (URL). Therefore, the scraper needs to be accompanied by a list of URLs from which to collect the data. The scraping process can run in parallel. For example, the list of URLs can be separated into smaller lists for multiple scraper instances.

The implementation of this component can be accomplished in different ways. The most naive way is to scrape every web page in the specified list, such as news article pages, and extract the desired content. Some websites provide an application programming interface (API), which can be used instead of actual web pages. APIs reduce necessary data transfer but oftentimes also limit the number of API calls per day. Collecting 250 million comments with a rate limit of 1000 comments per day would take over 685 years. In rare cases, access to web content might be limited based on geolocation. Thus, the scraper must be used through this location, for example, with a proxy server.

Once the comments have been fetched from different sources, they need to be integrated into a common data structure. Unifying various data formats and boilerplate removal are the challenging tasks for this step. It ensures that further processing of the data, for example, in experiments, works on a well-defined basis. Further, it can normalize different data formats on the same platform if they change over time. To connect this component with the second one, every unit that can be scraped independently should be accompanied not only with a unique identifier but also with a fingerprint. The identifier is necessary so that a re-scraped version's fingerprint can be matched and compared to the initial version's fingerprint.

Fingerprinting Component. The second component checks whether parts of the comment data have changed. In general, it is essential for web crawlers and scrapers to know whether the web content of interest has changed since the last visit. If it is unchanged, there is no need to update, for example, a search engine's index. Hash functions are used to detect content changes, and they can also be applied to unstructured text data. A popular function is Charikar's locality-sensitive simhash function [37], which has been applied for web crawling [111]. In contrast to cryptographic hash functions, similar input texts are hashed to similar hash values. We use this property to estimate the number of changed words based on the difference between two hashes. If hashes are used to compare larger amounts of data, they are called fingerprints, and the underlying technique is fingerprinting.

In our process, fingerprints are taken after initially collecting the data and after each re-scraping. A comparison of the fingerprints serves as an integrity check without having to compare the actual data. Moreover, a similarity function defined on the fingerprints allows measuring the extent of the changes. For example, this similarity can estimate the number of changed words in a comment. The fingerprints further allow identifying which subset of the data has changed and which has remained unchanged. Thereby, an experiment can be reproduced on an unchanged subset for better comparability.

Fingerprinting methods and especially locality-sensitive hash functions have properties that come in handy for detecting content changes. One of these properties is also desirable in our scenario: small content changes result in small fingerprint changes. This property is not guaranteed vice versa because of potential hash collisions. There is a small chance that two records with the same fingerprint are significantly different content-wise. However, a 64-bit fingerprint has a range of 2^{64} values, and thus, the probability of collisions when hashing 250 million records ($\approx 2^{28}$) is rather small. Therefore, in practice, similar fingerprints are assumed to mirror similar content.

The fingerprinting component checks whether a re-scraped dataset differs from the original version. If so, this component measures the difference and identifies the largest unchanged subset of records. An implementation needs to define a fingerprinting function and a distance function. The challenge is to find functions that allow distinguishing slight changes that do not hinder reproducibility from drastic changes that prevent reproducibility. There are several reasons why a re-scraped web dataset might differ from the original one:

1. Parts of the web page have changed, for example, content has been added;
2. The full web page has changed, for example, it has been deleted or moved to a different URL;
3. The website's API or source code has changed, and thus, the scraping tool does not work anymore.

More formally, the fingerprinting function ϕ maps arbitrary web content $x \in W$ to a fingerprint $y \in 0, 1^n$, where W is the domain of web content, and n is the number of bits used for the fingerprint. The distance of two fingerprints y_1 and y_2 is defined as their Hamming distance (number of differing bits) and thus is a natural number in the interval $[0, n]$. The similarity function SIM maps pairs of web content $x_1, x_2 \in W$ to real numbers between 0 and 1:

$$\begin{aligned} \phi : W &\mapsto \{0,1\}^n \\ \text{SIM} : W \times W &\mapsto [0,1] \\ \text{Hamming distance} : \{0,1\}^n \times \{0,1\}^n &\mapsto [0,n] \end{aligned}$$

An example of SIM is the edit-distance for texts, with the adjustment that the integer distance is mapped to real numbers between 0 and 1. ϕ is called a locality-sensitive hash function *corresponding* to the similarity function SIM. Similar input (according to SIM) is mapped to similar fingerprints (according to their Hamming distance).

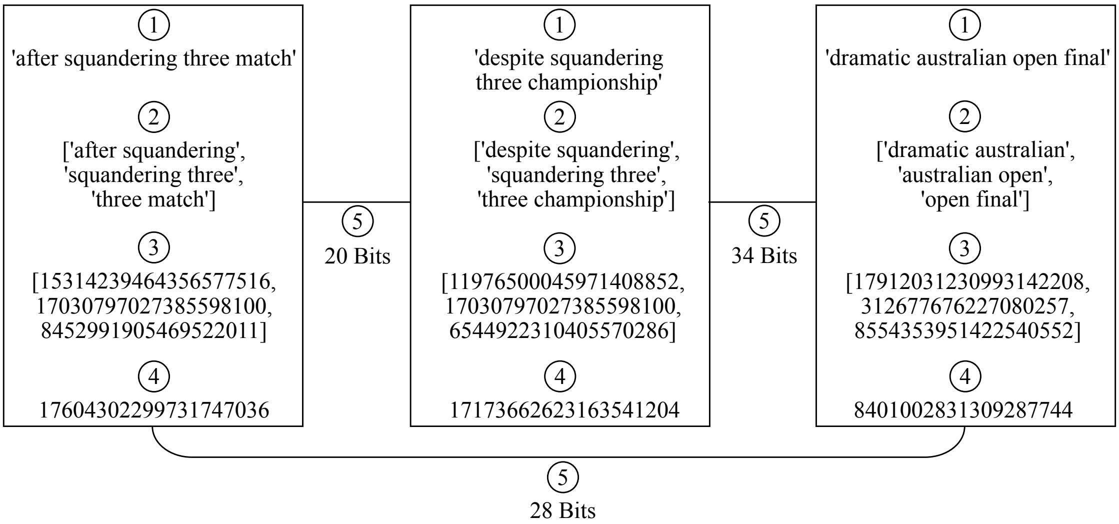


Figure 3.6: A text (1) is transformed to word bi-grams (2) and an md5 hash is calculated for each of those (3). A locality-sensitive hash of this bi-gram sequence serves as a fingerprint (4). Fingerprints can be compared based on their Hamming distance (5).

There is a trade-off between the granularity of discoverable data changes and memory consumption. On the one hand, if one fingerprint represents multiple units, the granularity of discoverable data changes gets worse. On the other hand, memory consumption to store the fingerprints decreases. For text data, the fingerprinting component takes a text as input and tokenizes it. The tokenized text is then converted to k-shingles, where k denotes the number of words of each shingle. Shingles are all possible consecutive subsequences of k tokens. A different name for the same concept is word n-grams. A hash function (which does not need to be locality-sensitive) is applied to each shingle. Typically, the *md*5 hash function is used. From the sequence of hashes, a fingerprint is taken. We use fingerprints of 64-bit length. This fingerprint can be compared to others based on their Hamming distance. Figure 3.6 exemplifies this procedure.

3.3.3 Experiments

We implement the process and demonstrate its practical feasibility with a collection of 250 million reader comments from five English-language online news platforms (Table 3.3).[18] We collected the comments for the first time in 2018 and then re-scraped them based on the same URLs one year later. For the two versions of the dataset, we measure the exact number of changed comments (ground truth). The difference between the fingerprints serves as an estimation of this number. All data is integrated according to the unified data model introduced in Section 3.2 and visualized in Figure 3.2. With a second experiment, we demonstrate that the process also works for other web data, in particular news articles.

[18] Links to these platforms are `www.dailymail.co.uk`, `www.theguardian.com`, `www.foxnews.com`, `www.independent.co.uk`, `www.rt.com`.

Table 3.3: Statistics of the initially collected version of the dataset.

News Platform	Comments	Articles	Users
Daily Mail	129,732,977	1,414,258	1,764,557
The Guardian	61,491,774	625,690	1,213,555
Fox News	52,224,398	49,266	465,954
The Independent	5,598,425	171,052	211,114
Russia Today	687,436	65,384	49,333

Reader Comments. The reasons why a re-scraped comment dataset might differ from the original one are:

1. a comment has been added;
2. a comment has been deleted;
3. the full article has been deleted or moved;
4. the way to access comments has been changed for all articles.

We assume that the reader discussions of most articles remain unchanged within one year. One reason for this assumption is the relatively short attention span in online news. An article is rarely commented on a few days after its publication. In Section 3.1.2, we described that 90 percent of an article's comments at Zeit Online are posted within two to three days, which supports our assumption. A second reason is that news platforms typically close each reader discussion after some time. No more comments can be added, and thus, moderators can focus on a smaller set of discussions.

Except for The Guardian, news platforms in our study do not provide identifiers for comments. URLs identify only news articles and their respective discussions. Therefore, we calculate one fingerprint per full discussion.[19] More specifically, a first step calculates one fingerprint per comment based on shingles of length eight and the simhash function [37]. The comparison of fingerprints uses the Hamming distance. Thus, if a comment is slightly changed, its fingerprint remains similar. A second step calculates a single fingerprint for the full sequence of all fingerprints of an article's comments. Again the fingerprint is based on shingles of length eight and the simhash function. As a result that can be published online, we store an article URL and a fingerprint per discussion.

After a one-year pause, we use the URLs to re-scrape the comments. In the following, we first compare the actual records of the two datasets and then their fingerprints. Two comments are assumed identical if their timestamps and texts are exact matches. The question is: How many comments of the original dataset have been re-scraped successfully and did not change within one year?

Table 3.4 lists the relative number of re-scraped comments and articles per news platform. About 90 percent of the original number of comments and articles have been

[19] For the dataset from The Guardian, the alternative is to share a list of the identifiers of the comments. The only disadvantage of this method is the size of the list: roughly 500MB for 61 million comments.

Table 3.4: Relative number of unchanged comments and articles.

News Platform	Re-scraped Comments	Re-scraped Articles
Daily Mail	.89	.94
The Guardian	.99	.93
Fox News	-	-
The Independent	.73	.83
Russia Today	.88	.99

retrieved. For The Guardian, 61,469,631 out of 61,491,776 comments are retrieved (more than 99.9 percent). In contrast, Fox News switched its third-party commenting system from Disqus to Spot.IM within the considered period, which renders earlier comments inaccessible. Thus, this platform is excluded from the rest of our study.

For The Independent, the original dataset contains about 5.6 million comments. When we re-scrape the same article URLs, 1.6 million comments are missing. For the large majority (1.4 million) of the missing comments, the article URL does not correspond to a retrievable article anymore. The remaining 0.2 million missing comments have been deleted from the platform since the first crawling. The median percentage of unchanged comments is 0.97, while the mean percentage is 0.76. The median is much higher than the mean because there is only a small number of articles with a large number of missing comments.

However, a similar number of retrieved comments does not necessarily mean that their content did not change. For example, on the English-language platform Russia Today, moderated comments are replaced with the text "DELETED". While 99 percent of the articles were re-scraped successfully, only 88 percent of the comments were retrieved. The 12 percent missing comments are distributed across 55 percent of the article discussions. Thus, only 45 percent of all discussions remain unchanged. We come to a similar conclusion if we compare 64-bit fingerprints of the discussions based on shingles of length four. They suggest that a relatively large set of article discussions changed (34 percent). We assume that this underestimation is due to rather small shingles and a too-small number of bits per fingerprint. Moderated comments on the platform of The Guardian are typically replaced with the text "Deleted by Moderator.". Replacements like this make up about five percent of the comments in our dataset. Thus, the scraping can recreate at most 95 percent of the original data. This limitation is the reason why only *partial* dataset reproducibility can be achieved.

The second experiment is a simulation. We artificially alter the initially crawled dataset stepwise by deleting comments and adding others. Each step randomly selects an article and replaces a random comment of this article with a random comment from a different article. Roulette wheel selection favors articles with more comments. We assume that longer discussions are more likely to change, not only because they contain more comments but also because they involve more users. Each simulation step also updates the discussion's fingerprint and the distance to the fingerprints of the original dataset. Figure 3.7 visualizes the linear relationship of comments and fingerprint bits

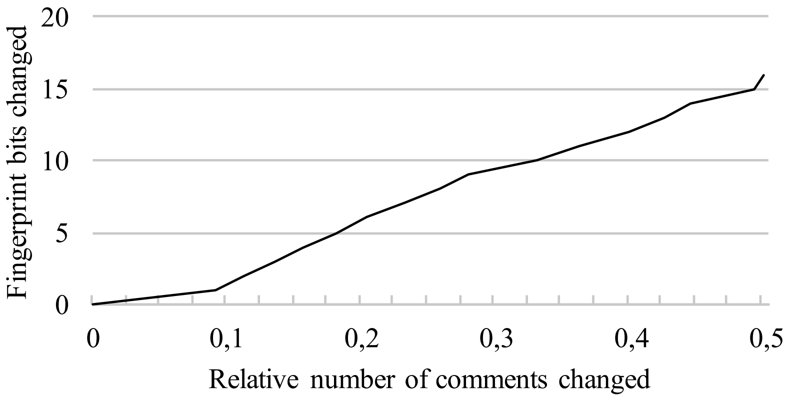
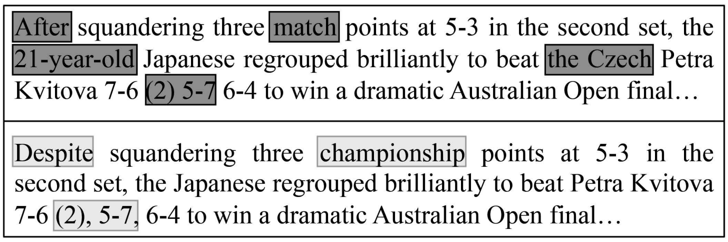


Figure 3.7: We simulate the deletion and addition of comments in an article's discussion and measure how the difference of fingerprints increases. Due to a linear correlation, the relative number of comment changes can be estimated based on the number of fingerprint bits changed.



After squandering three match points at 5-3 in the second set, the 21-year-old Japanese regrouped brilliantly to beat the Czech Petra Kvitova 7-6 (2) 5-7 6-4 to win a dramatic Australian Open final…

Despite squandering three championship points at 5-3 in the second set, the Japanese regrouped brilliantly to beat Petra Kvitova 7-6 (2), 5-7, 6-4 to win a dramatic Australian Open final…

Figure 3.8: An excerpt of an article on sports scraped right after publication (top) and a few hours later (bottom) exemplifies that only minor stylistic, changes are made.[20]

changed. For example, if fewer than five fingerprint bits change, we assume that fewer than 20 percent of the comments have changed.

News Articles. We study a second use case besides reader comments, which are news articles. We consider experiments on a dataset of news articles (partially) reproducible if the articles' texts change only slightly or not at all. Table 3.4 shows that the large majority of articles can be re-scraped from the same URL even after one year. Typical changes of an article are not full deletions but text corrections and event updates, which happen within a short time after publication. To analyze article text changes within a shorter time period, we crawled news articles published on a particular day and re-scrape the same articles two days later. Out of 147 articles from The Guardian, the texts of 132 articles (90 percent) remained unchanged within this time period. Based on fingerprint comparison, we correctly identify all of these articles, but also misclassify two additional articles as unchanged. However, only minor stylistic changes are made, which do not significantly alter their meaning. Figure 3.8 exemplifies how an article text changed over time.

[20] `www.gu.com/sport/2019/jan/26/naomi-osaka-wins-australian-open-final-petra-kvitova-grand-slams-tennis`

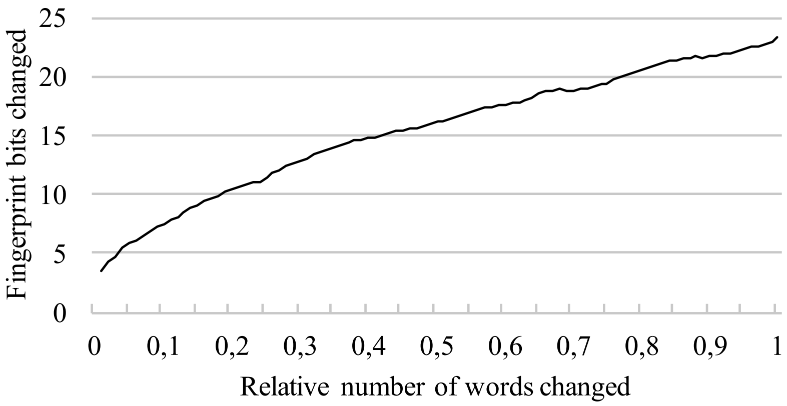


Figure 3.9: We simulate the deletion and addition of words in an article's text and measure how the difference of fingerprints increases. Due to a linear correlation, the relative number of changed words can be estimated based on the number of the fingerprint bits changed.

The second experiment on news articles simulates changes of an article text iteratively. We hypothesize that the fingerprint distance of the original text and the altered text is a good estimation of the texts' similarity. We further hypothesize that the measured distance can be used to estimate the number of changed words. We test these hypotheses with this experiment. To alter an article's text in a way that keeps a realistic language use, we blend it with the text from another article. With increasing probability, we replace a word of article A with the word of article B at the same position. For example, the first word of A is replaced with the first word of B with a low probability. Each iteration increases this probability until, at the end of the process, all words are replaced with a probability of 1. Figure 3.9 visualizes the relation between the relative number of changed words and the number of fingerprint bits changed. The plot corresponds to the average of ten randomly selected pairs of articles with ten random simulations each. The linear relation allows estimating the number of changed words based on fingerprint distance. For example, if the fingerprints of two scraped versions of an article differ in five bits, we assume that about five percent of the words have changed. The experiment hence confirms that the fingerprint distance is a good estimation of the similarity of two article texts.

3.3.4 Discussion

Our process enables scientists to re-use datasets even if their distribution is restricted. However, reproducibility, by definition, relies on the exact same conditions for re-running an experiment. Our experiments indicate that most comments and articles remain unchanged, but not all of them. In accordance with the definition of reproducibility, the identification of unchanged subsets is, strictly speaking, not enough. However, *partial reproducibility* can be ensured by our process. At the borderline of reproducibility and replicability, our process tries to reproduce the experiment setup regarding the dataset as precisely as possible. Thus, we aim for reproducibility, but if the exact setup cannot be reproduced, we make it as similar as possible. Assuming that the reproduced data subset is a representative sample of the full set, re-running an experiment only on the former is sufficient and justifiable. A similar assumption underlies training, validation, and test data splits in machine learning in general. In the context of dynamic comment datasets, the subset is presumably not a perfectly random sample of the full set. However, there is no reason to assume a systematic bias either.

A limitation of our study is that we looked at the dataset aspect of reproducibility and neglected software, algorithms, theorems, and proofs. All aspects need to be taken into account to make a complete experiment reproducible. In addition to publishing scrapers and fingerprints, a thorough description of how the data was selected is required. This information is needed to rule out any sampling bias and understand whether the data is useful for other experiments. A list of URLs to scrape defines the data selection within our proposed process. The question is, how is this list compiled? Is there some potential bias in this compilation? In our example with online news comments, we crawled all available news articles and their comments from selected platforms at a fixed point in time. While the point in time was chosen arbitrarily, the selection of platforms might introduce some bias, which we are, however, unaware of.

The impact on web content providers must not be neglected. On the one hand, scraping the Web supplies researchers with precious datasets. On the other hand, it increases the load for platform providers. The need of users and platform providers for an option to delete data and the need of researchers to reproduce an experiment on the same data are a trade-off. For example, toxic comment classification focuses on comments that are removed by discussion moderators. Once removed, there is no way to re-scrape them. Our approach supports this option for platform providers, and therefore, experiments on removed (toxic) comments cannot be reproduced.

Sometimes datasets are manually labeled after scraping them. These labels need to be distributed together with the scraping component. To match these labels to the re-scraped data records, the mapping of labels to comments needs to be documented. This documentation could consist of pairs of identifiers and labels. In the example use case, comments might not have unique identifiers but only the full news article. In this case, a fingerprint of a comment itself might be used as an identifier, although there is no guarantee that there is a unique match.

3.4 Summary

This chapter introduced comment datasets that we collected in preparation for experiments in the following chapters and addressed the challenges of dataset reproducibility. The novelty of these datasets lies in the large number of comments, the long covered time span, the labels provided by professional moderators, and the rich metadata. They are the basis for extensive analyses and enable the training of word embeddings specific to the language used in online discussions. However, the dataset aspect of experiment reproducibility is especially difficult to establish for comment datasets because of legal restrictions, ethical concerns, and the variability of web data in general. Therefore, we designed, implemented, and evaluated a process for measuring and facilitating dataset reproducibility. The results showed the effectiveness of this process, demonstrating that other researchers are enabled to re-run experiments on the exact or almost same data without the need to publish the dataset itself.

4

Classifying Toxic Comments

The application of deep learning methods to toxic comment classification on online news platforms is challenging for mainly two reasons: limited training data and high demands on the classifier's integration into the moderation process. First, as illustrated in Chapter 2, most comment datasets contain only a few thousand labeled samples, while standard neural network architectures have millions of parameters. This discrepancy makes training neural models without overfitting issues nearly infeasible. Second, classifiers are not only required to achieve competitive accuracy but also to provide explanations for their results. Incomprehensible classifiers lack the trust and acceptance of the moderators and readers. Thus, they cannot support the moderation process in real-world application scenarios. We address and solve both problems in the following.

This chapter is divided into six sections. First, Section 4.1 presents fine-grained subclasses of toxicity, current approaches for toxic comment classification, and an analysis of the shortcomings of these approaches. Section 4.2 then describes a data augmentation method and a neural network architecture tailored to small training datasets. Extending the work on small datasets, Section 4.3 introduces an ensembling approach for transformer-based models. The next two sections focus on challenges that arise from applying toxic comment classification in practice: Section 4.4 compares four methods to explain classification results and Section 4.5 presents a real-world application scenario for semi-automatic comment moderation. Finally, Section 4.6 summarizes this chapter.

4.1 Comparative Study

We begin with an overview of subclasses of toxicity and then compare baseline methods and state-of-the-art approaches for toxic comment classification, followed by combining them in an ensemble model. This ensemble reduces misclassifications by selecting the most suitable basic model for each comment individually, e.g., a character-based model instead of a word-based model, if none of the comment's words are in the vocabulary. Our ensembling method falls into the category of stacking because it combines heterogeneous models. By analyzing the ensemble's false negatives and false positives, we gain insight into open challenges that all of the approaches share.

4.1.1 Classes of Toxicity

Inspired by the Kaggle challenge on toxic comment classification, we consider five classes of toxicity and give examples for illustration.[1,2]

Profanity: "That guideline is bullshit and should be ignored." The first class considers swear or curse words. In the example, the single word "bullshit" comprises the toxicity of this comment. There is no need to take into account the full comment if at least one profane word has been found. For this reason, simple lists of profane words can be used for detection. To counter these lists, malicious users often post variations or misspellings of such words.

Insults: "Do you know you come across as a giant prick?" While the previous class of comments does not include statements about individuals or groups, the class *insults* does. It includes rude or offensive statements that concern an individual or a group. In the example, the comment directly addresses another user, which is common but not necessary.

Threats: "I will arrange to have your life terminated." Severely toxic comments are threats against the life of another user or the user's family. Statements that announce or advocate for inflicting punishment, pain, injury, or damage on oneself or others fall into this class. A common threat in online discussions is to have another user's account closed.

Identity Hate: "Mate, sound like you are jewish. Gayness is in the air." In contrast to insults, identity hate targets groups defined by religion, sexual orientation, ethnicity, gender, or other social identifiers. Negative attributes are ascribed to the group as if these attributes were universally valid. For example, racist, homophobic, and misogynistic comments fall into this class.

Otherwise Toxic: "Bye! Don't look, come or think of coming back!" Comments that do not fall into one of the previous four classes but still make other users leave a discussion are considered *toxic* without further specification. Trolling, e.g., by posting off-topic comments to disturb the discussion, falls into this class. Similarly, an online discussion filled with spam messages would quickly become abandoned by users. However, spam detection is not the focus of toxic comment classification.

The listed classes are not mutually exclusive. Comment classification problems are sometimes modeled as multi-class classification and sometimes as multi-label classification. Multi-class means that different labels are mutually exclusive, e.g., a comment can be either an insult or a threat. In contrast, multi-label means that a comment can have multiple labels at the same time. Multi-label classification better mirrors real-world applications because a comment can be, for example, both an insult and a threat. In research, this problem is often slightly simplified by assuming analyzed classes are mutually exclusive.

[1] `www.kaggle.com/c/jigsaw-toxic-comment-classification-challenge`

[2] Warning: The remainder of this chapter contains comment examples that may be regarded as profane, vulgar, or offensive. These comments do not reflect the views of the author and exclusively serve to explain linguistic patterns. The following examples stem from a dataset of annotated Wikipedia article page comments and user page comments [209], which is publicly available under CC BY-SA 3.0.

4.1.2 Classification Models and Ensemble Learning

We compare six different approaches, which are described in this subsection: logistic regression, an LSTM model, a bidirectional LSTM model, a bidirectional GRU model, a bidirectional GRU model with an additional attention layer, and a CNN model.

The logistic regression algorithm (LR) is widely used in combination with manual feature engineering for binary classification tasks. Contrary to deep learning models, it allows obtaining insights about the model by observing the coefficients learned for each individual feature. Waseem and Hovy [205] showed that word and character n-grams are among the most indicative features for the task of hate speech detection. For this reason, we investigate the use of word and character n-grams combined with logistic regression models.

Our LSTM model takes a sequence of words as input. An embedding layer transforms one-hot-encoded words into dense vector representations, and a spatial dropout, which randomly masks 10 percent of the input words, makes the network more robust. The embedding layer uses either GloVe [132] or FastText [28] embeddings. The GloVe word embeddings were pre-trained on a large Twitter corpus by Pennington et al. [132]. In contrast, we pre-trained the FastText sub-word embeddings on another dataset of 95 million comments on Wikipedia user talk pages and article talk pages.[3] To this end, we applied the skip-gram method with a context window size of 5 and trained for five epochs. To process the sequence of word embeddings in our neural network model, we use an LSTM layer with 128 units, followed by a dropout of 10 percent. Finally, a dense layer with a sigmoid activation makes the prediction for the multi-label classification, and a dense layer with softmax activation makes the prediction for the multi-class classification.

In contrast to the standard LSTM model, the bidirectional model uses two LSTM layers that process the input sequence in opposite directions. Thereby, the input sequence is processed with the correct and reverse order of words. The outputs of these two layers are averaged. As a result, this network architecture supports recognizing long-range dependencies in the input sequence of words. Similar to the bidirectional LSTM model, we use a bidirectional GRU model consisting of two stacked GRU layers. We use layers with 64 units so that the number of parameters of each unidirectional model and its bidirectional counterpart is the same. All other parts of the network architecture are inherited from our standard LSTM model. As an extension of our bidirectional GRU model, we add an attention layer following the work by Yang et al. [211]. According to Gao and Huang [66], attention mechanisms can help to detect toxic words or phrases in long comments. The architecture of our CNN model is comparable to the approach by Kim [92].

Each classification model varies in its predictive power and has specific weaknesses that are other model's strengths. For example, GRUs and LSTMs are powerful in capturing phrases but miss long-range dependencies for very long sentences with 50 or more words. Bidirectional LSTMs (Bi-LSTMs) and attention-based networks can compensate these errors to a certain extent. Sub-word embeddings can handle even misspelled or obfuscated words. We build an ensemble model that determines which of the single classifiers is most powerful on a specific kind of comment. The ensemble extracts features from comments and learns an optimal classifier selection for a given feature combination.

[3] www.figshare.com/articles/Wikipedia_Talk_Corpus/4264973

The features comprise: (1) the comment length (number of characters), (2) the relative number of uppercase characters, (3) the relative number of non-alphabetical characters, (4) the relative number of exclamation marks, and (5) the relative number of words that have a GloVe embedding. We include the latter feature to measure how many uncommon words are used. After the feature extraction, we perform a 5-fold cross-validation. The set of out-of-fold predictions from the various approaches then serves to train an ensemble with gradient boosting decision trees. We average the final predictions on the test set across the five trained models.

4.1.3 Experiments

Our hypothesis is that the ensemble learns a combination of classifiers that outperforms each individual classifier because they have different strengths and weaknesses. We expect that the individual classifiers have comparable performance, and none outperforms the others significantly. This homogeneity is important because otherwise, the ensemble would always prioritize the outperforming classifier.

We compare the models on two datasets: 150,000 comments from Wikipedia talk pages presented by Google Jigsaw in context of the Kaggle challenge on toxic comment classification and 25,000 posts from Twitter collected by Davidson et al. [50]. Both datasets are de-facto standards for research on toxic comment classification because of the easy access to the data and their relatively large size. Discussions on Wikipedia and news platforms are more similar than they may appear at first glance. Both exhibit well-defined discussion topics (either Wikipedia articles or news articles), and both refrain from underlying social network structures (no concept of followers or friendship connections between users). The two datasets include common difficulties of toxic comment datasets: They are labeled based on different definitions; they contain language use specific to user comments and tweets; and they present a multi-class and a multi-label classification task, respectively. We expect our ensemble to perform well on online comments and tweets despite their different language characteristics such as text length and use of slang words. To evaluate our hypotheses, we compare the six described methods and use the following setup. We combine the neural network approaches with two different word embeddings each and the logistic regression with character and word n-grams as features. For the multi-label classification (Wikipedia dataset), we measure macro-average precision and recall for each class separately and average their results to get the F1-score per classifier. For the multi-class classification (Twitter dataset), the F1-score per classifier can be obtained without any special adjustments. We choose the macro-average F1 since it is more indicative than the micro-average F1 for strongly unbalanced datasets [219].

The ensemble can only outperform the individual models if they achieve comparable classification performance and make only weakly correlated predictions. Therefore, a second experiment measures the correlation of the different classifiers' predictions using the Pearson correlation coefficient. We consider a set of combinations, e.g., logistic regression combined with a neural network, and assess their potential for improving the overall prediction.

Table 4.1: Precision, recall, and F1-score on datasets from Wikipedia and Twitter.

	Wikipedia			Twitter		
Classification Model	P	R	F1	P	R	F1
CNN (FastText)	.73	.86	.78	.73	.83	.78
CNN (GloVe)	.70	.85	.75	.72	.82	.77
LSTM (FastText)	.71	.85	.75	.73	.83	.78
LSTM (GloVe)	**.74**	.84	.78	.74	.82	.78
Bi-LSTM (FastText)	.71	.86	.76	.72	.84	.78
Bi-LSTM (GloVe)	**.74**	.84	.78	.73	**.85**	.78
Bi-GRU (FastText)	.72	.86	.77	.72	.83	.77
Bi-GRU (GloVe)	.73	.85	.77	.76	.81	.78
Bi-GRU Attention (FastText)	**.74**	.87	.78	.74	.83	**.79**
Bi-GRU Attention (GloVe)	.73	.87	.78	**.77**	.82	**.79**
Logistic Regression (Char N-Grams)	**.74**	.84	.78	.73	.81	.76
Logistic Regression (Word N-Grams)	.70	.83	.75	.71	.80	.75
Ensemble	**.74**	**.88**	**.79**	.76	.83	**.79**

Experimental Results. The results in Table 4.1 show that the ensemble outperforms all individual classifiers, except for the bidirectional GRU network with an attention layer on the Twitter dataset. The ensemble outperforms this model on the Wikipedia dataset by approximately one percent F1-score.[4] We find that the difference in F1-score between the best individual classifiers and the ensemble is higher on the Wikipedia dataset than on the Twitter dataset. This finding is accompanied by the results in Table 4.2, which show that most classifier combinations present a high correlation on the Twitter dataset and are therefore less effective as an ensemble model. An explanation for this effect is that the text sequences within the Twitter dataset show less variance than those in the Wikipedia dataset. This reduced variance can be reasoned from (1) their sampling strategy based on a list of terms, (2) the smaller size of the Twitter dataset, and (3) less disparity among the three classes in the Twitter dataset than among the six classes in the Wikipedia dataset. With less variant data, one carefully selected classifier can be sufficient for a particular type of text.

Table 4.2 further shows that ensembling can be especially effective on the minority classes *threat* (Wikipedia) and *hate* (Twitter). The predictions for these two classes have the weakest correlation. This can be exploited when dealing with strongly imbalanced datasets, as often the case in toxic comment classification and related tasks. The table also indicates which classifiers make the most weakly correlated predictions and are therefore most suited to be combined. For example, word and character n-grams used by our logistic regression classifier make weakly correlated predictions. Similarly, the logistic regression model with character n-grams seems to be a good addition to neural network models. Contrary to that, we see that varying the word embeddings hardly influences the predictions and is therefore unsuited for ensemble learning.

[4] Our ensemble model also was among the top 2 percent of all 4551 submissions in the Kaggle challenge on toxic comment classification, confirming its competitiveness compared to a broad range of approaches.

Table 4.2: Pearson correlation of the classifiers' predictions (G: GloVe, FT: FastText).

Dataset	Class	F1		Pearson
		Different Word Embeddings		
		GRU+G	GRU+FT	
Wikipedia	Average	.78	.78	.95
	Threat	.70	.69	.92
Twitter	Average	.79	.79	.96
	Hate	.53	.54	.94
		CNN+G	CNN+FT	
Wikipedia	Average	.75	.78	.91
	Threat	.67	.73	.82
Twitter	Average	.77	.78	.94
	Hate	.49	.53	.90
		Different NN Architectures		
		CNN	BiGRU Att	
Wikipedia	Average	.78	.78	.85
	Threat	.73	.71	.65
Twitter	Average	.78	.79	.96
	Hate	.50	.49	.93
		NN and Logistic Regression		
		CNN	LR Char	
Wikipedia	Average	.78	.78	.86
	Threat	.73	.74	.78
Twitter	Average	.78	.76	.92
	hate	.50	.51	.86
		BiGRU Att	LR Char	
Wikipedia	Average	.78	.78	.84
	Threat	.71	.74	.67
Twitter	Average	.79	.76	.92
	Hate	.49	.51	.88
		Character and Word N-Grams		
		LR Word	LR Char	
Wikipedia	Average	.75	.78	.83
	Threat	.70	.74	.69
Twitter	Average	.75	.77	.94
	Hate	.50	.51	.91

4.1.4 Error Analysis

We performed an extensive error analysis on the classification results of the ensemble to identify remaining challenges. To this end, we sorted false positives (non-toxic comments that are misclassified as toxic) and false negatives (toxic comments that are misclassified as non-toxic) into five common error classes. We considered the class *toxic* of the Wikipedia dataset and the class *hate* of the Twitter dataset. Both classes are of high importance for the task of comment moderation. Our ensemble resulted in 1,794 false negatives and 1,581 false positives for the Wikipedia dataset. We randomly selected 200 samples out of each set for our analysis. On the smaller Twitter dataset, there are 55 false negatives and 58 false positives, and we analyzed all of these samples. The following examples are Wikipedia talk page and user page comments.

Comments without swear words can convey a toxic meaning that is hard to be detected by machine-learned models. This toxic meaning is only revealed with the help of context knowledge and understanding the full sentence, as exemplified by the toxic comment: “she looks like a horse”. The word “horse” is not insulting in general. To understand the toxicity of the comment, a model needs to understand that “she” refers to a person and that “looking like a horse” is, in that case, considered insulting. However, this insult is not revealed by looking at the words of the sentence independently. In contrast to these false negatives, there are false positives that contain toxic words, although they are overall non-toxic. If a user posts a self-referencing comment, annotators rarely consider these comments toxic, for example: “Oh, I feel like such an asshole now. Sorry, bud.”. However, the learned model focuses on the mentioned swear words, which triggers the misclassification. Taking into account a full sentence and getting its meaning still remains a challenge for deep learning approaches.

State-of-the-art models cannot take into account *references to other comments* in the discussion, or interpret *metaphors and comparisons.* Therefore, examples of false positives are otherwise non-toxic comments that cite toxic comments. Because of the toxic citation, the overall comment can be misclassified as toxic. Example: “I deleted the Jews are dumb comment.” An example of false negatives is the comment: “Who are you a sockpuppet for?”. The word sockpuppet is not toxic in itself. However, the accusation that another user is a sockpuppet attacks the user without addressing their comment — a so-called *ad hominem* argument.

Sarcasm, irony, and rhetorical questions have in common that their interpretation differs from their literal meaning. This disguise can cause false negatives in the classification. While they are not the focus of this thesis, we at least give examples for this reported problem for toxic comment classification [124, 139]. Example comment: “hope you’re proud of yourself. Another milestone in idiocy.”. If the first sentence in this example is taken literally, there is nothing toxic about the comment. However, the user who posted the comment actually means the opposite, which is revealed by the second sentence. Other examples are rhetorical questions, which do not ask for real answers. Example: “have you no brain?!?!”. This comment is an insult because it alleges another user to act without thinking. Rhetorical questions in toxic comments often contain subtle accusations, which current approaches hardly detect.

The annotation of toxic comments is a challenging task and occasionally leads to *mislabeled comments.* Annotation guidelines cannot consider each and every edge case. For example, a comment that criticizes and therefore cites a toxic comment is not necessarily toxic itself. Example: “No matter how upset you may be there is never a reason to refer to another editor as ‘an idiot’ ”. State-of-the-art approaches classify this comment as non-toxic, although it is labeled as toxic. We argue that this comment is actually not toxic. Thus, this false negative is not a misclassification by the current models but rather a mislabeling by the annotators. Similar to false negatives, there are false positives caused by wrong annotations. Ill-prepared annotators, unclear task definition, and the inherent ambiguity of language may cause a minority of comments in training, validation, and test dataset to be labeled wrongly. An example is the following comment, which was falsely labeled as non-toxic by the annotators: “IF YOU LOOK THIS UP UR A DUMB RUSSIAN”.

Obfuscated words, typos, slang, abbreviations, and neologisms are a particular challenge in toxic comment datasets. If there are not enough samples containing these words in the training data, the learned representations, e.g., word embeddings, may not account for the words’ true meaning. Thus, wrong representations may cause misclassifications. Example: “fucc nicca yu pose to be pullin up”. Similarly, the classification of the comment: “WTF man. Dan Whyte is Scottish” depends on the understanding of the term “WTF”. The amount of slang used is platform-specific. We found that misclassifications due to rare words are twice as high for tweets than for Wikipedia talk page comments.

4.2 Training Deep Neural Networks on Limited Data

The previous section demonstrated the strength of ensemble learning for toxic comment classification on large datasets. However, most comment datasets are only a tenth or a hundredth in size. Therefore, this section is dedicated to training deep learning models on such limited datasets and studies whether ensemble models outperform single models also in this context.

4.2.1 Methodology and Data

Our method is based on two main ideas: (1) increasing the amount of available training data by data augmentation and (2) leveraging the larger amount of training data for deep learning. Besides this deep learning approach, we propose three other models and combine all four models in an ensemble. Figure 4.1 is a system overview, which shows how this combination is implemented.

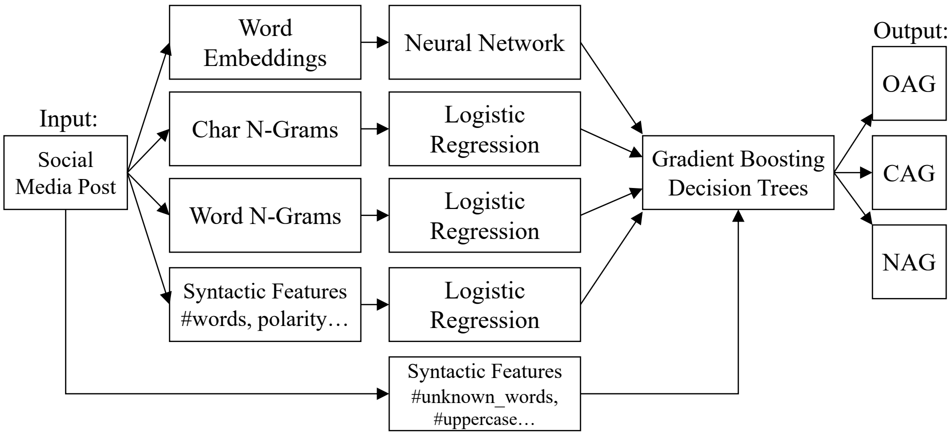


Figure 4.1: Given a social media post, we apply four different models and combine their predictions in an ensemble to identify the aggression level of the post: overtly aggressive (OAG), covertly aggressive (CAG), or non-aggressive (NAG).

Our data augmentation method is based on the following insight: Machine translating a user comment into a foreign language and then translating it back to the initial language preserves its meaning but results in different wording.[5] This change in wording is essential for our approach: If the translation did not change the wording, it could not augment our dataset because the dataset already contains the exact same comment. However, because the wording is different, the translated comment adds to our dataset. Only if the meaning is preserved, we can assume that the toxicity label of the initial comment also holds for the translated comment. Thanks to the recent advances in neural machine translation and its continuously improving accuracy, we can assume that machine translation generally preserves the meaning. We give two examples for the data augmentation. The first example shows that different translations use different words, such as "loot", "spoils", and "prey".

1. Initial English post: "Loot of people's mandate, is it democracy?"
2. English to French to English: "Is the spoils of the people democracy?"
3. English to German to English: "Prey of the People's Mandate, is it Democracy?"
4. English to Spanish to English: "The spoils of the spoils of people, is it democracy?"

The second example is a post whose translations are less diverse. However, for example, the abbreviation "'u" for the word "you" is resolved.

1. Initial English post: "AAP dont need the monsters like u"[6]
2. English to French to English: "AAP does not need monsters like you"
3. English to German to English: "AAP does not need the monsters like you"
4. English to Spanish to English: "AAP does not need monsters like you"

We applied this data augmentation method to a dataset of 15,000 English Facebook posts and 15,000 Hindi Facebook posts introduced by Kumar et al. [100] in the context of a shared task on aggression identification that they organized. For English posts, we used three intermediate languages (French, German, Spanish) but only one intermediate language (English) for Hindi posts. Each Hindi post was machine-translated into English and afterward translated back to Hindi. However, for Hindi posts, this method did not work as well. Often the intermediate step of translating to English already failed in preserving the meaning of the initial Hindi post. Consequently, the meaning of the translated posts did not match with the initial labels, and the translated posts could not be used for training. Presumably, the quality of machine translations from Hindi to other languages is worse due to a lower amount of training data. Our augmented dataset contains the 15,000 initial English posts and 45,000 translated versions and is published online.[7] Each post is labeled with one of three aggression levels: overtly aggressive (OAG), covertly aggressive (CAG), or non-aggressive (NAG). Covertly aggressive

[5] The same technique was also used by several teams in the Kaggle challenge on toxic comment classification: `www.kaggle.com/c/jigsaw-toxic-comment-classification-challenge/discussion/52557`

[6] AAP is an Indian political party.

[7] `www.hpi.de/naumann/projects/repeatability/text-mining.html`

posts include indirect attacks that use, e.g., satire or rhetorical questions, while overtly aggressive posts contain lexical features that are considered aggressive [100].

Social media posts are similar to reader comments on online news platforms, but one difference is the use of hashtags and user mentions. To this end, we implemented a special tokenization method for hashtags and user mentions. Many hashtags in the dataset are concatenations of multiple words. For example, the meaning of *#realsurgicalstrike, #deathtoPakistan,* and *#saysiamaproudchutiyawhodoentknowshitabouthistory* can only be understood if the words are split correctly. In some cases, the full post contains only a hashtag but no other content, such as the post *#THANKY0UTAKER,* which is labeled as covertly aggressive. We propose to split the strings after # and @ symbols into their original words with a dynamic programming approach. Our assumption is that the best splitting is the one that maximizes the product of each word's individual probability of occurrence. For example, the splitting *real surgical strike* is to prefer over the splitting *real surgicals trike,* because the probability of *surgical* is higher than the probability of *surgicals* and the probability of *strike* is higher than the probability of *trike.* A word's probability of occurrence can be inferred from a large corpus of natural language, e.g., Wikipedia. As the final pre-processing step, all posts containing more than 150 tokens are truncated.

Figure 4.2 visualizes our recurrent neural network architecture, which is based on a bidirectional GRU layer. We use pre-trained, 300-dimensional FastText embeddings [28], more specifically, common crawl embeddings[8] for the English dataset and Hindi Wikipedia embeddings[9] for the Hindi dataset [74]. The word embeddings serve as the input to a spatial dropout, which blocks the embeddings of 10 percent randomly chosen input words. A bi-directional layer of 64 GRUs processes the remaining 90 percent of the input. The next layer performs global average pooling and global k-max-pooling independently on the sequence of the outputs of all GRUs. A k-max-pooling with $k = 2$ extracts not only the largest, but also the second-largest element of the previous layer. A Lambda layer implements this non-standard pooling technique. The average, the maximum, and the second-largest element are concatenated into one vector, and a dropout of 10 percent is added to counter overfitting. Finally, a dense layer with softmax activation outputs three class probabilities to match the three mutually exclusive labels: OAG, CAG, and NAG. The model is trained for two epochs with a batch size of 32. We observe that the data augmentation slightly reduces the number of epochs until the validation loss increases and overfitting starts.

For a tf-idf-based approach, we extract character n-grams of length 2 to 6 and limit the set to the 50,000 most frequent character n-grams. Further, we extract word n-grams of length 1 to 2 and filter English stopwords, but also all words that occur in more than 50 percent of all documents or in less than two documents. We normalize the frequency of occurrence of all n-grams using tf-idf. As the classifier, we choose logistic regression for both word and character n-grams. Based on the extracted n-grams, we train logistic regression models according to the one-vs.-rest strategy: We train one classifier per aggression level. For each aggression level, all posts of that level are positive training samples, and all other posts are negative training samples.

[8] `www.fasttext.cc/docs/en/english-vectors.html`

[9] `www.fasttext.cc/docs/en/crawl-vectors.html`

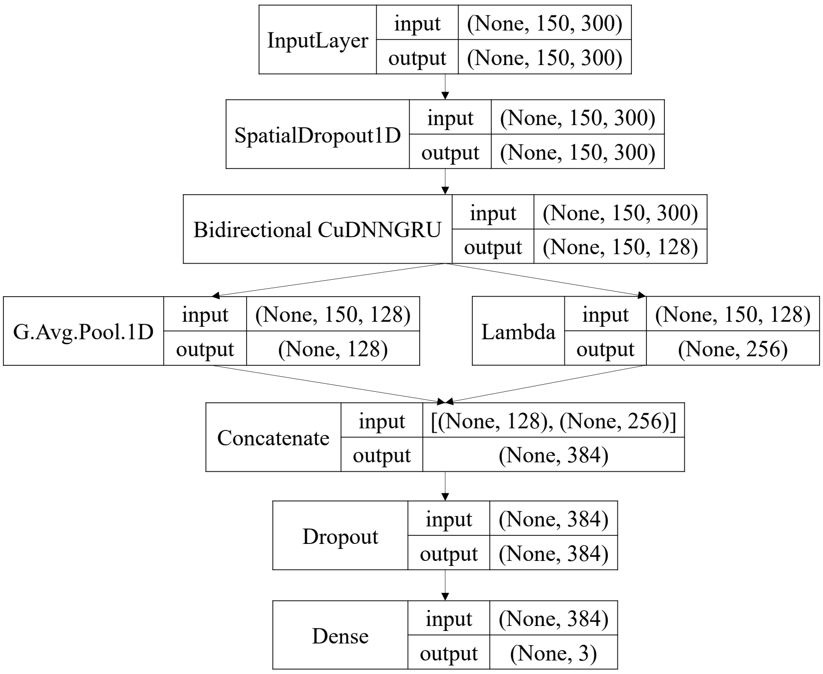


Figure 4.2: The main buildings blocks of the neural network architecture are gated recurrent units (GRUs), average pooling, and k-max-pooling (Lambda layer).

Our set of hand-picked features captures various properties, such as punctuation and capitalization, but also emoticons. Overall, a combination of 35 extracted features serves as input to three logistic regression models, which are also trained according to the one-vs.-rest strategy for each level of aggression. To capture emoticons, 25 of these features comprise regular expressions for sad, happy, and neutral faces. The remaining 10 features capture, for example, the number of words, the proportion of uppercase characters to lowercase characters, the number of negation words, and also the polarity of the post. We apply the VADER sentiment analyzer to extract polarity scores [69].

Word embeddings, word n-grams, character n-grams, and hand-picked features capture different properties of user posts and therefore have different strengths and weaknesses similar to the models in Section 4.1. For example, word n-grams suffer from out-of-vocabulary problems, which makes them sensitive to obfuscated words. The dataset used in this section contains posts that make extensive use of obfuscation, such as "Son of a B****", "***k them!!!!". Word embeddings and word n-grams cannot capture the meaning of these obfuscated posts, but character n-grams or the number of asterisks and exclamation marks as hand-picked features can. For the four models, we analyze the pairwise Pearson correlation of their predictions as listed in Table 4.3. The word n-gram and the character n-gram models have the highest correlation. In contrast, the recurrent neural network and the word n-gram model have a relatively low correlation. Their low correlation motivates combining their predictions in an ensemble model because we can assume that they complement each other well. If they both have a similarly high F1-score, their combination outperforms the single models.

For each model, we run 10-fold cross-validation and create out-of-fold predictions. For each of the 10 runs, we also make predictions for the test set and average all 10

Table 4.3: Pearson correlation of the different models (LR word: word n-grams, LR char: character n-grams, LR features: hand-picked features).

Class	RNN+ LR Word	RNN+ LR Char	RNN+ LR Features	LR Word+ LR Char	LR Word+ LR Features	LR Char+ LR Features
OAG	.73	.75	.45	.84	.41	.42
NAG	.81	.82	.42	.88	.40	.42
CAG	.62	.65	.18	.77	.15	.21

Table 4.4: F1-score of 10-fold cross-validation on English Facebook posts (En FB) with or without augmentation (augm.) and Hindi Facebook posts (Hi FB).

System	En FB	En FB augm.	Hi FB
Random Baseline	.33	.34	.34
RNN	.57	.58	.54
Word N-Grams	.58	.58	.59
Char N-Grams	.58	.58	.61
Feature Selection	.40	.39	.37
Ensemble	**.61**	**.61**	**.63**

predictions per model. We use the out-of-fold predictions to learn which combination of the single models performs best. Instead of a simple weighted average of the different models, we follow a stacking approach as in Section 4.1. Given a comment, we extract features and, based on them, decide how to weight the different models' predictions for this particular comment. The stacking uses gradient boosting trees, more precisely, 75 trees with a depth of 3, a bagging fraction of 0.8, and a feature fraction of 0.45.

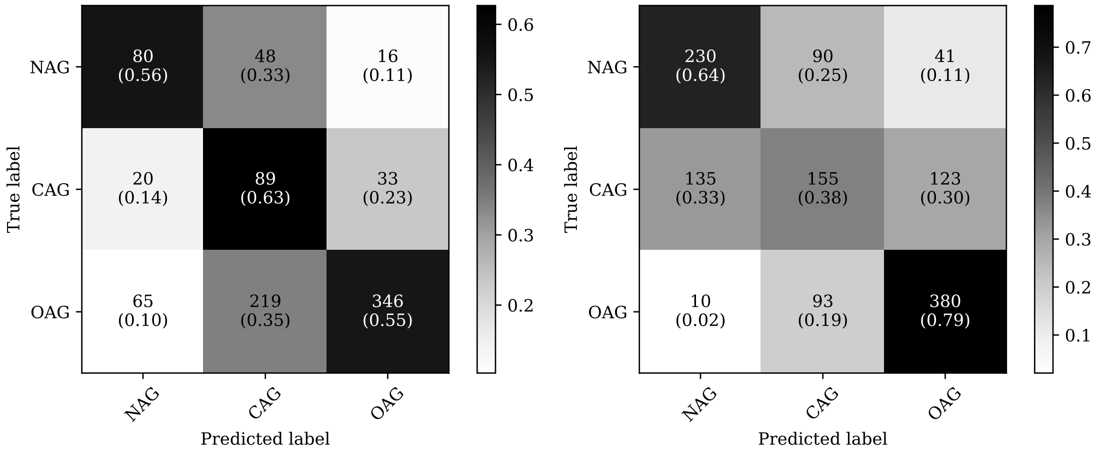
4.2.2 Experiments

Table 4.4 lists our cross-validation results. The RNN and the logistic regression models with word n-grams or character n-grams perform equally well on the English data. The data augmentation makes only a small difference overall. However, it improves the F1-score of the RNN from 57.2 to 58.4 percent. On the Hindi data, character n-grams clearly outperform all other models. We assume that the performance of the RNN could be improved with better word embeddings, such as embeddings trained on Hindi social media posts. The hand-picked feature selection is inferior to all other models except for a random baseline.

Table 4.5 lists our test set results. Our results on both English test sets are the most stable results across all teams participating in the shared task: We achieve an F1-score of 60.0 percent on both datasets, Facebook and Twitter. These results show that our approach does not suffer from overfitting to the training dataset and generalizes well to other datasets. Our approach achieves rank 6 out of 30 on the Facebook dataset with an F1-score of 60.0 percent (F1-score of the top team: 64.2 percent). On the Twitter dataset, it achieves rank 2 out of 30 with an F1-score of 60.0 percent (F1-score of the top team: 60.1 percent).

Table 4.5: F1-score on the English (En) and Hindi (Hi) test datasets of Facebook (FB) and Twitter (TW) posts.

System	En FB	En TW	Hi FB	Hi TW
Random Baseline	.35	.35	.36	.32
Ensemble	**.60**	**.60**	**.63**	**.38**



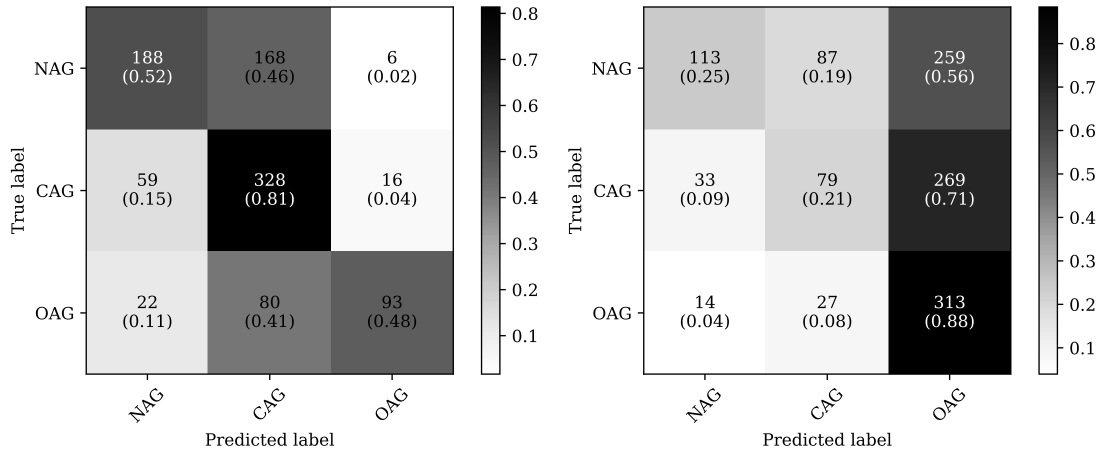
(a) Facebook

(b) Twitter

Figure 4.3: Confusion matrices for the ensemble model on the English test datasets.

Our results on the Hindi test sets differ for the Facebook dataset and the Twitter dataset: Our approach achieves rank 4 out of 15 on the Facebook dataset with an F1-score of 63.1 percent (F1-score of the top team: 64.5 percent). On the Twitter dataset, it achieves rank 8 out of 15 with an F1-score of 38.3 percent (F1-score of the top team: 49.9 percent). The F1-scores of each team differ between the Facebook dataset and the Twitter dataset by 13.1 percent on average. Therefore, we assume that the differences in classification performance are inherent to the datasets.

Figure 4.3 shows the confusion matrices of the ensemble model on the English test datasets. The model works equally well for all three classes: overtly aggressive, covertly aggressive, and non-aggressive. As to expect, the non-aggressive class is more often confused with the covertly aggressive class than the overtly aggressive class. Similarly, the overtly aggressive class is more often confused with the covertly aggressive class than the non-aggressive class. While on the English Twitter dataset the classifier works well for non-aggressive posts and for overtly aggressive posts, it is only slightly better than a random baseline for covertly aggressive posts. Covertly aggressive posts are often misclassified as either non-aggressive or overtly aggressive. Figure 4.4 shows the confusion matrices of the ensemble model on the Hindi test datasets. It is difficult for the classifier to distinguish overtly aggressive from covertly aggressive Facebook posts and covertly aggressive from non-aggressive ones. For the Twitter dataset, the majority of the posts are misclassified as non-aggressive. The model was not able to generalize from the Facebook training data to the Twitter test data. However, in general, we found that ensemble models are suitable for toxic comment classification.



(a) Facebook

(b) Twitter

Figure 4.4: Confusion matrices for the ensemble model on the Hindi test datasets.

4.3 Transformer-Based Ensemble Models

The current trend for research on natural language processing with deep neural networks is to develop more and more complex models. The complexity is expressed in the number of parameters, which is in the hundreds of millions for transformer-based language models, such as bidirectional encoder representations from transformers (BERT) [52]. More precisely, *large* BERT models span 24 layers and 340 million parameters, and even *base* BERT models span 12 layers and 110 million parameters. As described in Subsection 2.1.3, these models are pre-trained on large corpora, e.g., collections of web pages with billions of tokens. For down-stream tasks, including text classification, they are fine-tuned on smaller datasets. While the pre-training is unsupervised, the fine-tuning for down-stream tasks is typically supervised learning. In our initial experiments, we found that the fine-tuning step fits BERT models well to labeled toxic comment datasets, and the models' bias is typically low. The strong classification performance across training, validation, and test datasets indicates that the models do not suffer from underfitting. In fact, overfitting can be more of an issue, especially for smaller datasets. The number of parameters is much larger than the number of samples in hand-labeled datasets. Standard regularization techniques, such as dropout and early stopping to limit the number of training steps, can be used to cope with overfitting problems. However, the model's variance still remains high. Even slight variations in the input data or a slight change of the random seed result in large changes in classification performance. The random seed affects, for example, the randomly initialized weights of the final prediction layer (prediction head). In our initial experiments, we found that the classification performance varies in a range of up to five percentage points in F1-score. We address the issue of high variance of fine-tuned BERT models on small datasets with an ensembling approach. To this end, we propose to combine the predictions of multiple BERT models that are trained with bootstrap aggregating on slightly differing training datasets and with varying weight initialization in the final prediction layer. Our experiments show that an ensemble achieves a two percentage points higher F1-score than single models.

Further, we optimize the number of ensembled models and find that the performance increases for up to 15 models and remains constant for larger ensembles.

4.3.1 Bootstrap Aggregating BERT Models

This subsection presents our ensembling approach. It begins with a brief introduction of the dataset and further describes the classification model, the training procedure, and the ensembling strategy.

The data used for the experiments in this section originates from a shared task organized by Kumar et al. [101] and comprises three datasets: an English, a Hindi, and a Bangla dataset of 2000 to 4000 social media posts each [101]. There are two independent tasks. The first task, task A: aggression identification, is a three-way classification into non-aggressive (NAG), covertly aggressive (CAG), and overtly aggressive (OAG) posts. Another task with the same classes was described in Section 4.2. The second task, task B: misogynistic aggression identification, is a binary classification task with two labels: gendered (GEN) and non-gendered (NGEN). Gendered aggression in this dataset is defined as attacks based on gender (roles) and includes homophobic and transgender attacks [100]. Figure 4.5a and Figure 4.5b list one English example post per class label.

The tokenizer for BERT uses word pieces so that the model learns an embedding for each subword token. The vocabulary consists of 30,000 tokens. Custom tokens can be added to extend this vocabulary, but then there is no pre-trained representation for these tokens. A larger dataset than the one provided for this task would be needed to make proper use of custom tokens. We refrain from any complex data pre-processing and use only three small steps. First, all characters are converted to lowercase. Second, we insert whitespaces before and after every emoji so that they can be tokenized as separate tokens. Third, we limit the sequence length to 200 tokens. The sequence length defines how many tokens are cut off from overly long sequences. Only a few posts are affected by this choice. With a maximum sequence length of 200 tokens, 0.9 percent of all training samples are affected. A maximum sequence length of 220 or 230 tokens reduces this number to 0.5 percent. The tokenizer is the same as used for pre-training the BERT model. For this reason, emojis and non-Latin characters are unknown tokens, which are replaced with a common `[UNK]` symbol. Without inserting spaces around emojis, the example post "Great video😘😘😘" would be tokenized as "Great, `[UNK]`". With our pre-processing, it is tokenized as "Great, video, `[UNK]`, `[UNK]`, `[UNK]`". On the word embedding level, we use a dropout of 10 percent, which means that every tenth word is randomly removed from the input to regularize the model. We use the BERT *base* model, which has 768 hidden units.[10] Therefore, the final prediction layer is a dense layer with softmax activation, which maps the 768-dimensional vectors either to three outputs for the multi-class classification or two outputs for the binary classification.

For our ensemble, we train multiple BERT models. We train each model for up to ten epochs and halt the training if no learning progress is made for two subsequent evaluation steps. This early stopping mechanism monitors the weighted F1-score on a 10 percent validation set. An evaluation on this set runs every 40 batches. With a batch size of 48, there are approximately two evaluations per epoch. Each training process

[10] `www.huggingface.co/bert-base-uncased`

Text: Great video😘😘😘
Tokens: Great, video, `[UNK]`, `[UNK]`, `[UNK]`
Label: non-aggressive (NAG)

Text: RSS agenda is to demolished opposite options
Tokens: RS, ##S, agenda, is, to, demolished, opposite, options
Label: covertly aggressive (CAG)

Text: You are soo fucked up that you can't understand someone else's perspective...
Tokens: You, are, so, ##o, fucked, up, that, you, can, ', t, understand, someone, else, ', s, perspective, ., ., .
Label: overtly aggressive (OAG)

(a) Task A: aggression identification

Text: I think feminists are lesbians,OAG,GEN
Tokens: I, think, feminist, ##s, are, lesbian, ##s
Label: gendered (GEN)

Text: kill all those womens who file faje rape and dowry cases,CAG,NGEN
Tokens: kill, all, those, women, ##s, who, file, f, ##aj, ##e, rape, and, do, ##wry, cases
Label: non-gendered (NGEN)

(b) Task B: misogynistic aggression identification

Figure 4.5: Training samples with their respective tokenized texts and labels.

starts with a different random seed. Thereby, not only does the random initialization of the weights of the final prediction layer vary among the models, but also the random data split for the early stopping is chosen differently. As the loss function, we use cross-entropy loss weighted by the class distribution. The learning rate is set to $5 \cdot 10^{-5}$ but uses a warmup phase as it is standard for fine-tuning BERT models. We use a linear learning rate warmup for the first 30 percent of the training up to the rate of $5 \cdot 10^{-5}$. Afterward, the rate linearly decays until the end of the training. Deviations from this general configuration for different runs of our approach are described in the next Subsection, Subsection 4.3.2.

The motivation for our ensembling approach is the instability of the classification performance across different fine-tuning runs of the same model. For example, Devlin et al. report that the accuracy on small datasets, such as the Microsoft Research Paraphrase Corpus (MRPC) with 3,600 samples, varies between 84 percent and 88 percent.[11] This variance occurs when fine-tuning even the exact same pre-trained model. The recommended approach is to restart the fine-tuning step multiple times. We are confronted with the same varying classification performance when fine-tuning BERT models on the

[11] `www.github.com/google-research/bert/blob/master/README.md`

shared task dataset. Slight changes to the training data and model hyperparameters, e.g., the random seed, cause the fine-tuned models to achieve very different results on the hold-out test dataset. These models only differ in the model weights in the final dense layer (the prediction head) when the training starts. In summary, the BERT models that are fine-tuned on the small shared task dataset are unstable and have a high variance. Our ensembling strategy is a variance reduction technique: bootstrap aggregating (bagging). We train up to 25 BERT models of the same kind on slightly different subsets of the data. A soft majority voting combines the predictions of these models:

$$\hat{y} = \underset{j}{\operatorname{argmax}} \sum_{i=1}^{n} p_{i,j}$$

where $p_{i,j}$ is the probability for class label j predicted by the i-th classifier (out of n classifiers). It sums up the probability mass assigned per class label and chooses the label with the highest probability as the ensemble's prediction $\hat{y}$. In other words, it chooses the class label that is most likely predicted. In contrast to that, a hard majority voting would choose the label that is most frequently predicted.

4.3.2 Experiments

This subsection contains three experiments. First, we evaluate our approach for both shared tasks on the test dataset and report the best model configurations. Two additional experiments study how the ensembling affects classification performance. To this end, the second experiment shows how many models should be ensembled to achieve the best performance. The third experiment is an ablation study to determine whether the random data splits or the random weight initialization cause the ensemble's superior performance compared to single models.

The shared task uses the weighted F1-score for the evaluation. As a consequence, the score for the majority class is more critical than for the other classes. Table 4.6 lists the performance that our approach achieved on the test dataset. In five out of six subtasks of the shared task on aggression identification organized by Kumar et al., our approach outperformed all other 15 participating teams [101].[12] The only exception is the English-language version of task B. We believe our model's inferior results for this task are caused by using a case-sensitive BERT model. For all other tasks, we used case-agnostic BERT models, which outperform the case-sensitive ones. The largest gap to the second-best submission is at the English-language version of task A. Our approach achieves a 4.4 percentage points better F1-score than the second-best approach. Table 4.7 lists the model configurations that achieved the best results on the test dataset. Note that the number of submissions for the test dataset was limited to three per task and language. Therefore, we could evaluate only a small set of different configurations. This limitation is also the reason why we can only assume that a case-agnostic BERT model would achieve a higher F1-score for the English version of task B than the case-sensitive model that we used for our submission. We did not submit the predictions of such a case-sensitive model due to the limited number of allowed submissions.

[12] Based on a preliminary version of our ensembling method, we also made a submission [159] to another shared task on implicitly and explicitly offensive language classification organized by Struß et al. and outperformed all other six participating teams [186].

Table 4.6: Weighted F1-score on the test dataset. Our approach outperforms the best submission by other teams in five out of six subtasks.

	English		Hindi		Bangla	
	Task A	Task B	Task A	Task B	Task A	Task B
Our Submission	**.80**	.85	**.81**	**.88**	**.82**	**.94**
Best Other Submission [19]	.76	**.87**	.79	.87	.81	.93

Table 4.7: Configurations of our best-performing submissions on the test dataset.

	English		Hindi		Bangla	
	Task A	Task B	Task A	Task B	Task A	Task B
Model Language	English	English	multiling.	multiling.	multiling.	multiling.
Number of Models	20	25	15	15	15	25
Letter Casing	uncased	cased	uncased	uncased	uncased	uncased
Sequence Length	220	220	200	200	200	230
Cross Entropy Loss	weighted	weighted	non-weighted	weighted	weighted	weighted
Hold-Out Data	10%	10%	20%	10%	20%	10%
Patience	2	2	1	2	1	2

With the second experiment, we study how many models should be included in the ensemble to achieve the highest weighted F1-score at the shared task. To this end, we fine-tune 100 BERT models that only differ in the initial random seed. All these models have the same architecture and the same hyperparameters, such as batch size or learning rate. However, the varying seed determines the randomly initialized weights for the final dense layer of the model (the prediction head), the order in which the training samples are processed, their distribution among the training batches, and finally, the 90 percent training and 10 percent validation split.

For each number from 1 to 50, which we call ensemble size, we select subsets of the 100 fine-tuned models of that size. For example, to build an ensemble of 50 models out of 100 trained models, there are $\binom{100}{50} \approx 10^{29}$ possible combinations. Since we cannot evaluate that many combinations, we randomly sample 1,000 combinations per ensemble size. The ensemble's predictions are generated with soft majority voting. Each ensemble is then evaluated on the exact same hold-out test dataset.

The top line in Figure 4.6 (random dataset split, random weight initialization) shows the weighted F1-scores that are achieved on average across the 1,000 combinations per ensemble size. The score increases for ensembles of up to 10 to 15 models, after which the advantage of adding even more models diminishes. A single model's performance is, on average, about four percentage points worse than the best ensemble. We could not use the official test dataset for our experiment because its labels were not available at the time of writing. Therefore, we use the official validation dataset for the evaluation and 90 percent of the official training dataset for training. 10 percent of the official training dataset are used as validation data for the early stopping mechanism. The model seems to underfit because this mechanism halts the training too early on the smaller dataset.

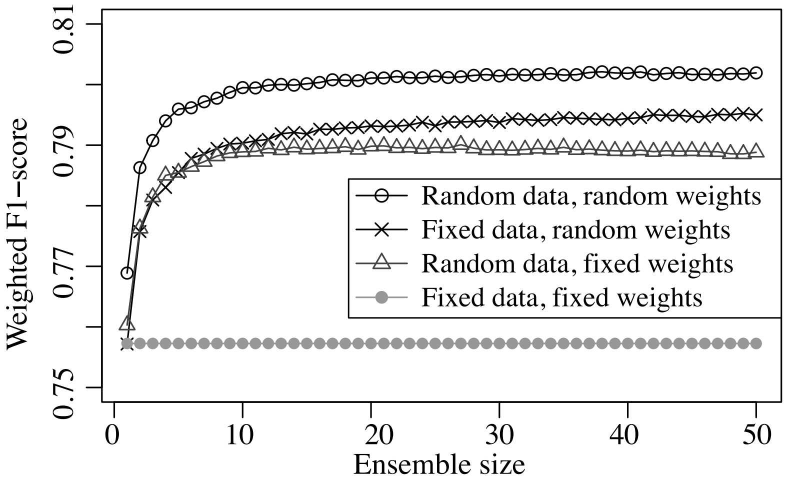


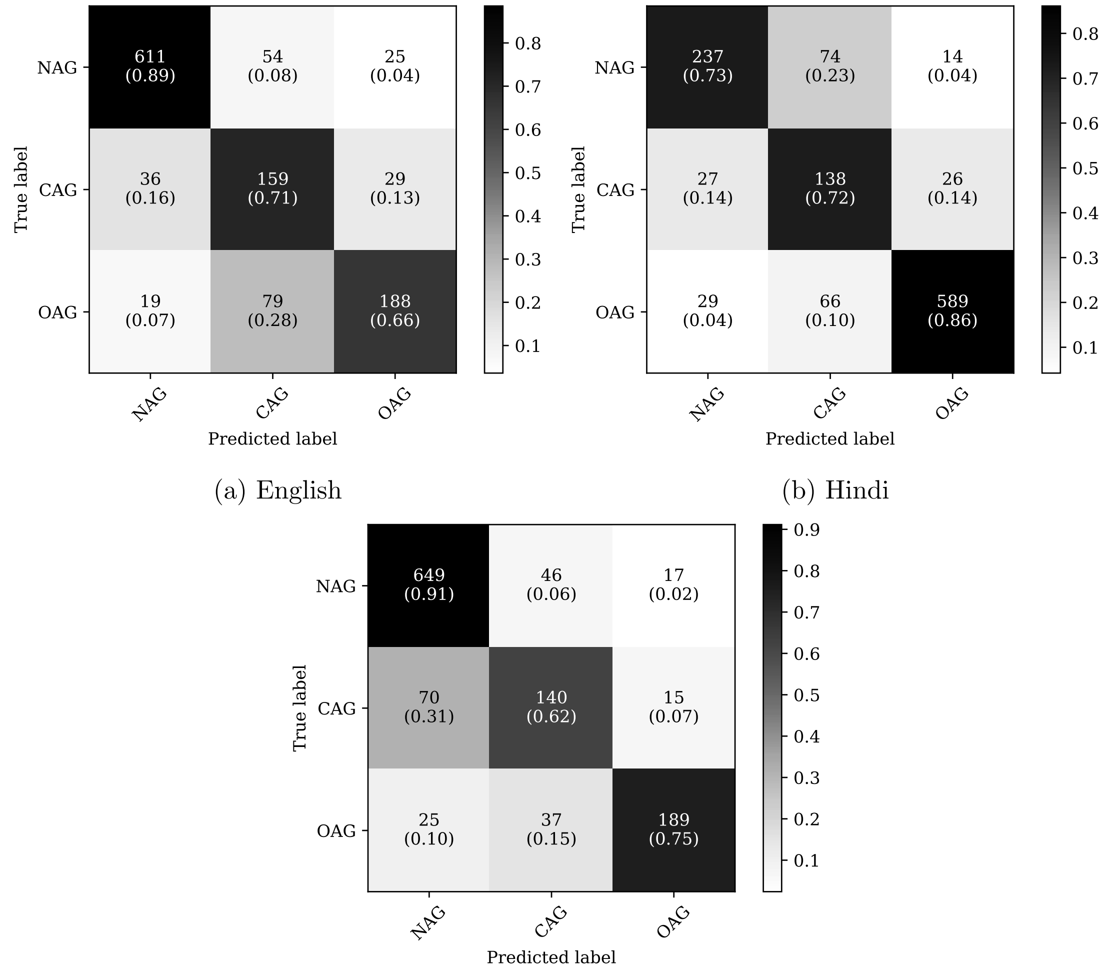
Figure 4.6: The increased performance of an ensemble of BERT models is mainly due to random weight initialization rather than random splits of training and validation data.

This experiment — in particular the fine-tuning of 100 BERT models and combining and evaluating the predictions of thousands of subsets of these models — is computationally expensive. Despite the small size of the dataset, it took approximately seven hours on two Nvidia GeForce GTX 1080 Ti GPUs with 11GB memory to complete the experiment. Training time and inference time increase linearly with the ensemble size.

The third experiment studies whether training on slightly different subsets of data or differently initialized weights in the final prediction layer (prediction head) causes the ensemble's strong performance. Our hypothesis is that the main reason is the weight initialization. To test this hypothesis, we compare four different variations of our approach. Figure 4.6 shows the weighted F1-scores for all four variations per ensemble size. First, we vary not only the random seeds for the weight initialization but also the training and validation split. As a consequence, the training data of the models differs slightly. Second, we vary the random seeds for the weight initialization while using the exact same training and validation split. For this variation, all models are trained on the exact same training data. Third, we use the same weight initialization for all models but vary the random splits of training and validation data. Fourth, we keep both the weight initialization and data splits fixed across all models. In the fourth variation, all trained models are identical, and thus, ensembling does not improve the performance. The test set is the exact same in all four variations. Figure 4.6 confirms our hypothesis. The strong performance of our ensembles is mainly caused by using varying weight initializations for the individual models. The varying training and validation dataset splits have a smaller effect.

4.3.3 Discussion

Figure 4.7a, Figure 4.7b, and Figure 4.7c show normalized confusion matrices for task A on the test datasets. For task A on the English test dataset, the most frequent misclassification (with regard to relative numbers) is the prediction of CAG instead of



(a) English

(b) Hindi

(c) Bangla

Figure 4.7: Confusion matrices for task A on the test datasets of different languages.

OAG (28 percent of all posts labeled as OAG). On the Hindi dataset, NAG is more frequently misclassified as CAG (23 percent of all posts labeled as NAG). On the Bangla dataset, CAG is most often misclassified as NAG (31 percent of all posts labeled as CAG). For all three languages, NAG and CAG are often mixed up, and the same holds for CAG and OAG. This result is not to our surprise as NAG is more similar to CAG than to OAG and OAG is more similar to CAG than to NAG. A non-aggressive post is easier to distinguish from an overtly aggressive post than from a covertly aggressive one.

A weakness of our approach is the limited vocabulary of the BERT models. First, the meaning of emojis is ignored, and they are tokenized as unknown symbols, although they frequently occur in the dataset. For example, 😂 is the most frequent emoji in the English training dataset (488 occurrences) followed by 👍🏼 (239 occurrences). We assume that the model's performance could be improved by replacing each emoji with its text representation from the Unicode standard, such as *face with tears of joy* or *thumbs up*.

Moreover, the Hindi and Bangla datasets contain non-Latin characters. The pre-trained multilingual BERT that we use for our submission discards all these characters.

However, there is another recent BERT model that overcomes this issue. It is called *multilingual cased* and is trained on non-normalized text (no lower casing, accent stripping, or Unicode normalization). This model is tailored to datasets with non-Latin characters, and we assume it would perform better than our current approach for the Hindi and Bangla datasets.

Last but not least, note that the class distribution of the Hindi test dataset for both tasks is much different compared to the training and validation datasets. Presumably, the reason for that is that the test dataset was sampled from a different social media platform than the training and validation datasets. More details can be found in the dataset description paper [22].

4.4 Explanation Methods

This section focuses on the evaluation and comparison of attribution-based explanation methods for toxic comment classification. To this end, we use a word deletion task to compare an interpretable machine learning model, a model-agnostic explanation method, a model-based explanation method, and a self-explanatory model. In a second experiment, we use the explanatory power index as an evaluation metric. Further, we take into account the classification accuracy of each approach and discuss strengths and weaknesses in the application context of (semi-)automatic comment moderation.

4.4.1 The Need for Explanations

Explanations play an essential role in real-world recommender and classification systems. Users trust recommendations and algorithmic decisions much more if they provide an explanation as well. An example are the “other customers also bought” recommendations on e-commerce platforms. By explaining why a particular product was recommended, the recommendations are considered better and more trustworthy.

In the context of comment classification, explanations are also very much needed to establish trust in the moderation process. Online news platforms list their discussion rules in the form of guidelines, and they overlap considerably with the “netiquette”, the basic rules about communication over the Internet. However, that does not mean all readers have these rules in mind when they post comments. Therefore, moderators on online discussion platforms explain why they intervene. For example, they replace a removed comment with the following text: “Removed. Please refrain from insults.” or “Removed. Please refrain from insinuations and personal attacks.”. In case they ultimately close a discussion, they post a final comment, for example, stating: “This discussion has been closed due to (racist) generalizations, baseless assumptions up to conspiracy theories and extreme polemics.”. On the one hand, the idea behind these explanations is transparency. On the other hand, they aim to educate readers to adhere to the discussion rules.

Supervised machine learning approaches for comment classification often use black-box models, which cannot explain automated decisions. Therefore, they cannot be appropriately applied to comment moderation. Readers and moderators are skeptical about incomprehensible automation. If no reason is provided why a reader's comment was removed or not published in the first place, readers might get the feeling of being censored or their opinion otherwise oppressed. Explanations can help to build trust and increase the acceptance of machine-learned classifiers. Only then can a fair and transparent moderation process be ensured.

There are two more reasons for explanations in general. First, there are legal reasons to utilize machine-learned classifiers only if they can give explanations for their decisions. For example, under certain circumstances, the General Data Protection Regulation (GDPR) in the EU grants users the right to "obtain an explanation of the decision reached" if they are significantly affected by automated decision-making, e.g., if a credit application is refused.[13] A second reason is that explanations help to reveal a model's strengths and weaknesses. They could also benefit the task of identifying a potential bias in the decisions of a model. Scientists can then work on improving the model based on these insights.

4.4.2 Explainability and Interpretability

There is plenty of research on toxic comment classification, but one aspect of this task has gone mostly unnoticed: the need for explaining classification results. Research on explanation methods distinguishes explainability from interpretability. The former refers to locally comprehending individual decisions, while the latter refers to globally comprehending the decision function [57, 117, 118]. Unfortunately, there is no universal definition of these two terms. The definitions used in this thesis are:

- A decision function f is called explainable if the decision $f(x)$ for each single input $x \in X$ (in domain X) can be explained in comprehensible terms.
- A decision function f is called interpretable if the whole function f (for the whole domain X) can be explained in comprehensible terms.

In the area of image classification, CNN-based explanation methods are prominent. For example, DeConvNet [217] inverts the convolutional operations to gain explanations, and an approach by Simonyan et al. [180] applies sensitivity analysis to achieve similar results. There have been several follow-up papers that compare these two approaches and propose combinations [93, 183].

Explanation methods for text classification are rarely studied. Nguyen [123] compared human evaluation and automatic evaluation for explanation methods. The comparison uses the twenty newsgroups dataset and a dataset of movie reviews. To the best of our knowledge, the only publication on explanation methods in the field of offensive language detection is by Carton et al. [33]. The authors use an attention mechanism to generate explanations for the detection of personal attacks.

An empirical study by Chakrabarty et al. [36] shows the importance of contextual attention and self-attention for abusive language detection. Whether attention weights can also be used as explanations is under discussion [86, 208]. We consider an LSTM model with an attention mechanism [211] as an example of a self-explanatory model. The

[13] www.eur-lex.europa.eu/eli/reg/2016/679/oj

inherent attention weights provide attribution-based explanations. Further, we consider a naive Bayes classifier, which is an example of an interpretable model. Each classification result (and the entire model) can be understood with the help of the classifier's discrete conditional probabilities. The relevance of a word w is the probability that the class c is predicted given w:

$$P(c|w) = \frac{P(c) \cdot P(w|c)}{P(w)}$$

The attention-based LSTM and the naive Bayes classifier are two a priori explainable models. We also consider two post-hoc explanation methods in this section: layer-wise relevance propagation (LRP) and local interpretable model-agnostic explanations (LIME). The idea behind LRP [14] is to backpropagate the relevance scores from the output layer to the input layer of a neural network. To this end, the relevance of each input value (feature) is derived from the neuron activations in the output layer. This procedure makes LRP a *model-based* explanation method. The idea behind LIME [146] is to use a local approximation of the classifier f at a point x and its neighborhood. This local approximation needs to be an interpretable classifier and a good approximation of f in the local neighborhood of point x. The authors evaluated their *model-agnostic* explanation method with text and image classification tasks.

4.4.3 Explanation Methods

For our comparative study, we implement a variety of classifiers for offensive language detection and suitable explanation methods. To train the classifiers, we again use the dataset from the Kaggle challenge on toxic comment classification.[14]

There are four different classifiers that we implement and pair with different attribution-based explanation methods. First, there is a multinomial naive Bayes classifier, which serves as a baseline. It is interpretable by default and provides explanations in the form of conditional probabilities. Further, we implement an SVM and an LSTM model. The input to the SVM is a tf-idf vector representation of the unigrams in the comment text. GloVe word embeddings [132] serve as the input to the neural network. Both the SVM and the LSTM model are paired with the two explanation methods LRP and LIME. To this end, we adapt the LRP implementation by Arras et al. [12][15] and the LIME implementation by Ribeiro et al. [146][16]. To generate explanations for SVM and LSTM with the model-agnostic method LIME, we first sample perturbations of the input text by randomly deleting words. For each sample, we calculate the class probabilities with the SVM and the LSTM by applying a softmax function as the final calculation step. The default ridge regression algorithm is used to train an interpretable linear model. This model learns the word relevance scores based on the classified samples. Last but not least, we implement an LSTM model with an attention mechanism, which is an example of a self-explanatory model. It uses attention weights on the word level (not on the sentence level) and implements the architecture by Yang et al. [211].

The GloVe word embeddings are trained from scratch on the full dataset. We restrict the input length of the basic LSTM model and the LSTM model with an attention

[14] `www.kaggle.com/c/jigsawtoxic-comment-classification-challenge`

[15] `www.github.com/ArrasL/LRP_for_LSTM/`

[16] `www.github.com/marcotcr/lime`

mechanism to a maximum of 250 words. Further, we use 50 LSTM units, which means the output of this layer is 50-dimensional. The training of the networks runs for five, respectively, three epochs with the Adam optimizer until the validation loss increases.

The task on our dataset is a multi-label classification task. The network architecture addresses this multi-label task by sharing the same LSTM layer across all class labels. However, for each label, an independent dense layer follows after the output of the last LSTM unit. The attention mechanism is also trained for each label individually and is optionally inserted between the LSTM output and the following dense layer.

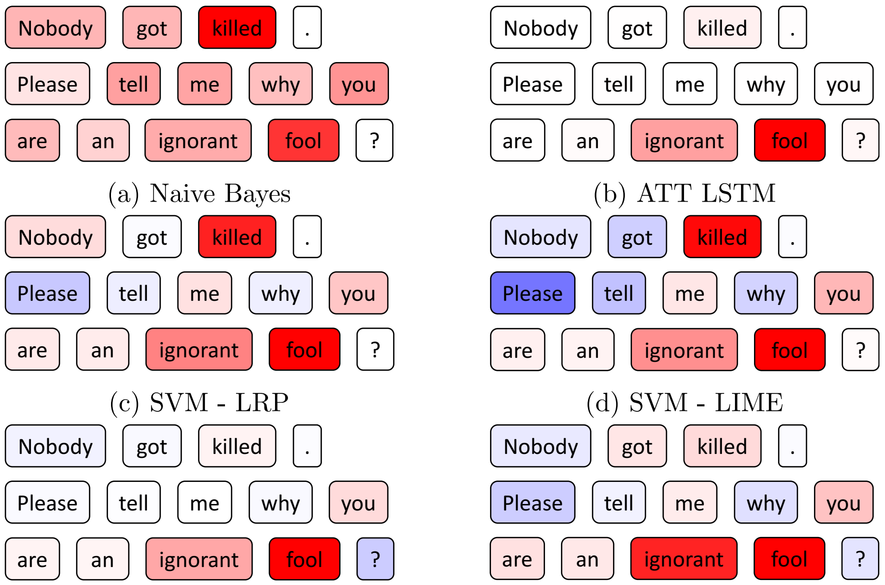
With the SVM model and the naive Bayes model, we use stemming to reduce the vocabulary size. They are trained according to a one-vs.-rest scheme to conform to the multi-label classification task. Therefore, the trained models can be seen as six independent binary naive Bayes classifiers and six independent binary SVMs. The SVM uses a linear kernel. There is only one hyperparameter to choose, which is the regularization term c. We set $C = 0.6$ and thereby relax the penalty for misclassifications.

To give an example of the explanations, Figure 4.8 and Figure 4.9 visualize the word relevance scores generated by the different explanation methods for two toxic comments. The conditional probabilities of the naive Bayes model and the attention weights of the attention-based LSTM model define positive word relevance scores between 0 and 1. In contrast to that, LIME and LRP define unbound relevance scores, which can also be negative. A negative word relevance score means that the respective word indicates the absence of a particular class rather than its presence. Because the attention weights are class-independent, these weights can only explain the predicted class. All other methods can also be used to explain a class that was not predicted by the classifier. This property can be used to analyze which words speak in favor of a not predicted class.

In Figure 4.8, the naive Bayes model marks the words *killed* and *fool* as most relevant for the decision to classify this comment as toxic. Similarly, the SVM model with LRP and LIME mark these two words. In contrast to that, the word *killed* is less relevant for the LSTM models (with and without attention). Only the naive Bayes model and the SVM model use stemming but not the LSTM models. The stemming collapses *killed* to *kill*. Therefore, our naive Bayes model and SVM model cannot distinguish the active form of the verb from other words with the same stem. In this particular context, the non-stemmed word is not toxic. The stemming misleads the models to wrongly explain the comment's toxicity with this word.

The attention mechanism highlights the words *ignorant* and *fool*. The word *killed* is marked as slightly relevant, and all other words as irrelevant. This explanation aligns with an explanation a human would give. In general, we find that the attention mechanism gives meaningful explanations for toxic comments. For non-toxic comments, however, its explanations can be misleading. The attention mechanism distributes a relevance score of 1 among the words — even if there is nothing toxic in the comment. To our surprise, the attention mechanism often marks punctuation as relevant in non-toxic comments.

The basic LSTM model marks only a few words as relevant, and most words have relevance scores close to zero. These sparse explanations are suitable for our dataset, as there is typically a small set of toxic words that explains the toxicity of the entire comment. In Figure 4.8c to 4.8f, LIME and LRP assign negative relevance scores to



(a) Naive Bayes

(b) ATT LSTM

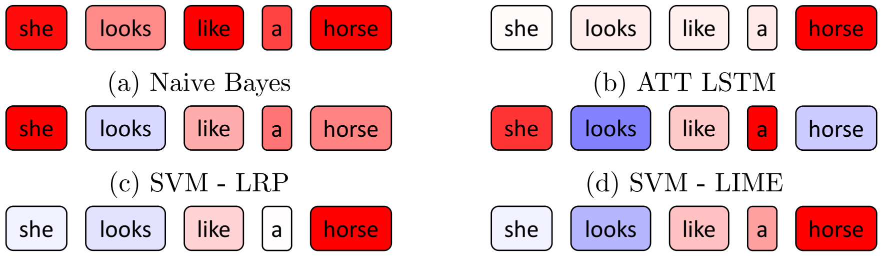
(c) SVM - LRP

(d) SVM - LIME

(e) LSTM - LRP

(f) LSTM - LIME

Figure 4.8: This heatmap visualizes positive (red) and negative (blue) word relevance scores generated by combinations of different classifiers and explanation methods for an explicitly toxic comment.



(a) Naive Bayes

(b) ATT LSTM

(c) SVM - LRP

(d) SVM - LIME

(e) LSTM - LRP

(f) LSTM - LIME

Figure 4.9: This heatmap visualizes positive (red) and negative (blue) word relevance scores generated by combinations of different classifiers and explanation methods for an implicitly toxic comment.

the word *Please*. This negative relevance score means that this word speaks against the toxicity of the comment.

The heatmaps in Figure 4.9 visualize the word relevance scores of another comment. It contains no swear words, but it is still offensive. The negatively connoted association of a person with an animal falls into the category of dehumanizing language. Without the full context, none of the single words explains the toxicity of the comment. Therefore, it is difficult to assess the toxicity and provide an attribution-based explanation. Only the basic LSTM model classifies this short example comment correctly.

Table 4.8: Precision, recall, and F1-score of the classifiers on the toxic comment dataset. Bold font indicates the best F1-score per class.

Class	Metric	NB	SVM	LSTM	ATT
Toxic	P	.70	.83	.82	.85
	R	.64	.66	.68.	.70
	F1	.67	.74	.74	**.76**
Severe Toxic	P	.14	.52	.57	.58
	R	.92	.18	.22	.08
	F1	.25	.27	**.32**	.14
Obscene	P	.52	.86	.81	.86
	R	.76	.68	.72	.67
	F1	.62	**.76**	**.76**	.75
Threat	P	.04	.72	.31	.89
	R	.60	.29	.15	.35
	F1	.07	.42	21	**.51**
Insult	P	.48	.78	.73	.78
	R	.76	.58	.69	.60
	F1	.59	.67	**.71**	.67
Identity Hate	P	.12	.64	.55	.66
	R	.73	.23	.29	.50
	F1	.20	.34	.38	**.57**

4.4.4 Evaluation

The following evaluation is three-fold. First, we compare the different classifiers (naive Bayes, SVM, LSTM, and LSTM with attention mechanism) with regard to their classification performance. Second, we pair them with attribution-based explanation methods and evaluate the generated explanations based on a word deletion task. The third part of the evaluation uses the explanatory power index (EPI) by Arras et al. [11].

Classification Performance. To evaluate the classification performance of the different classifiers, we use a multi-label classification task on the toxic comment dataset. Due to the imbalanced class distribution of this dataset, we refrain from using accuracy as the evaluation metric and instead use precision, recall, and F1-score. Table 4.8 lists the results on the test set and shows that the naive Bayes model is weakest, followed by the SVM model. The basic LSTM model and the LSTM model with attention mechanism overall achieve similar F1-score with larger differences in the less populated classes *severe toxic*, *threat*, and *identity hate*. For the following evaluation of explanation methods, we consider a binary classification task based on the *toxic* class label only. All classifiers achieve their best performance for this most frequent label.

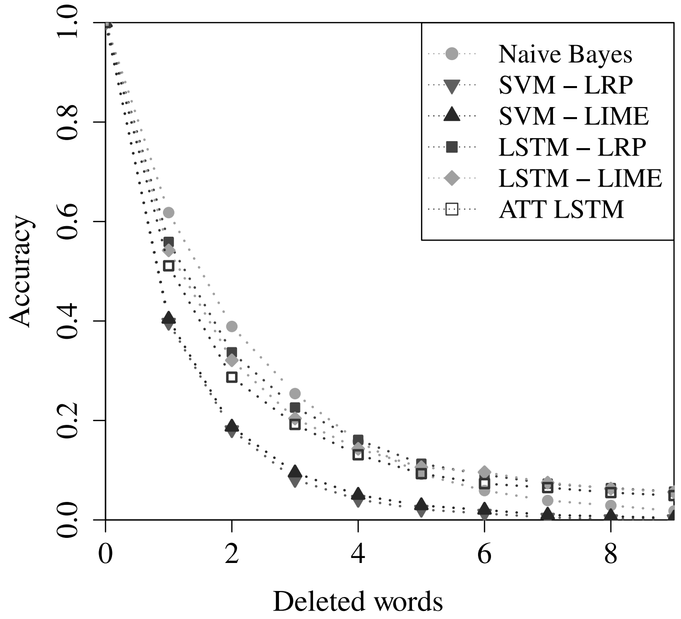


Figure 4.10: Correct classifications into the *toxic* class change to *non-toxic* if the most relevant input words are deleted. This result shows that the word relevance scores successfully mirror a word's influence on the classification result.

Word Deletion Task. We consider a word deletion task to evaluate whether the explanation methods correctly identify those input words that are most relevant for the classifier's output. It is based on an idea by Arras et al. [12]. The task evaluates whether the words that the explanation points out to be relevant for the classification indeed have a strong influence on it. Therefore, each explanation method needs to calculate a relevance score for each input word. The word with the highest relevance is deleted, and it is checked whether the model's classification result changes for the perturbed input.

Given the set of true positives (toxic comments that are correctly identified as toxic), we use each explanation method to calculate relevance scores for the words in each comment. For each method, we then delete the most relevant words from each comment. If the word is indeed relevant for the classifier's decision, the classification most likely changes for the perturbed comment. Step-by-step, we delete more and more words with decreasing relevance scores. An explanation method is considered to provide good relevance scores if the classification changes for many comments after deleting only a few words.

Figure 4.10 shows how the accuracy quickly drops as more and more words are deleted. By deleting four words, more than 80 percent of the comments that were previously correctly classified as toxic (true positives) are classified as non-toxic. This result confirms that the classifiers provide those words as explanations that often constitute the toxicity of a comment, e.g., swear words.

Further, Figure 4.10 suggests that SVMs provide better explanations than LSTMs. This suggestion is misleading and reveals one limitation of the experiment. Each method starts with its own set of true positives. Therefore each line in the plot corresponds not only to a different explanation method but also to a slightly different dataset. While the overlap of the sets is relatively large, the LSTM model's set of true positives is almost a superset and slightly larger. It also contains some of the more difficult samples of toxic comments, which are correctly classified by the LSTM model but misclassified by the naive Bayes model. One idea to get rid of this problem is to use the intersection of all sets of true positives. The resulting comments are unanimously correctly classified. However, when we further explored this idea, we found that this set is rather small. More importantly, it contains only the most simple cases — the comments that *all* classifiers correctly detect as toxic.

Still, for those comments that it classifies correctly, the SVM classifier definitely provides the best explanations according to the word deletion experiment. However, the true positives of the LSTM model also include comments whose toxicity can only be detected with context. A comment that contains a single swear word is easier to perturb to be classified as non-toxic than a comment that is toxic in its entirety.

Explanatory Power Index. Arras et al. [11] proposed a two-step approach to quantify the explanatory power of a text classifier with their explanatory power index (EPI). Intuitively speaking, the EPI describes how well the document summary vectors capture the semantic similarity of documents of the same class by clustering them in the high-dimensional vector space. We follow this approach and first calculate one document summary vector per comment in the test set based on each combination of a classifier and an explanation method. The document summary vector is either calculated as a weighted average of the comment's GloVe word embeddings or as the comment's weighted tf-idf vector representation. We compare a variety of approaches for weighting the words based on word relevance scores.

In the second step, we perform a k-nearest neighbor (kNN) classification on these document summary vectors based on each classifier's predictions. This step is repeated ten times on different random splits of the data and with different values of k. The classification accuracy of the kNN classifier is averaged for each k over the ten runs. The EPI is defined as the maximum achieved classification accuracy. We limit the dataset to all toxic comments and a random sample of non-toxic comments of the same size. This downsampling reduces the data to a balanced set of $4,300$ comments and allows using accuracy as the evaluation metric.

Table 4.9 lists the EPI for the different classifiers paired with the respective explanation methods. The results show that weighting a document's bag-of-words vector representation with conditional probabilities from the naive Bayes baseline has the weakest explanatory power. It is outperformed by the other two baselines: the SVM model with tf-idf weights and the basic LSTM model with averaged GloVe vectors to obtain document summary vectors. The explanatory power of the basic LSTM classifier combined either with LIME or LRP is superior to all other methods. Although the LSTM model with attention mechanism achieved slightly better classification results (F1-score of 76.4 percent vs. 74.4 percent), the attention weights are not as suited for explanations as word relevance scores generated with LIME or LRP for the basic LSTM model.

Table 4.9: Explanatory Power Index (EPI) of classifiers and explanation methods. Hyperparameter k denotes the number of nearest neighbors that maximizes the EPI.

Classifier	Explanation Method	EPI	k
Naive Bayes	Conditional Probability	.82	3
SVM	Tf-idf	.88	25
	LRP	.93	19
	LIME	.93	19
LSTM	GloVe	.85	15
	LRP	**.99**	3
	LIME	**.99**	9
ATT LSTM	Attention Mechanism	.92	11

4.4.5 Discussion

LIME and LRP achieve similar results in our experiments. However, they strongly differ in their computational costs. The runtime to generate explanations with LIME is about 40 times higher than with LRP. This difference is because LRP needs only one backpropagation run to propagate the relevance scores from the output layer to the input (word) layer. In contrast to that, LIME requires perturbing a large set of samples. These samples need to come from the local neighborhood of the comment to be explained. For example, they need to have many words in common. The more samples are used, the more stable are the explanations. In the word deletion experiment, LIME has an unfair advantage over the other explainability methods due to the way it is trained. The perturbation in its training process is similar to the perturbation in the word deletion task. Therefore, LIME is tailored to this task.

A downside of the attention mechanism is that it cannot provide class-specific word relevance scores. Strictly speaking, the attention weights and the derived relevance scores do not refer to the word level. The weights instead refer to the hidden states in the sequence of LSTM units. The attention mechanism explains which states are most relevant for the network's final output. The activation of a hidden state is the result of processing a subsequence of the input word sequence — regardless of the actual classification output (toxic/non-toxic). The heatmap visualizations in Figure 4.8b and Figure 4.9b show that the attention mechanism distributes the relevance only among a few words, more precisely, hidden states. One reason for that is that a single hidden state actually captures information gained from a sequence of input words.

A limitation of attribution-based explanations for offensive language detection seems to be a focus on words that are toxic regardless of the context. This limitation might render them useless for the detection of implicitly offensive language. The latter defines offensiveness that is not directly expressed but only arises from the context, uses irony or sarcasm, or can be inferred from metaphors, comparisons, or ascribed properties [186].

In the application scenario of comment moderation on an online platform, a classifier that achieves slightly worse accuracy might be preferable if it provides explanations. The reason for this trade-off is not only the importance of transparency of the moderation process and acceptance by the user community. Explanations also facilitate the maintenance of a trained classification model. Since news topics and the corresponding reader discussions change daily, adaptation is necessary — also the adaptation of machine-learned models. For example, on one day, an offensive comment might be removed from the platform. However, the same comment might be the legitimate center of the discussion on the next day because it is a quotation from a well-known politician. In industry applications in general, explanations can support software developers and maintainers to better understand machine-learned models and the associated software.

4.5 Application Scenario

This section addresses the transfer of our scientific findings to the industrial application of machine-learned models for toxic comment classification. Based on a collaboration with the large online news provider Zeit Online, we consider their dataset introduced in Section 3.1 and propose a semi-automatic approach for comment moderation to assist moderators. In a holistic approach, we combine features of comments, news articles, and users in a logistic regression model. While we could have used a deep learning approach, such black-box models do not readily fulfill the requirement of comprehensible classification results. Even with the explanation methods presented in the previous section, deep neural networks still lack trust and acceptance due to their complexity. Moderators and readers both need to know the reasons for a classification result. As an advantage of our logistic regression model, we can give insights into how each feature influences the classification and which features make a comment toxic in a specific context.

4.5.1 Logistic Regression with a Diverse Feature Set

We define three categories of features to classify a given comment:

- Comment features aim to model linguistic, syntactic, and semantic properties of the comment's text. Comment metadata, such as the publication date, is also part of this feature.
- User features introduce information about the individuals behind comments and their behavior, in particular the time span between consecutive comments, previous toxic comments, and topics of interest.
- Article features relate to the news article referenced by a comment, e.g., its news section, publication date, or article author.

We propose a logistic regression model trained in a supervised fashion on binary-labeled data. For a given comment with information about the associated user and article, the model predicts a probability of toxicity. To obtain binary labels of toxicity, we choose a probability threshold that is tailored to achieve a high recall: we want to prevent false negatives. The set of presumably non-toxic comments can be published immediately and without manual intervention. In contrast, the set of presumably toxic comments is presented to a moderator for review. In practice, a high recall corresponds to the situation where moderators get to see mostly all actually toxic comments. The downside of a lower precision is that moderators also need to check a few non-toxic comments. This trade-off ensures that moderators can (almost) be sure that no toxic comment escapes their review.

Comment Features. Many features can be derived from a comment's text or its nesting level in the threaded structure of a discussion. Our linguistic features include character-level and word-level features. Neither normalization, such as stemming or lemmatization, nor any other pre-processing is applied. As the most basic feature, we consider a comment's number of characters and words. The combination of these two features describes the average word length of a comment. In our dataset, toxic comments are, on average, shorter (48 words) than non-toxic comments (61 words). Our other text features focus mostly on the use of punctuation or capitalization. We assume that extensive use of punctuation or capitalization of whole words indicates an aggressive tone. Further, comments with web links to external pages frequently violate user guidelines. For example, the linked page's content might contain insults or advertisements. We count occurrences of "http" to capture web links, but do not distinguish between internal and external links. Interestingly, toxic comments contain, on average, fewer negation words (0.92 words), such as "not" or "never", than non-toxic comments (1.28 words). In summary, our set of linguistic features for a comment includes: (1) the number of characters, words, and distinct words, (2) the number of question marks, exclamation marks, periods, colons, quotation marks, and uses of "http", as well as (3) the ratio of uppercase to lowercase letters.

While a comment's syntax might not be toxic itself, it can still serve as an indicator of toxicity. For example, the extensive use of personal pronouns might indicate personal attacks against others. Similarly, many adjectives might indicate extensive descriptions or name-calling of other users or organizations. We apply part-of-speech tagging and count the number of adjectives, determiners, personal pronouns, and adverbs in each comment to detect such patterns.

Off-topic comments that are not related to the news article are also considered toxic. To measure the topical similarity of an article and a comment, we apply topic modeling. On a set of roughly 25,000 articles, we learn a topic model with latent Dirichlet allocation. The topical similarity is then used as a feature in our classifier. Further, we compare the tf-idf vector of the comment with the tf-idf vector of its corresponding article and of toxic comments posted in response to this article. One feature is the cosine similarity of these vectors. Another similar feature is the Kullback-Leibler divergence of the word frequency distributions of a comment and an article.

The word cloud in Figure 4.11 illustrates German unigrams that are most frequently used in toxic comments in our dataset. Besides unigrams, we include 2-grams and 3-grams and thereby consider the context in which a word is used. The word cloud shows, for example, that mentioning the last name of the German chancellor, Merkel, strongly indicates a toxic comment. However, this indication changes depending on the preceding word: a comment containing "Thank you Merkel" is twice as likely to be toxic than a comment containing "Mama Merkel".

Arbitrarily long and rare compound nouns are a challenge specific to the German language. Coining new words increases the sparsity of the data: these words typically occur only once in an entire corpus and are so-called *hapax legomena.* At test time, this phenomenon leads to frequent out-of-vocabulary problems. Character n-grams do not suffer from these problems because they are able to capture substrings in compound nouns. Related work shows that character n-grams can be successfully applied to detect abusive language in English-language content [124, 175]. Obfuscated words and unusual spellings typically pose problems for word-based approaches due to high sparsity but can be countered with character-based techniques. For this reason, we add character n-grams ranging from length 3 to 5 to our feature set, targeting also the German-specific challenge of compound nouns in particular.

To capture the semantic meaning of words, we apply word embeddings. In particular, we use a standard Word2Vec approach and model each word as a 100-dimensional vector. The embedding of an article or a reader comment is simply the average embedding of all its words.

The depth of each comment, which corresponds to the nesting level, is used as another feature. For example, a depth of 3 means that the comment replies to another reader's reply. Standard interfaces allow users to post their comments as a reply to another comment. Thereby, a discussion thread can form a tree structure. In the dataset from Zeit Online, comments that are direct replies to an article have a higher probability of being toxic (4.0 percent) compared to those that are replies to other readers' comments starting at a depth of 3 (2.5 percent).

User Features. Our dataset also provides information about each user (commenter). We distinguish two categories of features based on user information: time-based and history-based features. We model four time-based features: the time in seconds since the user last posted a toxic or a non-toxic comment in the same or any other article discussion. These values indicate heated debates or reactions to a previously removed comment, which are frequent in the dataset. For example, suppose a comment was posted within 10 minutes after a toxic comment by the same user on the same article. In that case, it has a 19 percent chance of being toxic compared to the global average chance of 3 percent.

The history-based features are statistics of all comments by a user prior to posting a particular comment. We count the number of toxic and non-toxic comments in the same news section as the article and globally. Since our dataset is limited to the time span between January 2016 and March 2017, we do not have access to historical information before that time. Therefore, our extracted history-based feature values are only a narrow excerpt of a user's full history. For example, a user who posted only a few comments in our dataset might have been much more active before 2016.

Article Features. Each comment in our dataset is posted in the context of a news article. An article that was just published a few hours ago still has lots of potential for discussion, whereas articles older than a few weeks rarely get any new comments. As described in Section 3.1, the news section of the article also influences its probability to receive toxic comments. Controversial political topics lead to more toxic comments than sections about more mundane topics, such as sports. Based on these observations, we define the following features: (1) time since the article's publication, (2) time since the last comment on the article, (3) time since the last toxic comment on the article, and (4) section of the article.

Word n-grams are at the top of the best-performing features for hate speech detection [16, 50, 124, 175, 203]. For this reason, we consider word n-grams as a baseline approach that all other features compete against. Further, we combine all single features into our holistic approach as a large feature set for the logistic regression. To the best of our knowledge, especially the context of comments, such as information about commenters and corresponding articles, has not been taken into account so far and extends the state-of-the-art. Our approach could also be used for the subtask of hate speech detection or related subtasks. The broad range of toxicity (as described, for example, in platform guidelines) motivates our large feature set. For other scenarios, this set might be unnecessarily large and could be reduced. Further, other tasks could map probabilities of the linear regression to binary labels differently. If these tasks do not require explanations for automated decisions, one might refrain from linear models at all in favor of deep learning approaches.

4.5.2 Evaluation

We split our dataset time-wise into training and test set to train only on past data and evaluate on future data. The chosen cutoff timestamp ensures that 10,000 toxic comments remain in the test set. A balanced class distribution is obtained in the test set by randomly sampling 10,000 non-toxic comments from the comments posted after the cutoff timestamp. All comments posted before that timestamp serve as training data. The regression model outputs probabilities of a comment being toxic, but not a binary label. To be able to compare with our ground truth labels, we set a threshold and map these probabilities to binary labels accordingly. To this end, all comments with a predicted probability above that threshold are marked as toxic and are forwarded to a moderator for a final check. All comments with a probability below that threshold are marked as non-toxic and can be published right away.

Use cases other than our real-world example could consider two thresholds: one for almost certainly toxic comments and one for almost certainly non-toxic ones. Only comments between those two thresholds are left for manual assessment. A disadvantage is that the decision to delete a comment could be made completely automatic if the model is confident enough, which is unacceptable in our scenario.

Table 4.10 summarizes the results of our experiments. As can be seen in the table, we chose the threshold in a way that at least 75 percent of the toxic comments are correctly classified (recall). The reason for this decision is that it is acceptable to present more comments than necessary to the moderators, but explicitly toxic comments should not slip through. On the one hand, if the threshold is set for a higher recall, then the precision decreases until moderators do not benefit from machine support but have to check almost all comments. On the other hand, if the threshold is set for a lower recall, more and more actually toxic comments are falsely classified as non-toxic. As a consequence, toxic comments are inadvertently published without manual review, and moderators rely on reports from readers.

After manually inspecting false negatives (toxic comments misclassified as non-toxic), we find that they are often only implicitly toxic, further context is needed for the clas-

Table 4.10: Precision, recall, and F1-score on the test dataset from Zeit Online.

Method	P	R	F1
Linguistic	.55	.77	.64
Syntax & Topic	.54	.76	.63
Word N-Grams	.56	.78	.65
Char N-Grams	**.65**	.78	.71
Word2Vec	.63	.76	.69
Article	.54	.75	.63
User	.52	.83	.64
Combined	.62	**.86**	**.72**

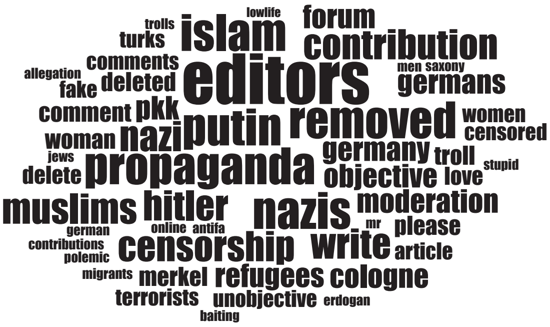


Figure 4.11: Indicative words for toxic comments.

sification, or the reasons for the assigned label are elusive. This finding might be due to the fact that various moderators labeled our ground truth dataset. They interpret the user guidelines differently and, therefore, make different decisions. Furthermore, different discussions with different previous comments might require more or less intervention by the moderators. As each comment in our dataset has been labeled by only one moderator, there is no way to evaluate the inter-rater reliability of these labels. Thus, similar comments posted at different times can be labeled differently, which is a tough challenge for classification algorithms. Even a more sophisticated classifier or more features might not help in situations where a team of moderators disagrees on the label of a particular comment. These issues are another reason why we propose a semi-automatic approach that integrates machine learning into the manual process of comment moderation, where humans make and account for the final decisions.

4.5.3 Discussion

In the following, we discuss the performance of selected features on our dataset. The analysis of single features helps to understand the result of an otherwise non-transparent automatic classification result. To assist moderators, these features can serve as indicators of toxicity and provide explanations for suggested automated decisions.

Table 4.11: Words in vicinity of the embeddings of comments.

Comment	"Hi, undermining democracy: correct, if such an arbitration court is in an agreement, it is an effect of this agreement. Thanks for the hint. DT is negotiating, or so he says, with 'America First' in mind. The existing agreements, especially NAFTA, were primarily disadvantageous to Joe Sixpack AND Juan Pérez in Mexico AND the USA. If Mexico does not like DT's proposals, they can cancel the agreement together with the USA and both can benefit. In other words, things can only get better for the lower middle class and below."
Nearby Words	free trade agreement, trade agreement, trade treaty, treaty, free trade contract
Comment	"The Kurdish population is being blackmailed by the PKK and hauled off into the mountains. If the Kurds in the FRG are for the PKK, then they should live there."
Nearby Words	Kurd, PKK, Kurdish area, Turkey, terrorist

Comment Features. The F1-score of character n-grams is exceeded only by the approach that combines all features. Further, the former achieves the best overall precision, as listed in Table 4.10. One reason for the strong performance of character n-grams might be the complexity of the German language. Word-based models are prone to out-of-vocabulary issues, and the German language allows creating very long compound words, which are used infrequently. Character n-grams are more robust than word-level features regarding compound words, neologisms, and typos. Some extreme examples for compound nouns in our dataset from Zeit Online are "Landesrundfunkanstaltsselbstbedienungsläden" loosely translated as "self-service stores of public service broadcasters" or "Stottertrottelkorbflechterautobahnlastkraftwagenfahrer" loosely translated as "stuttering douchebag who weaves baskets and drives trucks on highways". These nouns are unique in our dataset and also will almost certainly never be used again in any other comment. Although character n-grams do not capture whole words, they still capture the meaning of a comment well according to our results. Since word embeddings perform comparably well and achieve the second-best F1-score, we further explored words that are embedded in the neighborhood of the averaged word vectors that represent a comment. Exemplary results are shown in Table 4.11.[17] The examples suggest that embeddings are a suitable means of determining the broader topic of a comment.

Article Features. Article features describe the context of a posted comment. The classification of a comment as toxic only based on the article's section or the time since the article's publication obviously cannot lead to satisfying results. Article features are not a good stand-alone indicator of toxicity. For example, to achieve a recall above 75 percent, all comments in entire news sections would need to be considered toxic. Article features are more supportive evidence than stand-alone features.

[17] All examples in this section are translations from German into English.

User Features. An extensive user history is available only for a small subset of all users. A reason for that is the limited time span of the considered dataset. Since the available data covers roughly one year, the observed total number of comments posted by a user can only be a narrow excerpt of reality. In our dataset, every user initially starts at a comment count of 0. Only later on, our model learns to distinguish different users, for example, based on the frequency of their comments. We expect the features to perform better with the full user history. Nevertheless, several features show promising results. The number of toxic comments previously posted by the user (in the same news section as the article or in total) correlates with a new comment being toxic. Similarly, the time since the last toxic comment by the user also weakly correlates with a new comment being toxic.

4.6 Summary

In this chapter, we focused on the task of toxic comment classification. Based on our comparative study, we identified and addressed several challenges of this task. In particular, we concentrated on two challenges: learning from limited training data and integrating machine-learned models into the otherwise manual moderation process. To this end, we developed ensemble models that combine the strengths of multiple models including specially tailored deep neural networks that require only a few training samples. Top positions achieved in competitive shared tasks and our own experiments demonstrate the strong classification performance of this approach. Further, we identified the optimal ensemble size and the reasons why even an ensemble comprising multiple models of the same type outperforms single models. Then, we turned to real-world application scenarios and investigated explanation methods for neural network models, which aim to increase user acceptance and trust. In collaboration with a large online news provider, we designed, implemented, and evaluated a classifier that combines diverse features from comments, readers, and articles, demonstrating the practical feasibility of semi-automatic comment moderation.

5

Recommending Engaging Comments and Discussions

Only a minority of the comments on online news platforms are toxic, but they still hinder discussions. However, even if we were able to remove all toxic comments, there would still be another obstacle to reader engagement: It is infeasible to keep track of comments and discussions due to their vast number, and thus, the enormous flood of information prevents readers from joining in-depth discussions. Once readers access an online news platform, they are left with countless choices on which article discussions to follow, which comments to read, and where to post a comment or cast a vote. In the following, we address the question of how to automatically recommend articles and comments that are good conversation starters and will attract many active participants.

This chapter is divided into five sections. The first two sections consider reader engagement on the article level: In Section 5.1, we predict which news articles will engage many readers' attention and receive a large number of comments. Section 5.2 is then dedicated to the task of personalized discussion recommendation, where we model individual readers to identify articles that they are likely to discuss. The subsequent two sections consider reader engagement on the comment level. More precisely, Section 5.3 addresses the classification of engaging comments and, to this end, leverages the number of upvotes and replies as a measure of engagement. The focus of Section 5.4 is on the engagement of a special user group: the journalists. We analyze and foster interactions between them and their readership. Finally, Section 5.5 summarizes this chapter.

5.1 Engagement Prediction

Being able to predict which articles will receive a large number of comments would benefit three groups: (1) readers to decide which article discussion to join, (2) news directors to schedule the publication of articles, and (3) moderators to schedule their work. First, readers could get recommendations of the most active discussions, which promise vivid interaction with other readers. Thereby, they would not need to search through numerous discussions to find those where replies are posted within minutes, not hours. Second, news directors could sort articles to be published based on the expected number of comments. They could take this estimation into account in the decision process when to schedule which article for publication. For example, their adjustments could balance the distribution of highly controversial topics across a day, giving not only readers and commenters the chance to engage in every single one, but also evenly distributing the moderation workload. Moderators are the third group that could benefit from knowing which articles will receive many comments. Guiding their main focus of attention towards controversial discussion topics could facilitate efficient moderation and improve the quality of the discussions. According to the news platforms we collaborated with, moderators intervening in a discussion at an early stage help keeping it focused and fruitful.

In this section, we study the task of identifying the weekly top 10 percent articles with the highest comment volume (number of comments). We consider a real-world dataset of seven million reader comments from Zeit Online, described in Subsection 3.1.2. To enrich this unlabeled dataset and increase its meaningfulness, we transfer a classifier trained on the English-language Yahoo News Annotated Comments Corpus (YNACC) [122] to our German-language dataset and leverage the added class labels, such as comment sentiment or tone, in a post-publication prediction scenario. Experiments show that our logistic regression model based on article metadata, linguistic features, and topical features significantly outperforms a state-of-the-art approach.

5.1.1 Identifying the Weekly Top 10 Percent Articles

We address the task of predicting for each news article, whether it belongs to the weekly top 10 percent articles with the highest comment volume. We choose this relative amount to account for seasonal fluctuations and also to even out periods with low newsworthiness. This traditional classification setting enables us to use established methods, such as logistic regression, to solve the task and provide explanations on why a particular article will receive many or few comments.

As a baseline to compare against, we implemented a random forest model with features proposed by Tsagkias et al. [191], such as article publication time and the temperature in Celsius at that time. For our approach, we extend this feature set and categorize the features into five groups: metadata, context, publisher, article headline, and article body. The metadata features include the publication time and whether the article is promoted on the Facebook page of Zeit Online. As context features, we consider temperature and humidity during the hour of publication[1] and the number of competing articles. Competing articles are either only similar articles or all articles published by Zeit Online in the same hour. They compete for readers and their comments. Figure 5.1 visualizes that the number of received comments is not affected by the significantly larger number of published articles on Thursdays. The publication peek on Thursdays is caused by articles that are published in the weekly printed edition and at the same time online. Further, we incorporate publisher features, such as genre, news section, and which news agency served as a source for the article. We include these features to study their impact on the task of reader engagement prediction and not to focus on engineering complex features.

In addition, we propose to leverage the article content itself. Starting with headline features, we use word n-grams of length one to three as well as article keywords provided

[1] as obtained for the three German cities Berlin, Hamburg, and Frankfurt from `www.dwd.de`

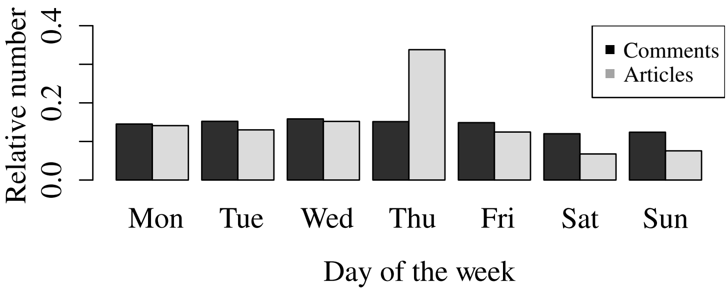


Figure 5.1: At Zeit Online, the number of reader comments is not affected by a peek of article publications on Thursdays.

by the journalists. To capture topical information in the article body and come up with document representations, we rely on topic distributions, document embeddings, and bag-of-words features. To this end, we apply standard latent Dirichlet allocation to model topics [25]. For the document embeddings, we use a Doc2Vec implementation that gives less weight to frequent words [114]. We choose the vector length, number of topics, and window size based on F1-scores on the validation set.

Despite recent advances of deep neural networks for natural language processing, there is a reason to focus also on other models: For the application in newsrooms and the integration into semi-automatic processes, the comprehensibility of the prediction results is very important. A black-box model — even if it achieved better performance — is not helpful in this scenario. Human moderators need to understand *why* the number of comments is predicted to be high or low, and current explanation methods for neural network models cannot sufficiently fulfill this need. The issue of comprehensibility justifies the use of decision trees and logistic regression models, which allow tracing back predictions to their decisive factors.

Automatic Translation of Comments. Whether the first comment is a provocative question in disagreement with the article or an off-topic statement influences the route of further conversation. We assume that this assumption holds not only for social networks [21], but also for comment sections at news websites. Therefore, we consider the tone and sentiment of the first comments received shortly after article publication as an additional feature. Typical layouts of news websites (including the website of Zeit Online) list comments in chronological order and show only the first few comments to readers below an article. Pagination hides later received comments, and most users do not click through dozens of pages to read through all comments. As a consequence, early comments attract a lot more attention, and, with their tone and sentiment, influence comment volume to a larger extent.[2] Presumably, articles that receive controversial comments in the first few minutes after publication are more likely to receive many comments in total.

To classify comments as controversial or engaging, we need to train a supervised classification model, which requires thousands of labeled comments. Such training corpora exist, if at all, mostly for English comments, while the comments in the dataset

[2] We discuss the varying visibility of comments in more detail in Section 5.3.

from Zeit Online are written in German. We propose to apply machine translation and transfer learning to overcome this language barrier: Given a German comment, we automatically translate it into English. From a classifier that has been trained on a labeled English dataset, we can derive labels for the translated comment. The derived labels serve as another feature for our actual task. For each article, we use the labels assigned to the first four comments, which are visible on the first comment page below an article. The first four comments are typically received within few minutes after article publication.

We reimplement the classifier by Napoles et al. [121] and train it on their English dataset, the Yahoo News Annotated Comments Corpus (YNACC). The considered annotations consist of 12 binary labels: addressed audience (reply to a particular user or broadcast message to a general audience), agreement/disagreement with the previous comment, informative, mean, controversial, persuasive, off-topic regarding the corresponding news article, neutral, positive, negative, and mixed sentiment. We automatically translate all comments in our German dataset from Zeit Online into English using the DeepL translation service.[3] For the translated comments, we automatically generate labels based on Napoles et al.'s classifier. Thereby, we transfer the knowledge that the classifier learned on English training data to our German dataset despite its different language. This method builds on the content's similarity across both datasets.

5.1.2 Evaluation and Discussion

The evaluation considers a binary classification task, which is to identify the weekly top 10 percent articles with the largest comment volume. We choose the F1-measure for the evaluation since precision and recall are equally relevant in our scenario. On the one hand, there is a need for a high recall so that no important article and its discussion is overlooked. On the other hand, news platforms have limited resources and cannot afford to moderate each and every discussion. High precision is crucial so that the moderators focus only on articles that need their attention. The experiments are conducted using time-wise splits of the dataset from Zeit Online with years 2014 to 2016 for training, January 2017 to March 2017 for validation, and April 2017 for testing. Table 5.1 lists the results on the validation set. Especially the bag-of-words and the topics of the article body, but also headline keywords and publisher metadata achieve higher F1-score than the metadata features. The highest precision is achieved with the binary feature of whether an article is promoted on Facebook, whereas features comprising the author or competing articles achieve the highest recall.

Table 5.2 lists the evaluation results on the hold-out test set and includes a comparison to the approach by Tsagkias et al. [191]. Our additional article and metadata features, but also the machine-labeled first four comments, outperform the baseline. Due to the diversity of the different features, their combination further improves the prediction results. In comparison to the approach by Tsagkias et al., we achieve an 81 percent higher F1-score.

Table 5.3 lists the results of a baseline feature for comparison. This feature is based on the number of comments received in a short time span after article publication. To

[3] `www.deepl.com`

Table 5.1: Precision, recall, and F1-score of the prediction of weekly top articles on the validation set.

Features	P	R	F1
Metadata	.12	.72	.21
Publication Time	.12	.74	.21
Promoted on Facebook	**.29**	.02	.01
Context	.13	.59	.22
Competing Articles	.11	.94	.20
Temperature and Humidity	.12	.27	.17
Publisher	.17	.85	.28
Author	.11	.96	.19
Genre	.16	.17	.17
News Section	.15	.91	.26
Sources	.10	.38	.16
Medium	.11	.86	.20
Editor	.12	.82	.21
Headline	.15	**.99**	.26
Ngram 1-3 Words	.23	.48	.31
Keywords	.21	.57	.30
Body			
Doc2Vec	.17	.63	.27
Stemmed Bag-of-Words	.27	.61	**.38**
Topic Model	.20	.66	.30

Table 5.2: Precision, recall, and F1-score of a baseline, article and metadata features, machine-labeled first four comments, and all features combined on the test set.

Features	P	R	F1
Tsagkias et al. [191]	.16	.72	.26
Article and Metadata	.26	**.75**	.39
First Four Comments	.29	.50	.36
Combined Approach	**.42**	.52	**.47**

allow for non-linear correlations, we pass the number of comments as an absolute count and a squared count. Further, we use the sequence of the numbers of comments received after 2, 4, 8, 16, 32, and 64 minutes as a combined feature. Comparing the results in Table 5.2 and Table 5.3, the labeled first four comments, article features, and metadata features significantly outperform the baseline until 32 minutes after article publication. After 32 minutes, the number of received comments outperforms every single feature (but not the combination of all features). This turnaround is because the difference between the final number of comments and the number of so far received comments converges over time.

With another experiment, we study the classification error introduced by translation. Therefore, we train two classifiers, each based on the approach by Tsagkias et al. [191]. First, we train and test a classifier on the original, English YNACC. Second, we automatically translate all comments in YNACC from English into German and use this translated data for training and testing of the second classifier. Comparing these

Table 5.3: F1-score of the prediction of weekly top articles based on the number of comments received in the first minutes after article publication.

Time since Publication	F1
2 min	.03
4 min	.03
8 min	.17
16 min	.33
32 min	.41
64 min	.45
Combined Sequence	**.46**

two classifiers, we find that both precision and recall slightly decrease after translation, as shown in Table 5.4. Based on this result, we can assume that translating German comments into English introduces only a small error. Although the two datasets differ in language, we can transfer a classifier that has been trained on YNACC to the dataset from Zeit Online.

Table 5.4: Translation from English into German slightly affects precision and recall.

	English		German	
Label	P	R	P	R
Audience	.80	.80	.81	.82
Agreement	.76	.18	.65	.09
Informative	.55	.71	.51	.85
Mean	.64	.52	.52	.37
Controversial	.61	.90	.58	.94
Disagreement	.60	.75	.58	.81
Persuasive	.51	.89	.44	.97
Off-Topic	.67	.57	.66	.40
Neutral	.68	.35	.62	.41
Positive	.46	.13	.80	.10
Negative	.70	.93	.71	.92
Mixed	.45	.52	.40	.78

5.2 Discussion Recommendation

In this section, we aim to encourage engagement by recommending selected article discussions to individual platform users. These recommendations can lead to higher user loyalty, higher retention rates, and increased website traffic. Further, they can make the group of users who contribute to a discussion more diverse by encouraging those who would otherwise stay passive. The personalization is contrary to the previous section, where the idea was to recommend the (same) most popular discussions to every user. The recommendations in this section are custom-tailored instead of being based on the community's preferences. To this end, we model users and their participation in discussions using comment texts and commenter co-occurrence. We propose the model

HyCoNN (Hybrid Cooperative Neural Networks), which jointly learns representations of users and discussions, and conduct experiments on datasets from Daily Mail and The Guardian. In contrast to previous work in the related domain of product reviews and rating prediction, e.g., DeepCoNN (Deep Cooperative Neural Networks) [221], we combine content-based and user-co-occurrence-based approaches. For the offline evaluation of the recommendation performance, we use a ranking task: Given a specific time and user, we reconstruct the state of the reader discussions active at that time. We then rank these discussions based on the estimated probability that the user posts a comment in a particular discussion.

5.2.1 HyCoNN Recommender System

We introduce HyCoNN, which combines a content-based and a user-co-occurrence-based recommendation approach. To this end, we first adapt the DeepCoNN model [221] to the task of discussion recommendation. Second, we describe how to learn user embeddings with node2vec [76] on a graph of user co-occurrences in discussions. Finally, we combine both approaches in our proposed HyCoNN architecture, visualized in Figure 5.2.

Adaptation of DeepCoNN. DeepCoNN was originally designed for item rating prediction in the domain of product reviews. Therefore, its architecture requires some adaptations to our task. We use a CNN architecture for text processing similar to the one proposed for DeepCoNN. The first layer translates every word in the input sequence into its corresponding pre-trained word embedding using a lookup table. It is based on 300-dimensional FastText word embeddings, which were pre-trained on the English-language common crawl dataset [28]. The resulting embeddings serve as input to a convolutional layer, which applies filter kernels to process groups of neighbored words. After the convolution, we apply a rectified linear unit (ReLU) activation function to each neuron's output and reduce the number of extracted features by using a max-pooling operation. Finally, the concatenation of all max-pooling outputs traverses a dense layer with another ReLU activation function.

The original DeepCoNN model uses two CNNs to predict a user's rating for an item. We adapt this model to our problem in the following way: We use the user's past comments as input to the first CNN. The output of that CNN serves as a user representation. The second CNN receives as input the concatenated texts of all comments of an article discussion. The output is the corresponding discussion representation. A dropout layer regularizes the outputs of both CNNs.

The concatenation of the user representation and the individual discussion representation is the input to a factorization machine [143] implemented as proposed by Zheng et al. [221]. A sigmoid function maps the output of the factorization machine to a real number between 0 and 1 so that it can be interpreted as the probability that the user posts a comment in the given discussion. We refer to that adapted model in the following as DeepCoNN.

Utilizing User Co-Occurrences. Commenting on online news platforms is a highly social activity and involves interacting with other users' comments. By leaving a comment, users implicitly join a community of people who share an interest in that particular

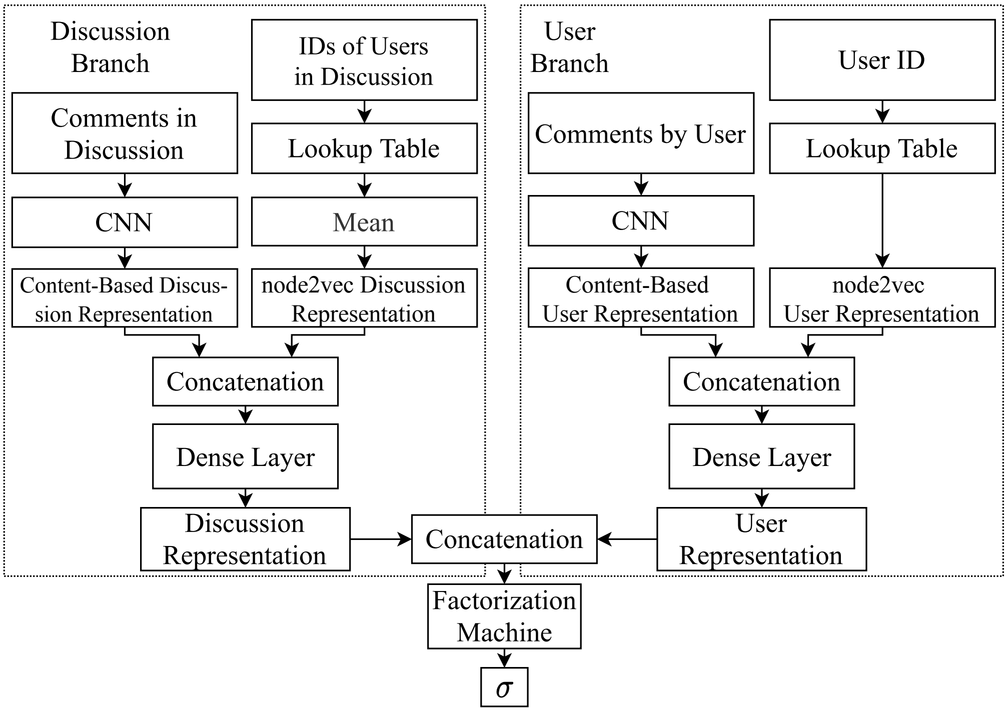


Figure 5.2: HyCoNN comprises a content-based and a user-based model branch.

discussion. Users who frequently participate in the same discussions are similar in the way that they co-occur in our dataset. If we encode this information of co-occurrence, we can compute the similarity between users who already posted a comment in a discussion and the user for whom we want to generate recommendations.

To leverage this kind of information, we create an undirected bipartite graph. Each node represents either a user or a discussion. A user node has an edge to a discussion node if the user participates in this discussion. Consequently, the length of the shortest path between the nodes of users who co-occurred in a discussion is two. On this graph, we learn user embeddings with node2vec [76]. In the resulting embedding space, embeddings of users who often co-occur in discussions because of rivalry, friendship, or shared interests appear closer to each other.

HyCoNN Architecture. Figure 5.2 depicts how we incorporate the learned user embeddings into the approach of jointly learning representations of discussions and users with DeepCoNN. We name this model HyCoNN (Hybrid Cooperative Neural Networks) since it combines the content-based DeepCoNN (Deep Cooperative Neural Networks) with the user co-occurrence-based embeddings learned with node2vec. Our model consists of two branches: one to model discussions, i.e., the comments of articles, and one to model users.

The integration of the node2vec user embedding into the neural network's user branch is accomplished with a lookup table that translates user IDs into their corresponding embedding. The concatenation of the node2vec user embedding and the user representation learned from the text content using the CNN then traverses a dense layer with ReLU as its activation function. The output of this layer is the user representation.

The discussion branch starts with the IDs of all users who posted a comment in a specific discussion until a given point in time. The same lookup table as before translates these IDs into the learned node2vec embeddings, and the mean of all user embeddings serves as the node2vec representation of the discussion. This representation is then concatenated with the content-based discussion representation, which the CNN calculates based on the comments in the discussion. The result traverses another dense layer with ReLU activation, which finally outputs the discussion representation. For regularization, we apply dropout to the output of the dense layers of both branches.

Similar to the DeepCoNN architecture, HyCoNN uses a factorization machine to process the concatenation of the discussion representation and the user representation. As in our DeepCoNN adaptation, the output finally traverses a sigmoid function so that it can be interpreted as the probability that the user posts a comment in that discussion.

5.2.2 Evaluation

Our main experiment is based on a recommendation task for a hold-out set of comments, where we aim to predict in which discussion a user posts a comment. In addition to that, we evaluate the user embeddings of the different approaches independently from the prediction task, with regard to the distance of similar users in the embedding space.

Data Sampling

For the evaluation, we consider two real-world comment datasets from The Guardian and Daily Mail, which were introduced in Section 3.2. We created two subsets of the data to (1) model only users who are still active, (2) limit the dataset to comments that were posted under similar conditions (no platform changes, e.g., introduction of stronger moderation strategies), and (3) obtain time-wise training, validation, and test set splits, where every user who appears in the test set or the validation set also appears in the training set. To ensure a realistic evaluation setting, we select appropriate relevant (positive) and irrelevant (negative) samples of discussions for each user. *Appropriate* means that we consider only those discussions and only those comments that were available at the time when the user visited the website.

For achieving this objective, we first sort all comments by publication timestamp of the corresponding news article. Then, we do a time-wise split. Thereby, during training, the model has no access to information from the future. Further, to avoid inconsistencies that could result from a new moderation policy introduced by The Guardian in 2016, we limit the dataset to 2017. For The Guardian training dataset, we select articles and comments published between March 1st and May 31st, 2017. For the validation set, we choose the time between June 1st and June 30th, and for the test set the time between July 1st and July 31st, 2017.

The dataset from Daily Mail is much larger than the one from The Guardian. Therefore, we set smaller time frames for Daily Mail to obtain similar-sized datasets from both news platforms. Consequently, we set the time frame for the training dataset

Table 5.5: Size of comment datasets from The Guardian and Daily Mail.

	The Guardian			Daily Mail		
	Users	Comments	Articles	Users	Comments	Articles
Training	111,961	2,278,816	13,419	128,927	2,892,083	22,906
Validation	63,042	690,229	4,004	98,413	1,415,855	10,070
Test	66,143	744,918	4,223	99,474	1,508,050	11,315

to April 1st to May 31st, for the validation set to June 1st to June 15th, and for the test set to June 15th to June 30th, 2017. For both datasets, The Guardian and Daily Mail, we picked users for validation and testing who appear at least four times in the training dataset. Table 5.5 lists the sizes of the resulting datasets.

Recommendation Task

The evaluation of the recommendation performance uses a hold-out set of discussions and simulates the website's state at the time the user actually posted a comment. For every moment in time when the user posted a comment in the test dataset, we recreate the state of the corresponding discussion at that time without the newly posted comment as a positive sample (a discussion where the user has contributed) and of 50 randomly chosen negative samples (discussions where the user has not contributed). The negative samples correspond to the state of reader discussions at the time the user posted the comment in the positive sample. We consider different top-k recommendation settings with $k \in \{1, 3, 5, 10, 15\}$ and assess the performance with regard to precision and recall at k. In our scenario, recall is more important than precision because a single discussion is labeled as relevant, but others — implicitly labeled as irrelevant — might actually also be relevant and might have been overlooked by the reader. Some related work uses ROC-AUC as an evaluation metric in similar scenarios [18, 177]. It corresponds to the probability that a relevant discussion is ranked higher than an irrelevant one. If a recommender system ranks a relevant discussion higher than most of the irrelevant discussions, it achieves a good ROC-AUC score. However, precision and recall at k are better suited to evaluate that the top-ranked recommendations are relevant. A problem with ROC-AUC is that different recommender systems achieve similar scores if the same discussions are irrelevant to most users and, therefore, can be easily identified and ranked down, i.e., discussions about unpopular topics [177].

For data pre-processing, we pass all comments through a word tokenizer and lowercase every token. A single vocabulary is created for all methods based on the training dataset to have a fair comparison of the neural network models and the baselines. Any token that occurs in more than 50 percent of the comments or in fewer than five comments is discarded. Mimicking a realistic application scenario, we strictly limit the learning of user representations to comments from the training dataset. In contrast to that, Catherine and Cohen [35] describe that DeepCoNN's predictions of item ratings are only good if the review text by the target user for the target item is already known. In our training process, we further omit a comment for computing the user representation if the user posted this comment in the discussion that represents the positive training sample. The omission of those comments ensures that the model does not learn direct relations between comments by that particular user and the respective discussions. Otherwise, predictions on the validation or test dataset would not be comparable since those relations only appear in the training dataset.

The validation of node2vec uses a pairwise ranking task. We examine the similarity between a given user u and (1) a corresponding reader discussion that functions as a positive sample and (2) a corresponding reader discussion that serves as a negative sample. If the similarity between the user embedding of u and the mean user embedding of the positive sample is higher than the similarity between the user embedding of u and the mean user embedding of the negative sample, the ranking is correct; otherwise, it is incorrect. During test time, we obtain recommendations by calculating ranking scores based on the cosine similarity of a user embedding and the mean user embeddings of participants in a given discussion.

Baselines. A tf-idf vector space model serves as a baseline approach. This model aims to rank those reader discussions higher that are more similar to the comments the user wrote in the past. To this end, we use the previously created vocabulary and calculate the inverse document frequency for all terms in the training dataset. For the user representation, we average the tf-idf vectors of the user's comment texts in the training dataset. For the representation of a discussion, we average the tf-idf vectors of all comments present in that discussion at a particular point in time. Finally, the cosine similarity between the user representation and the discussion representation corresponds to the ranking score.

For a collaborative filtering (CF) baseline, we build a matrix of users and discussions, where each row corresponds to a user, and each column corresponds to a discussion. Each cell describes how many times the user posted a comment in the discussion. This baseline computes higher ranking scores for those discussions, where the mean of the participants' representations is more similar to the user. While a user representation can be retrieved directly from the corresponding row in the matrix, discussion representations need to be calculated as the mean of all representations of users who participated in a particular discussion. The ranking score is determined by calculating the cosine similarity between the mean representation of the participants and the representation of the user.

Since HyCoNN combines a content-based method and a user-based method, we use a fusion strategy to compare to the combination of the two baselines, content-based tf-idf and user-based CF. To this end, we apply Reciprocal Rank Fusion (RRF) [47] for combining the individual rankings. Moreover, we also use RRF to combine the rankings produced by our node2vec-based approach and the DeepCoNN approach to compare whether the combination of both methods in HyCoNN is superior to a rank fusion strategy that combines their individual results. We refer to that approach as NDRF (Node2vec DeepCoNN Rank Fusion) and to the combination of tf-idf and CF as BRF (Baseline Rank Fusion).

User Representation and Graph Construction. For each news article in the test datasets, we recreate the corresponding discussion for at least one positive sample and 50 negative samples. The resulting dataset includes 53,185 different states of reader discussions for The Guardian and 59,125 for Daily Mail. We set the maximum number of comments to represent users to 42 for The Guardian and to 22 for Daily Mail. With these limits, we are able to represent 90 percent of the users in the training dataset from The Guardian and 80 percent of the users in the training dataset from Daily Mail with all their comments. For the minority of users who wrote more than 42, respectively 22, comments in the training set, we choose their most recent comments to represent them. The comments are sorted by descending timestamp and concatenated afterward so that their temporal order in the reader discussions is maintained.

To construct the bipartite graph, we use the 42 newest comments for The Guardian and the 22 newest comments for Daily Mail for each user in the respective training dataset. Every user is included in the training dataset, no matter the number of posted comments. While we make recommendations only to users who wrote at least four comments in the training dataset, other users are still included also in the validation and test dataset. Thereby, the embeddings of these users with fewer than four comments can contribute to the discussion representation and can improve the prediction and recommendation performance for other users in the validation set and test set.

We consider three variations of the CF baseline to allow for a fair comparison of CF and node2vec. CF42 and CF22 refer to two variations that include only the 42 and the 22 newest comments to represent users, and CF refers to the variation that includes *all* comments. Moreover, we also generate representations of users with fewer than four comments in the training dataset, similar to our approach with the learned user embeddings using node2vec.

Hyperparameter Optimization. For node2vec, we use Bayesian optimization to tune the number of walks per source $\in \{10, 20, 30\}$, the walk length $\in \{10, 20, 30, 50, 100\}$, the context window size $\in \{10, 20, 30\}$, and the embedding size $\in \{25, 50, 100\}$ on the validation dataset. On the dataset from The Guardian, this optimization leads to 20 walks with a length of 10, a window size of 10, and 25-dimensional embeddings. We give equal weight to local and global structures by setting node2vec's hyperparameters p and q to the default value 1. The training of DeepCoNN and HyCoNN uses the binary cross-entropy loss and the Adam optimizer. For tuning the hyperparameters of DeepCoNN, we use Bayesian optimization with ten steps. The search space comprises the number of neurons in the convolutional layer $n \in \{25, 50, 100\}$, the window size $o \in \{2, 3, 4\}$, and the latent factors $l \in \{25, 50, 100\}$. We manually set the learning rate to 0.0001, the dropout to 0.1, and the batch size to 100. The model achieves the best accuracy on the validation dataset with $n = 50$, $o = 2$, and $l = 100$ with two training epochs. HyCoNN reuses the hyperparameters of DeepCoNN and the user embeddings learned with node2vec. Furthermore, we initialize the weights and biases of the CNNs with the ones from the trained DeepCoNN model and keep the learning rate, batch size, and dropout. We tune only the number of neurons $r \in \{25, 50, 100, 150\}$ in the dense layer, which corresponds to the user and discussion embedding size. The model achieves the best accuracy after three epochs with $r = 100$.

We follow the same tuning approach on the dataset from Daily Mail. For node2vec, Bayesian optimization leads to the same settings as on the dataset from The Guardian. DeepCoNN uses the previous settings for learning rate, dropout, batch size, and Bayesian optimization. It leads to $n = 100$ neurons in the convolutional layer, a kernel size of $o = 3$, and $l = 50$. The best accuracy is achieved after training for one epoch. HyCoNN

Table 5.6: Precision and recall @k for $k \in \{1, 3, 5, 10, 15\}$ for the recommendation task on the dataset from Daily Mail.

	@1		@3		@5		@10		@15	
	P	R	P	R	P	R	P	R	P	R
CF	.18	.18	.11	.32	.09	.43	.06	62	.05	.75
CF22	.17	.17	.10	.31	.08	.41	.06	.58	.05	.70
Tf-idf	.10	.10	.08	.23	.07	.33	.05	.51	.04	.65
BRF	.18	.18	.12	.36	.10	.48	.07	.66	.05	.77
Node2vec	.15	.15	.11	.32	.09	.43	.06	.63	.05	.76
DeepCoNN	.14	.14	.10	.31	.09	.43	.06	.62	.05	.74
NDRF	.19	.19	.13	.39	.10	.52	.07	.71	**.06**	.82
HyCoNN	**.22**	**.22**	**.15**	**.46**	**.12**	**.59**	**.08**	**.78**	**.06**	**.87**

reuses the hyperparameters of DeepCoNN and the embeddings of node2vec. The model achieves the best results with $r = 50$ after one epoch.

Results. Table 5.6 lists precision and recall at $k \in \{1, 3, 5, 10, 15\}$ for the recommendation task on the dataset from Daily Mail and Table 5.7 on the dataset from The Guardian. Recommending random discussions achieves a recall@1 of 0.02, a recall@3 of 0.06, etc. on that recommendation task because there are always 51 samples with one being relevant and 50 being irrelevant. The results show that combining DeepCoNN and node2vec in HyCoNN results in better recommendations than applying the methods individually. On the dataset from Daily Mail, HyCoNN outperforms all other methods for every k. HyCoNN also yields better results than NDRF on both datasets.

However, the poor performance of node2vec for smaller k also results in CF outperforming HyCoNN for $k = 1$ and $k = 3$ on the dataset from The Guardian. For larger k, node2vec learns competitive embeddings. It achieves even better results than CF for top-k recommendations with $k \geq 10$. A remarkable point is that the rank fusion strategy BRF, which combines tf-idf with CF, results in worse recommendations on the dataset from The Guardian than using CF alone. Tf-idf and CF generate very different rankings and their combination in BRF results in worse performance. However, BRF on the dataset from Daily Mail yields better results than the baselines alone. The content-based methods DeepCoNN and tf-idf perform worse than the CF method on both datasets.

User Embedding Evaluation

For an additional, quantitative evaluation of the user embeddings on the dataset from The Guardian, we adapt the pair-distance correlation [24]. We call our adapted method pair-similarity correlation (PSC) to distinguish it from the existing pair-distance correlation. Blandfort et al. [24] calculate the distance between two users in a user-space as the mean-squared difference of their ratings on movies. To make use of this idea, we construct a user's implicit discussion rating by calculating the number of times this user posted a comment in this discussion. However, these representations are very sparse.

Table 5.7: Precision and recall @k for $k \in \{1, 3, 5, 10, 15\}$ for the recommendation task on the dataset from The Guardian.

	@1		@3		@5		@10		@15	
	P	R	P	R	P	R	P	R	P	R
CF	**.35**	**.35**	**.16**	**.48**	.11	.56	**.07**	.68	.05	.76
CF42	.34	.34	.15	.46	.11	.53	**.07**	.66	.05	.75
Tf-idf	.13	.13	.08	.25	.07	.32	.05	.47	.04	.58
BRF	.25	.25	.14	.41	.10	.51	**.07**	.67	.05	.77
Node2vec	.27	.27	.15	.44	.11	.53	**.07**	.68	.05	.78
DeepCoNN	.13	.13	.10	.30	.08	.41	.06	.60	.05	.72
NDRF	.24	.24	.15	.44	.11	.55	**.07**	.72	**.06**	.82
HyCoNN	.26	.26	**.16**	.46	**.12**	**.58**	**.07**	**.74**	**.06**	**.84**

We address this problem by creating *user section vectors*, which utilize fine-grained news sections of articles, such as politics, sports, or environment. For every news section in the validation dataset, we count the number of times a user posted a comment on articles in that section. These user vectors have a length of 47 since there are 47 different news sections in the validation dataset. Similar to Blandfort et al. [24], we take the similarities of the embeddings in the user section space as the ground truth and compare them with the similarities of user representations learned by HyCoNN and DeepCoNN.

To this end, we use the cosine similarity to calculate the similarity of users in the user section space and the similarity of users in the respective embedding space that we want to evaluate. To calculate PSC, we create a list of user pairs. The list includes the same number of user pairs that either (1) co-occur in at least five different discussions or (2) co-occur in fewer than five discussions. This policy ensures that user pairs with similar and dissimilar commenting history occur equally in the list. The result contains 510,334 different user pairs. The pair-similarity correlation is computed as the Pearson correlation between the similarity scores of user pairs in the user section space (ground truth) and the respective embedding space to be evaluated. Hence, the best possible score is 1, and the worst is -1. A random embedding would achieve a score of 0.

With a PSC score of 73.8 percent, HyCoNN best preserves the similarities of users' interests in categories on The Guardian. There is almost no difference in the performance of node2vec (69.5 percent) and DeepCoNN (69.4 percent), which outperform CF (59.1 percent) and tf-idf (37.6 percent). Note that this evaluation only compares user embedding approaches and, therefore, cannot include the rank fusion approaches from the recommendation task. Comparing the PSC results of CF with its recommendation results leads to the conclusion that good results in the recommendation task do not necessarily imply that the user embeddings preserve the similarities in the user section space. This finding is in line with Blandfort et al. [24]. The biggest difference to their method is that we compare similarities of embeddings in two different vector spaces. Further, we not only evaluate vectors learned by neural networks but also vectors based on collaborative filtering and tf-idf.

5.2.3 Discussion

On the dataset from Daily Mail, HyCoNN outperforms all other approaches for every k and on the dataset from The Guardian for $k \in \{5, 10, 15\}$ with regard to precision@k and recall@k. To our surprise, for $k = 1$ and $k = 3$ on the dataset from The Guardian, the CF baseline outperforms all other approaches. One reason might be our sampling strategy, which uses only implicit information to select negative samples. A user did not necessarily encounter every reader discussion that we selected as a negative sample. For instance, maybe the user did not encounter a discussion just because the corresponding article was not displayed on the main page of the news platform when the user was active. As a consequence, discussions that we assume to be irrelevant could likely be relevant, even though the user did not post a comment. Since the models only perform worse for $k = 1$ and $k = 3$, it could likely be that some negative samples are, in fact, good recommendations. Therefore, although the models perform worse according to the evaluation metric, the recommendations might still be valuable.

Since the combination of CF and tf-idf in BRF performs worse for The Guardian than for Daily Mail, a hybrid recommendation method for The Guardian is not necessarily the best strategy, which is also reflected in the results of HyCoNN. In contrast, on the dataset from Daily Mail, BRF achieves better results than tf-idf, and CF individually. We conclude that the hybrid recommendation methods, such as the model we propose, do not necessarily lead to much better results on every dataset.

The proportion of users in the test dataset that node2vec can represent with user embeddings affects the recommendations of node2vec, HyCoNN, and NDRF. In the test dataset from The Guardian, 90 percent of the comments were written by users who appear in the training dataset. Respectively, users in the training dataset from Daily Mail wrote 89 percent of the comments in the corresponding test dataset. Since node2vec and CF can represent every user appearing in the training dataset, we can rule out that the proportion of users in the test dataset, for whom user embeddings exist, is affecting the results when comparing both evaluation datasets.

The results show that the user embeddings learned with node2vec on the proposed bipartite graph are useful for recommendations and that their performance is on the same level as CF. However, memory-based CF strategies need a lot of runtime and memory, especially if the number of news articles is large. The approach we propose with node2vec overcomes these problems as it represents users in a 25-dimensional embedding space, which is, compared to the CF baseline, low-dimensional.

We can conclude that jointly modeling the user representations and discussion representations in the DeepCoNN architecture yields better results than a naive content-based approach, e.g., tf-idf. Finally, HyCoNN consistently outperforms DeepCoNN, node2vec, and NDRF. This result means that learning from content while incorporating the user embeddings with HyCoNN outperforms not only the individual approaches but also their combination with a rank fusion strategy. The strong PSC score in our second experiment shows that the learned user embeddings are not only tailored to the prediction task. They also preserve the similarities of users who share an interest in specific article categories. As a limitation, note that the cosine similarity is affected by the curse of dimensionality. With increasing dimensionality, the calculated distance between different pairs of points becomes almost equal. In particular, the user embeddings generated with tf-idf and CF could be vulnerable to this problem because of the high dimensionality of their embeddings. However, given the strong PSC of CF, we assume that it is not an issue. Therefore, we refrain from further optimizing tf-idf, e.g., with the help of a dimensionality reduction method.

5.3 Engaging Comment Classification

Most online news platforms display reader comments in chronological order — with a few exceptions, such as *Slashdot.org* and *Digg.com*. While one might argue that this approach is transparent and fair, it does not foster engaging discussions. Instead, it only gives an incentive to post comments as fast as possible after an article is published. In that case, the comment will get ranked high, gain visibility in the community, and possibly get some reactions to the comment, no matter its content. This competition goes so far that some users refrain from reading the article to be the first to post a comment. Chronological comment ranking neglects that some comments are more engaging than others and are better conversation starters. In this section, we address the task of automatically classifying the most engaging (top) and the least engaging (flop) comments. To this end, we systematically analyze reader engagement in the form of the upvotes and replies that a comment receives on a dataset from The Guardian. For illustration purposes, we list two comments that generated a large amount of engagement in the form of many upvotes or replies:

1. "The brexiters are achieving their wish: they're turning the UK into the kind of second rate country they can feel at home in." 2,615 upvotes, 3 replies
2. "Can somebody please explain to me why some people are so rabidly anti-gay marriage?" 82 upvotes, 20 replies

The first comment refers to an anticipated loss of 1,000 jobs in EU authorities located in the UK as a consequence of Brexit. The number of received upvotes is extraordinarily high, presumably because many anti-brexiters identify with the expressed opinion. The second comment was posted half a year before the UK parliament legalized same-sex marriage. It was a topic of controversial discussions at that time, with replies containing different opinions and addressing the user's request for an explanation.

There is an inherent position bias in the numbers of upvotes and replies, which complicates their analysis: Earlier comments receive, on average, more upvotes and replies. After removing this bias with a method that we specifically designed for comments on online news platforms, we reveal the differences between top and flop comments. Further, we visualize words that occur more often in either top or flop comments. Last but not least, we introduce a taxonomy to categorize different types of engaging comments systematically.

5.3.1 Biases in the Number of Upvotes and Replies

We assume that a comment receiving many upvotes or replies is relevant to many users, whereas a comment with no or only a few reactions is comparably irrelevant. However,

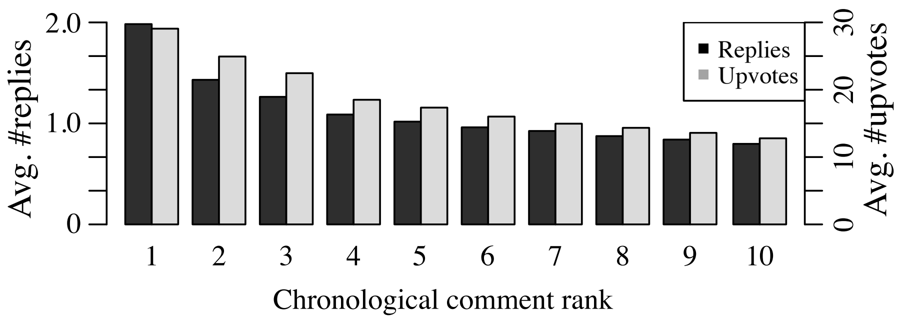


Figure 5.3: A comment's average number of received upvotes and replies correlates with its chronological rank in the discussion thread, which indicates a position bias.

this assumption only holds if malicious intentions to manipulate the user votes, e.g., voting multiple times with fake user accounts, can be ruled out. In our dataset with 260 million upvotes from The Guardian, there is no incentive for users to manipulate the upvote count because it does not influence the order in which comments are displayed. Occasional hoax upvotes can be neglected and considered noise. This leaves us with the vast majority of upvotes actually presenting an engagement signal on the level of individual comments. Still, the number of upvotes and replies is biased by a comment's visibility to readers, which is influenced by the comment's position in the chronological ranking and the article's popularity.

Figure 5.3 visualizes the position bias and reveals that the first comment receives, on average, almost twice as many upvotes and replies as the tenth comment. In line with related work [82], we attribute the advantage of earlier comments to their greater exposure to more readers. For this reason, the raw upvote and reply count is not enough to judge a comment's relevance to users in comparison with other comments. That is why we propose a method to normalize the counts and, thereby, prevent the position bias from distorting the results. As a side benefit, the effect of an article's popularity on the upvotes and replies is also removed.

In short, this method transforms the absolute counts to relative numbers and afterward groups all comments across different article discussions by their rank. For example, we compare a comment at rank 3 to all comments that appeared in other discussions at the same rank 3. Let us assume that the comment received 20 percent of all upvotes in its corresponding article discussion. If comments at rank 3 receive, on average, fewer than 20 percent of the upvotes, we have identified a top comment, otherwise a flop comment. We describe this method in more detail in the following.

News platforms sort comments chronologically and show only the first few, e.g., ten, comments to readers directly below an article text. All subsequent comments are hidden by pagination, which the user can access by browsing to the next pages. In practice, most users access only the very first page, which, by default, shows the oldest comments. They never read any subsequent comments. For this reason, we consider only the first ten comments directly below each article, ensuring that they were seen (and judged) by many readers. Articles with fewer than ten comments are discarded to allow for a fair comparison.

Some news articles draw more attention than others. Thus, they attract a varying number of readers who eventually consider voting on and replying to comments. To normalize this variation, we transform the absolute number of upvotes and replies into relative numbers within each article's discussion. To also remove the position bias illustrated in Figure 5.3, we group all comments across all articles by their rank. The result comprises ten groups of equal size. We sort the comments of each rank by the descending relative number of upvotes. Each sorted list now contains the comments in a normalized way: All comments in the top 50 percent of the list perform better than an average comment at this rank, which means they received a comparably large portion of upvotes. All comments in the bottom 50 percent received fewer upvotes than an average comment at this rank. Thereby, the list contains top comments and flop comments with regard to upvotes, which can be used as positive and negative training samples for supervised learning.

There is only one variation for processing the replies. Articles that received fewer than 20 replies on their first ten comments are discarded. In the same way as before, we then sort the comments of each rank by the descending relative number of replies. Splitting each list into halves results in sets of comments that receive more or fewer replies than an average comment at the respective rank. By further filtering the dataset, e.g., to only the top 10 percent and bottom 10 percent, we consider only comments that perform much better or much worse than average. This step can be seen as a way to filter for a higher agreement on a comment's rating among users. Typically, upvotes and replies exhibit a low agreement: Users do not agree on which comments deserve upvotes or replies. However, the agreement in the top 10 percent and bottom 10 percent is higher by our definition of this subset of the data. A much higher, respectively, much lower relative number of users reacted to the comments in these smaller sets.

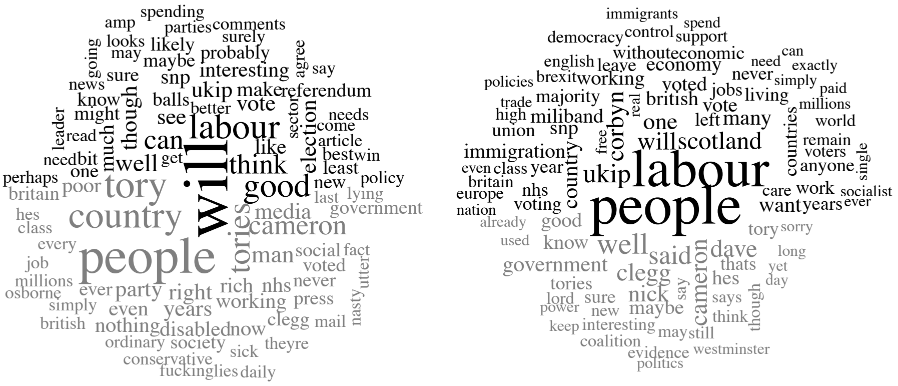
5.3.2 Differences of Most and Least Engaging Comments

Based on our method, we distinguish between two sets comprising the most and least engaging (top and flop) comments and analyze their differences. As user engagement varies by news topic [4], we reduce the topical variety by limiting our analysis to comments on articles in the politics section. It is the section with the largest number of comments received. Table 5.8 compares the most and least engaging comments with regard to their average length, readability, and sentiment. Comments that generate less engagement are, on average, shorter and more often have a neutral sentiment. However, there is no difference in readability or the use of function words and personal pronouns. We use the automated readability index (ARI) to evaluate the readability. It is a standard metric that takes into account the number of words per sentence and characters per word.

Figure 5.4 compares the usage of the 100 most frequent words in comments that received the most or the least upvotes or replies. The word clouds display a word in the top half (black font) if it occurs more often in the most engaging comments and in the bottom half (gray font) if it occurs more often in the least engaging comments. The font size corresponds to the difference in the word's relative frequencies in both classes. For example, the relative frequency of the word *Labour* is 0.39 percent in comments that receive the most replies and 0.27 percent in comments that receive the least replies. The

Table 5.8: The most and the least engaging comments differ in length and amount of neutral sentiment.

	Upvotes		Replies	
Average per Comment	Most	Least	Most	Least
Number of Words	75.54	43.68	76.82	38.52
Readability Index	9.82	9.08	9.50	9.14
Rate of Function Words	.43	.43	.44	.43
Rate of Personal Pronouns	.13	.12	.12	.13
Positive Sentiment	.47	.48	.49	.45
Neutral Sentiment	.07	.23	.09	.20
Negative Sentiment	.46	.30	.42	.34



(a) Upvotes

(b) Replies

Figure 5.4: Comparison word clouds show indicative words for classes of the most (black, top) and least (grey, bottom) engaging comments. For example, comments that mention *people* receive few upvotes but many replies.

comparably large difference between these frequencies is illustrated by the word's large font size.

The most engaging comments mention the word *Labour* more often and the word *Tory* less often. The same relation holds for politicians of the respective parties, e.g., for Jeremy Corbyn (Labour) and David Cameron (Tory). A reason for this might be the political orientation of *TheGuardian.com* readers: according to a post-election survey, 73 percent voted for the Labour party and 8 percent for the Tory party in the 2017 UK general election.[4] The readers tend to upvote comments about their preferred party more often than comments about the opposite Tory party. This bias exemplifies why upvote counts cannot readily be used to distinguish high-quality from low-quality comments. Upvotes are cast with a subjective opinion in mind rather than with an objective and unbiased view of the comment text only. Comments that mention the word *people* are another interesting example. These comments receive few upvotes but many replies, probably because they make generalized claims about groups of people, which are con-

[4] `www.yougov.co.uk/topics/politics/articles-reports/2017/06/13/how-britain-voted-2017-general-election`

troversial and serve as conversation starters. They are comparably unpopular on the platform but trigger many disapproving replies. Two examples are:

1. "The people who voted for the war should be sent to prison as well."
2. "People are disillusioned with mainstream politics, and are starting to look elsewhere."

5.3.3 Taxonomy of Engaging Comments

Taxonomies have been proposed for hateful comments [171, 206] but not for engaging comments. To foster a better understanding of engagement triggers, we propose a taxonomy for engaging comments, which is shown in Figure 5.5. We follow an open coding approach, also used by Salminen et al. [171], and code 1500 engaging comments. With this approach, we organize classes in a conceptual hierarchy. Table 5.9 exemplifies each class with a sample comment. For example, the class *Question* groups the subclasses *Explanation*, *Opinion*, and *Fact* together because all of them generate engagement by requesting answers in the form of comment replies. Comments in all three subclasses typically contain a question mark. Note that the example comments for other classes, such as *Joke/Humor* and *Speculation,* also contain questions. However, these questions are more of a rhetorical nature, and the corresponding comments trigger engagement for other reasons. The taxonomy also distinguishes between comments that trigger only upvotes, replies, or both. For example, while comments with jokes rarely receive replies, they frequently receive upvotes. It is the opposite if a comment asks for other users' opinions. However, if a comment dissents from a news article, other users express their approval or disapproval with both upvotes and comments. Our taxonomy is constructed in particular for comments from The Guardian and is by no means universal. Other platforms might exhibit other classes of engaging comments, for example, if they allow users also to downvote comments. We revisit our taxonomy in the evaluation to understand which types of engaging comments are especially challenging to detect automatically.

5.3.4 Distinguishing Top and Flop Comments

We present a neural network model to distinguish top and flop comments based on their text. Instead of labeling comments in a time-consuming process, we draw upon the bias-corrected number of upvotes and replies that a comment received. Given the positive and negative training samples (top and bottom 10 percent engaging comments) for supervised learning, we describe the architecture of our neural network model. While we train two separate models, one for upvotes and one for replies as the measure to distinguish top and flop comments, the models have the same architecture. We propose a recurrent neural network model based on GRUs. The network begins with a FastText word embedding layer, which uses fixed weights, pre-trained on all comments from The Guardian, as described in Subsection 3.2.4. The same pre-trained word embeddings are used for both tasks, and thereby, the learned word representations are shared across them.

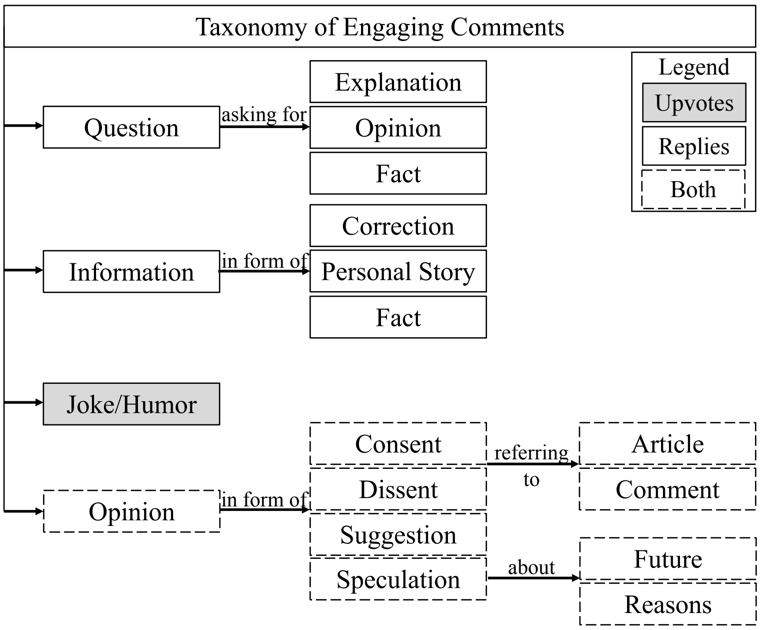


Figure 5.5: Our hierarchical taxonomy of engaging comments classifies comments that attract upvotes (grey fill), replies (solid outline) or both (dashed outline).

Table 5.9: Examples for each class in our taxonomy of engaging comments.

Question	
Explanation	"Can anyone explain (serious question!) what the long term economic plan is?"
Opinion	"Let's take a poll guesses: What do you think the outcome will be in 17 days [...]"
Fact	"[...] which [celebrities] are true supporters of Nigel Farage?"
Information	
Correction	"[...] is a herbicide, not an insecticide. Please correct this"
Personal Story	"The tiered system is incredibly unfair. I have a 16 yr old son, who is extremely [...]"
Fact	"In 2011 [...] 700,000 new National Insurance numbers were issued to foreign nationals."
Joke/Humor	"What is the difference between UKIP and a tandem? A tandem has two seats."
Opinion	
Dissent (Article)	"I don't like the way this article appears to link Cameron With the No recovery in the polls [...]"
Consent (Comment)	"I agree with [username] that no one can believe a word that he says."
Suggestion	"We need more [...] people that care about the country [...]"
Speculation (Future)	"[...] immigration during the next 5 years could be 1.5 – 2 million. Anyone want to argue that will not affect housing [...?]"
Speculation (Reasons)	"The reason for the Palestinians wanting to have a vatican-like status at the UN is [...]"

The second layer is a spatial dropout layer, which discards a fraction of the input words for regularization purposes. It is followed by a layer of bidirectional GRUs. The output of the GRU layer traverses a dropout layer and a dense layer. A dense layer with a softmax activation and two outputs handles the final classification. The network is trained with the Adam optimizer and binary cross-entropy as the loss function. Early stopping on the decrease of validation loss determines the number of training epochs. We set the number of neurons for the GRUs to 32, the dropout to 0.1, and the number of neurons of the dense layer to 16 based on our previously presented experiments, and refrain from extensive hyperparameter optimization.

5.3.5 Evaluation

The first experiment evaluates the classification accuracy on a dataset of comments from The Guardian. We compare four classifiers: (1) logistic regression on text length (baseline), (2) logistic regression on text and user features [127], (3) a CNN [92], and (4) our GRU-based neural network. Second, we use explanation methods for neural networks to investigate which words have the strongest influence on our model's predictions. For validation purposes, we finally evaluate classification accuracy on a different dataset, which consists of product reviews from *Amazon.com.*

Reader Comments. We consider the task of classifying reader comments into the classes *top* and *flop 10 percent* with regard to the bias-corrected number of upvotes or replies received. For example, a comment classified as *top 10 percent* received a larger relative number of upvotes than 90 percent of the comments with the same rank. We use classification accuracy as the evaluation metric because of the balanced class distribution.

For comparison, we implemented two approaches from related work: a CNN for sentence classification by Kim [92] and a feature-based classification approach by Park et al. [127]. Kim's CNN uses a single layer of convolutions and max-pooling. Due to the relatively small number of parameters in this layer, the emphasis is put on the word embedding layer. The feature-based classification approach by Park et al. [127] was specifically developed to support moderators in identifying high-quality online news comments. It uses the following features: comment length, comment readability, average comment length per user, average comment readability per user, and average number of received upvotes per user. These features serve as the input for a logistic regression classifier. In addition to the two approaches from related work, we consider a naive baseline: a logistic regression classifier with the only feature being the comment length.

Table 5.10 shows the accuracy on the task of classifying top and flop comments with regard to upvotes and replies. The column with the name *10* refers to training on a dataset with the two classes *top 10 percent* and *flop 10 percent,* which contains 20,000 comments. There are two more variants of the experiment, also listed in Table 5.10. The column with the name *25* refers to training on a dataset with the two classes *top 25 percent* and *flop 25 percent,* which contains 53,000 comments. The column with the name *50* refers to training on a dataset with the two classes *top 50 percent* and *flop 50 percent,* which contains 106,000 comments. While the training data differs, we use a shared test dataset split from the top/flop 10 percent because the labels in this dataset are the most reliable. The other datasets contain more samples but are noisier. The remaining data

Table 5.10: Classification accuracy on the task of distinguishing top and flop comments from THE GUARDIAN based on the number of received upvotes and replies.

	Upvotes			Replies		
Top/Flop Percent	10	25	50	10	25	50
Baseline	.61	.61	.61	.63	.63	.63
Park et al. [127]	.65	.66	.67	.61	.59	.60
Kim [92]	.67	.63	.62	.69	.65	.67
Our Approach	**.71**	**.71**	**.71**	**.70**	**.72**	**.68**

for each variant is split into 80 percent training and 20 percent validation set. We make sure that there is no overlap between the shared test set and any training or validation dataset. Each experiment is repeated ten times.

We perform a paired one-tailed t-test with a 95 percent confidence level to test the significance of our findings. Our null hypothesis is that the true mean difference of the classification accuracy of the GRU and CNN approach is less than or equal to zero. The null hypothesis is rejected for all our experiments, leaving us with strong evidence that the GRU approach outperforms the CNN approach with regard to classification accuracy. The results in Table 5.10 further show that the limitation to the top/flop 10 percent for training, in general, does not improve classification accuracy. More reliable labels but also a smaller number of training samples are a consequence of this limitation. The GRU approach achieves the best performance on both tasks, upvote and reply prediction. To our surprise, this approach, the logistic regression baseline on comment length only, and the feature-based approach by Park et al. [127] are robust and insensitive to the different variants of training data (top/flop 10, 25, 50 percent). However, the CNN approach is less robust and performs better if trained on the top/flop 10 percent dataset. If trained on the other dataset variants, the model overfits and does not generalize well to the test data.

Explaining Predictions. We revisit the explanation methods for neural network models presented in Section 4.4 and use them to better understand what makes some comments more engaging than others. To this end, we sort all words in the vocabulary according to their relevance for our model predicting many upvotes or replies. These word relevance scores are calculated with four methods: layer-wise relevance propagation (LRP) [14], gradient-based sensitivity analysis (SA) [107], integrated gradients [187], and a random baseline.

The goal of the experiment is to measure how the deletion of different words changes the classification accuracy of our GRU model. If we consider only true positives, the accuracy in this set is initially 1. The accuracy decreases if we delete the words that are most relevant for the model's prediction and re-run the classification afterward. If we consider only false negatives, the accuracy in this set is initially 0. The accuracy increases if we delete the words that are least relevant for the correct class and re-run the classification. The words that are deleted speak against the correct class. Therefore, if their deletion changes the classification in favor of the correct class, the accuracy increases. Figure 5.6 visualizes how deleting the most/least relevant words affects the

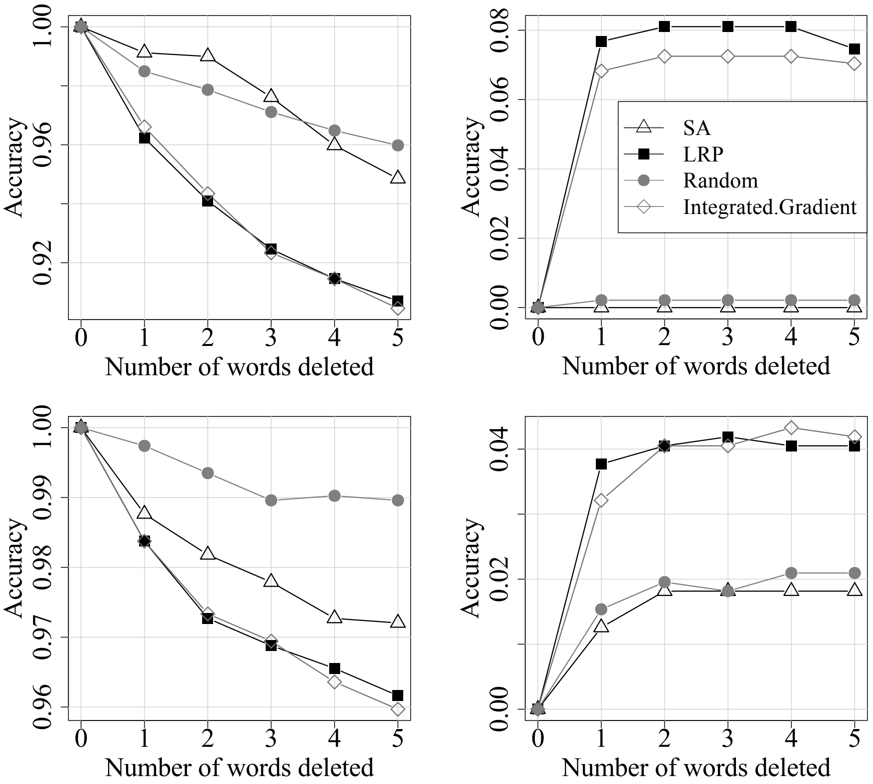


Figure 5.6: Deleting the most relevant words from true positives (left) and the least relevant words from false negatives (right) has the strongest effect on reply prediction (top) and upvote prediction (bottom) when using word relevance scores by LRP.

classification accuracy of our GRU model. The larger the change in accuracy, the better are the calculated word relevance scores. The two methods LRP and integrated gradients provide almost the same relevance scores, and both outperform the gradient-based sensitivity analysis and the random baseline.

Based on layer-wise relevance propagation (LRP) [14], we identify the most and least relevant words for our model's decisions. Words that refer to strong emotions or controversial topics *(arrogant, depressing, fantastic, bearable, Brexit)* are most relevant for predicting upvotes. Least relevant are stop words *(won't, wasn't)* or emotions that are typically expressed in short comments *(lol, sigh)*. Most relevant for predicting many replies are words referring to the Labour party *(socialist, lefty)*, which corresponds to the political orientation of most readers of The Guardian. The least relevant words are names of British public figures *(Pickles, Keir, Tanner, Morgan)*.

Furthermore, we labeled all positive samples in the test set of the *top/flop 10 percent* dataset according to our taxonomy of engaging comments. For each class, Figure 5.7 shows our model's recall at distinguishing top and flop comments. The classes *Correction* and *Comment consent* are omitted because there was only a handful of such samples in the test set. The recall for *Joke/humor* is lowest, whereas the recall for *Comment dissent* or speculation about *Future* and *Reasons* is highest. This discrepancy means that the model's predictions could be improved by better detection of *Joke/humor*. Further, ques-

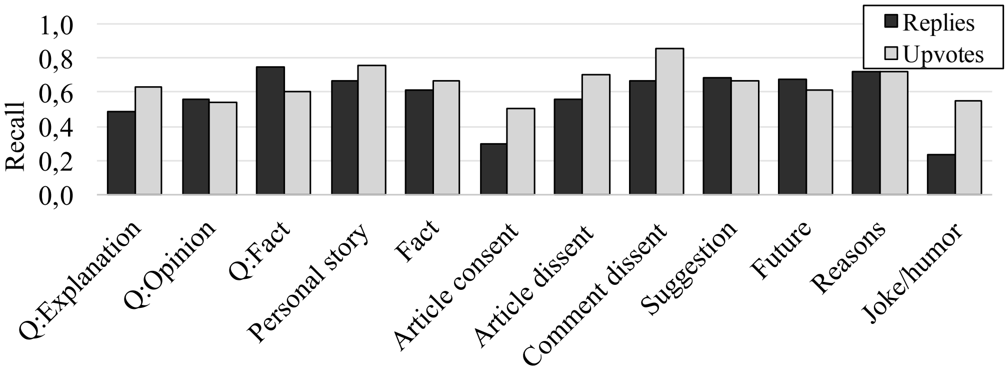


Figure 5.7: Recall for identifying engaging comments differs per class, e.g., jokes are less often identified than dissent.

tions asking for facts (*Q:Fact*) are identified with higher recall than comments providing facts (*Fact*). Besides these differences, the recall for all classes is similar.

Product Reviews. Reader comments on news platforms and product reviews on online retail platforms have several properties in common: (1) popular news articles, as well as popular products, generate an overwhelming number of posts, (2) posts on both platforms are typically short, and (3) both allow users to vote on posts. On product review platforms, upvotes resemble votes on the helpfulness of a review. However, news discussions differ from disconnected posts on Amazon or Twitter, where no discussions take place and the communication is mostly unidirectional. Reviews focus on a particular product and do not refer to each other. Danescu-Niculescu-Mizil et al. [49] analyzed a dataset of Amazon product reviews and their helpfulness votes. They concluded that users find a review more helpful if the associated product rating is closer to the average rating for this product (conformity bias).

We consider product reviews posted on *Amazon.com* to study the applicability of our approach to other domains. The dataset contains 82 million Amazon product reviews, spanning from May 1996 to July 2014, and is available online [79]. 170 million upvotes ("Was this review helpful?") were cast in total. We filter the dataset so that only the ten earliest reviews per product remain. Products with fewer than ten reviews are discarded. Similar to the dataset of reader comments from The Guardian, we learn word embeddings on this large dataset. The reviews comprise 7.6 billion tokens, which is approximately twice the number of tokens in the English Wikipedia.

For our classification experiment, we use a subset of 9 million book reviews to reduce topical variety. We apply the same normalization steps to the number of upvotes as described earlier and construct three different variants of the dataset. They correspond to the top and flop 10, 25, and 50 percent of the product reviews and contain 220,000, 550,000, and 1.1 million product reviews, respectively. The test set is shared for all variations of the training data and comprises 10 percent of the *top/flop 10 percent* dataset (22,000 reviews). The remaining data for each variant are randomly split into 80 percent training and 20 percent validation set. We compare the classification accuracy of the logistic regression baseline on review length, the CNN by Kim [92], and our approach on the task of distinguishing helpful (top) and non-helpful (flop) product reviews. The

Table 5.11: Classification accuracy on the task of distinguishing top and flop product reviews on *Amazon.com* with regard to the number of received helpfulness upvotes.

Top/Flop Percent	10	25	50
Baseline	.67	.67	.34
Kim [92]	.67	.72	.64
Our Approach	**.76**	**.75**	**.66**

feature-based classifier by Park et al. [127] cannot be applied to the product reviews because it requires user information, which the dataset of product reviews does not contain.

Table 5.11 lists the results of the experiment. The GRU model outperforms the CNN. In contrast to our comment dataset, the limitation to the top/flop 10 percent on the product reviews dataset improves classification accuracy. Here, the training dataset is ten times larger, diminishing the disadvantage of limiting the data to the top and flop 10 percent. The more reliable labels in the top/flop 10 and 20 percent training datasets make the difference.

Training on the top/flop 50 percent dataset results in the worst performance. The baseline that considers only the comment length is even worse than random guessing, which achieves 50 percent accuracy. The different value distributions of review lengths in training and test data explain this result. The baseline is unable to learn an appropriate threshold for the comment length. The most and least engaging product reviews in the top/flop 50 percent dataset have a similar average length (1076 vs. 1055 characters). In contrast, there is a clear separation for review lengths in the top/flop 10 percent dataset (677 vs. 1387 characters).

5.3.6 Discussion

Our approach introduces an alternative to chronological comment ranking. The idea is to sort comments by the expected reader engagement in the form of upvotes and replies. If applied, on the one hand, the visibility of top-performing, engaging comments increases. They are shown to more users. On the other hand, the least engaging, flop comments lose visibility and are practically hidden at the end of the ranking list, which usually no user accesses. Furthermore, our engaging comment classification could reduce the manual effort of selecting editor's picks. Instead of reading through all comments, editors could narrow their search down to the most engaging comments.

A limitation of our study is that we only consider a comment's text content and no user-based features. The reputation of the comment author presumably affects its impact in terms of visibility and thus received upvotes and replies. Further, the most comment texts that we explored are well-formed and grammatically correct, simplifying the analysis. Emoticons and slang are rarely used on the platform of The Guardian. However, they might be more frequent on other platforms and pose a potential challenge. Platforms' design changes are an additional challenge for analyzing a long time span. For example, with the most recent platform features, readers can sort comments by time or by the number of upvotes. The default setting of the sorting, e.g., newest/oldest first, is an important factor for the visibility of individual comments. Editor's picks change the visibility of selected comments in a similar way.

5.4 Journalists' Engagement in Reader Discussions

Although primarily meant as forums where readers discuss amongst each other, comment sections can also spark a dialog with the journalists who authored the article. A small but important fraction of comments address the journalists directly, e.g., with questions, recommendations for future topics, thanks and appreciation, or article corrections. However, the sheer number of comments makes it infeasible for journalists to follow discussions around their articles in extenso. A better understanding of this data could support journalists in gaining insights into their audience and fostering engaging and respectful discussions. To this end, this section describes how we constructed a dataset of dialogs in which journalists from The Guardian replied to reader comments and identify the reasons why. Based on this data, we formulate the novel task of recommending reader comments to journalists that are worth reading or replying to, i.e., ranking comments in such a way that the top comments are most likely to require the journalists' reaction. As a baseline, we trained a neural network model with the help of a pairwise comment ranking task.

5.4.1 From Letters to the Editor to Online Comment Sections

Not long ago, the interaction between newspapers and their readership was mostly unidirectional. A reader's letter to the editor was a time-consuming task and, therefore, a rare exception. The editor could decide to publish the letter in the next issue of the newspaper together with a statement or reply.

Nowadays, online news platforms offer comment sections, where any reader can easily post comments and discuss article topics at any time from anywhere. Some comments on these platforms directly address the journalist who wrote the article. Once posted, readers expect the journalists to read their comments and reply back. This is the case, for example, if they stumble over a mistake in an article's text, which could be a simple typo or wrong information. Other comments are questions to the journalists, for example, asking for background information on a topic.

The majority of comments address the broader audience of all readers. Still, journalists can foster respectful discussions by joining as moderators. For highly controversial topics or when a discussion drifts to a disrespectful tone, their intervention could ensure compliance with the platform's rules. However, the sheer number of comments makes it infeasible for journalists to read each and every comment. As a consequence, they find comments that are interesting for them only once in a while and are not aware of all those that would require a reaction. Interesting ideas get lost, discussions get out of focus, and journalists and readers get disappointed. A first requirement to increase journalist engagement is to make them aware of the comments that require a reaction. We define these comments that are worth reading for journalists as relevant comments. They could be worth reading for different reasons: praise or criticism of the article or journalist, which does not necessarily require a reaction, or direct questions to the journalists, which should be answered.

5.4.2 Recommending Reader Comments to Journalists

We introduce the novel task of recommending relevant reader comments to journalists. This task can either be interpreted as a classification or a ranking task. For the former, given a set of reader comments, all comments that require a reply from a journalist need to be identified. For the latter, a given set of reader comments needs to be sorted by the necessity or likeliness of a reply from a journalist. As a baseline, we present a neural network model that is trained on a pairwise-ranking task. In an application scenario, a recommender system could show the most relevant comments to journalists together with an explanation of what makes them relevant. For example, journalists could get to see a personalized view of the website that differs from the public view. Instead of displaying the comments in chronological order, the comments would be ranked based on each comment's relevance to the journalists. Thus, they could find the relevant ones at the very top and could reply to them quickly if necessary. If time permits, more comments could be explored in the order of their estimated relevance.

Closest to the kind of data needed for this task is the dataset provided by Schabus et al. [174]. It consists of almost 12,000 German-language news comments that have been labeled with regard to nine categories. One of their labeled categories is "feedback", which includes questions and suggestions addressing the article authors. A subset of comments in this category might require a reply from the journalist or the editor. Häring et al. [78] study a feature-based approach for identifying comments that address a journalist or the news platform provider in general. They consider, for example, the news section of the article as a feature but also the timestamp. In contrast, we focus on the comment text only and do not aim to predict journalist replies but to recommend which comments need a reply. This difference is significant because we do not want to predict the journalist replies as they were in the past, but we want to increase journalist engagement. Therefore, a good predictor that takes, e.g., the timestamp of a comment into account as a feature, does not help in our scenario. We argue that our task is related to but different from prediction. To the best of our knowledge, no research has been conducted on the task of recommending reader comments to journalists so far. Further, no publicly available dataset can readily be used for such research.

5.4.3 Dataset Construction

In the following, we describe the process of collecting the dataset of journalists' interactions with their readership, introduced in Subsection 3.2.3. We collected 51 million reader comments from The Guardian posted between November 2011 and December 2018. Before that time, there was simply no option to post a comment in the form of a reply to another reader's comment on this platform.[5] We selected all 18,877 reader comments that received a reply from the journalist (positive samples), the 18,877 replies, and 18,877 randomly sampled reader comments that did not receive a reply from the journalist (negative samples). The negative samples were selected from the set of comments that were posted one hour before or after the journalist's reply. Thus the journalist presumably was active at that time and could have replied to the negative sample.

[5] `https://www.theguardian.com/help/insideguardian/2011/nov/03/responses-in-comments`

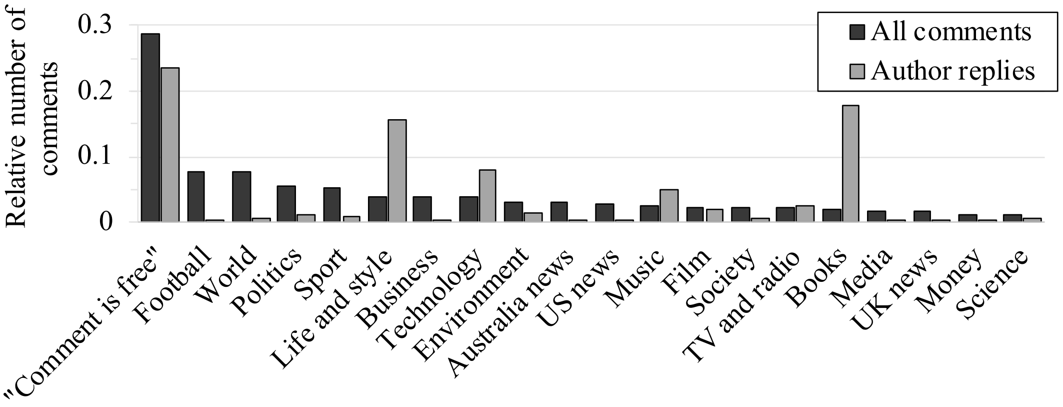


Figure 5.8: For the top 20 news sections of The Guardian, dark bars show the relative number of comments by readers and light bars show the relative number of journalist replies. For example, journalists engage especially in the section on books.

For each considered news article, we sample the same number of positive and negative samples. Thereby, we prevent an article bias and ensure the same number of positive and negative samples per topic. Figure 5.8 shows the relative number of comments in the top 20 news sections. The majority of all reader comments and also of all journalist replies are posted in the *comment is free* section. This section is where The Guardian main commentators and selected contributors from outside publish opinion articles. Interestingly, journalist replies peek for the sections *life and style, technology, music,* and *books.* Journalists seem to be more active in these sections: For example, while articles in the section on books receive only two percent of all comments, they receive 18 percent of all journalist replies.

Enriching the Data with Machine Labeling. We enrich the dataset with additional labels generated by a pre-trained machine learning model. This model classifies comments with regard to various labels, such as *informative* or *controversial.* As a result, it allows investigating whether informative reader comments are more likely to receive a journalist's reply than controversial ones. We trained this model using a dataset of 9,000 comments by Napoles et al. [122], YNACC. Following their approach, we train a logistic regression classifier on their data to automatically label the comments in our dataset afterward.

Table 5.12 lists the mean label scores for the set of comments that received a reply from the journalist (positive samples) and the set of comments that did not receive a reply from the journalist (negative samples). The scores differ only slightly for the majority of labels. The label with the most significant difference is persuasiveness: the average score is 38.3 percent for the positive samples, while it is 35.8 percent for the negative samples. However, the variance of this label's scores is also the highest among all labels.

Analyzing the comments and the machine-labeled scores by hand, we find that journalists not only reply to comments where it is obvious that a reply is required but also to random-looking comments. For example, a comment that only consists of the emoticon ":^)" is machine-labeled as 2.4 percent persuasive and 4.5 percent informative, but still

Table 5.12: Mean and variance of machine-labeled scores.

	Positive Samples		Negative Samples	
Label	Mean	Var	Mean	Var
Audience	78.7	0.5	79.5	0.5
Agreement	19.4	0.1	20.0	0.1
Informative	37.0	1.6	35.1	1.5
Mean	35.4	0.2	35.7	0.2
Controversial	61.1	0.7	60.0	0.8
Disagreement	60.5	0.3	60.5	0.3
Persuasive	38.3	2.4	35.8	2.4
Off-Topic	57.7	0.6	58.7	0.6
Neutral	46.0	0.1	46.1	0.1
Positive	12.9	0.1	13.2	0.1
Negative	69.3	0.1	69.3	0.1
Mixed	32.9	0.9	31.5	0.9

received a reply from the journalist. Similarly, the comment “Sounds good to me...!” is machine-labeled 12.3 percent persuasive and 15.4 percent informative, but still received a reply from the journalist. An inherent limitation of the machine labeling approach is that it was trained on a different dataset, and the classification is, therefore, more error-prone. For example, even if a comment is not informative, the model sometimes assigns a high probability to this label.

Manual Labeling Procedure. In addition to the machine labeling approach, we manually labeled a subset of the data with regard to the reasons why journalists replied to reader comments. The labels are based on our hierarchical taxonomy of engaging comments presented in Subsection 5.3.3, where it was used for labeling the reasons why some comments receive an above-average number of replies and upvotes.

Three annotators labeled 1,000 reader comments that received a reply from a journalist according to this taxonomy in parallel. During the annotation process, the annotators had access to the reader comment, the journalist reply, and the news article’s title. On the coarse-grained level (with only the four classes *Question, Information, Joke/humor,* and *Opinion*), at least two out of three annotators agreed on the labels for 90 percent of the comments. The inter-annotator agreement in terms of Fleiss’ Kappa is 0.42. On the fine-grained level, at least two out of three annotators agreed on the labels for 70 percent of the comments. The inter-annotator agreement in terms of Fleiss’ kappa is 0.37.

One collective label was derived from the individual labels by the annotators. To this end, in cases where two annotators agreed, their label decision overruled the third annotator. In cases where all annotators disagreed, we excluded the sample from the published dataset. Figure 5.9 shows the resulting label distribution. The majority of the comments that received a journalist reply were labeled as dissent with the article topic or negative sentiment towards the article topic. The labels serve as a starting point for investigating *why* journalists reply to particular reader comments.

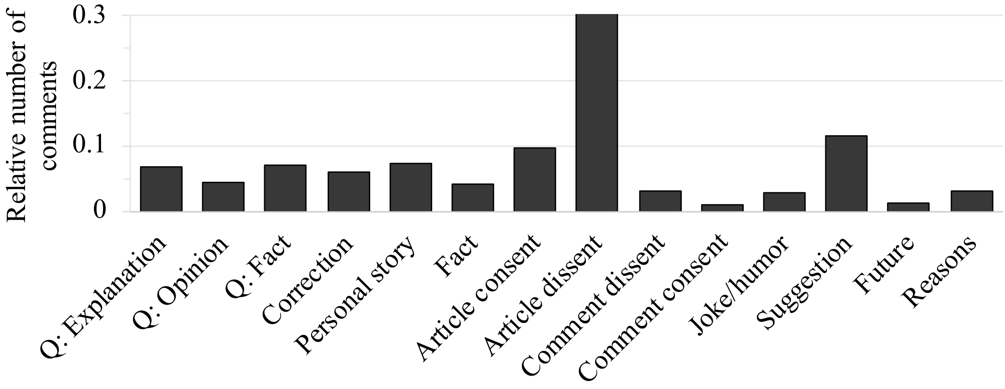


Figure 5.9: Distribution of the manually created class labels for the reader comments. The most frequent comments that receive a reply from the journalist are comments with negative sentiment towards the article topic. Corrections are at seventh rank. For a detailed description of the classes, see Subsection 5.3.3

Accessing the Dataset. The dataset can be downloaded via the official web API of The Guardian. To this end, we provide predefined lists of 38,000 reader comments and 19,000 journalist replies identified by their comment IDs, and a small script that accesses the API.[6] This script also joins the retrieved comment texts and metadata with our labels. An API key is required for using the API, which can be applied for via an online form by providing name and email address and accepting the terms and conditions of the Guardian Open Platform. On the one hand, the procedure allows other researchers to reproduce the dataset and the experiments and continue research in this field. On the other hand, platform users still have the option to delete (or edit) their own comments so that they are not shown on the website and cannot be retrieved via the API anymore. The procedure follows our considerations of dataset reproducibility, as described in Section 3.3.

5.4.4 Baseline Approach and Experiments

The labels provided with the dataset allow investigating many aspects of the interactions between journalists and their readership. Exemplary aspects are the dynamics of the sentiment of their comments, journalists' explanations and apologies for mistakes in their articles, or correlations of a reply's text length and the number of upvotes it receives. We focus on the task of identifying comments that require a reply from the journalist and ranking the comments accordingly. There is no clear borderline between comments that do or do not require a reply. While the journalist, in the end, needs to make a binary decision (to reply or not to reply), the binary training data with positive and negative samples is only a small excerpt of reality. There is no single correct solution, and different journalists react for different reasons. Therefore, we consider a ranking task of comments instead of a binary classification task. We rank comments by the likelihood of receiving a reply so that journalists can get the most relevant comments displayed at the top of the discussion sections. We suggest generating a ranking of all comments step-by-step

[6] www.github.com/julian-risch/CIKM2020

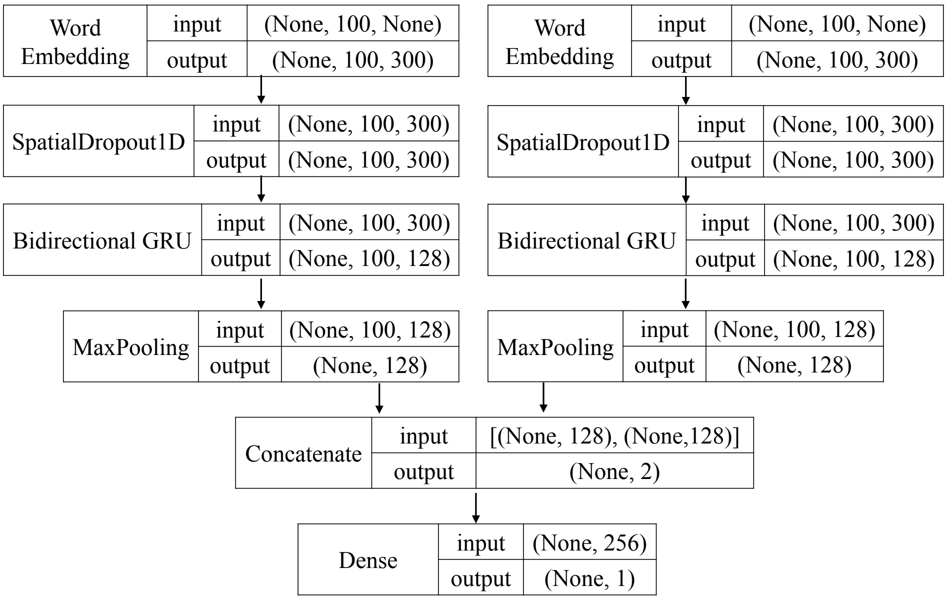


Figure 5.10: A Siamese neural network architecture allows a pairwise ranking of comments.

with a pairwise ranking approach. Given two comments, the task is to decide which one is more relevant to journalists, i.e., which comment is more likely to receive a reply from them. The pairwise decisions can be aggregated with a variety of methods to obtain a global ranking, which are beyond the scope of this thesis.

To solve the described task, we train a deep neural network on pairs of positive and negative samples. The input of the neural network consists of two comments, where one received a reply from the journalist, and the other did not. Figure 5.10 visualizes the network's architecture. The network exhibits a Siamese structure, where the two inputs traverse an encoder of two word embedding layers and two bidirectional GRU layers that run in parallel and share weights. The two bidirectional GRU layers each encode the sequence of word embeddings of a comment, and each of them is followed by a max-pooling layer. The output of the two max-pooling layers is concatenated. The final output of the network is calculated by a dense layer with a single sigmoid activation function and determines which of the two input comments is more likely to receive a reply. For training the network, we use binary cross-entropy as the loss function and an Adam optimizer. Not considering the word embedding layer, the neural network has a rather simple structure with a small number of trainable parameters. This limited capacity is tailored to the comparably small size of the training dataset.

For the evaluation of the model, the dataset is split into 80 percent training, 10 percent validation, and 10 percent test data. To tune the number of training epochs, we use early stopping and monitor the loss on the validation set. Because of the balanced class distribution, the evaluation uses classification accuracy. The model achieves an accuracy of 64.0 percent, which means that the two input comments are ranked in the correct order in about two-thirds of the cases — leaving room for improvement.

5.4.5 Discussion

The presented dataset and neural network model provide an insight into how the media and publishing industry can benefit from the ranking and classification of reader comments. However, the studied machine learning approaches demand labeled training data, which is costly to obtain. Identifying and gathering this critical component appears to be a major challenge.

Recommending to journalists when to reply to a reader comment is a hard task, not least because it finally comes down to a journalist's personal decision. Some journalists are more active on the platform than others. There are several reasons for this variety. First, journalists work in different news sections, such as politics, sports, or books, and therefore, the discussion topics differ. Some topics are suited for journalists to post their personal opinion, for example, when it comes to book recommendations. Other topics demand strict neutrality, such as football matches, where readers support opposing teams. Further, journalists might be unavailable at the time of publication of their article and, therefore, cannot reply to reader comments in time. Because of the short attention span and fast-paced media business, a journalist reply that is posted a few hours later would go mostly unnoticed by users. The presented baseline approach neglects that journalists have different notions of what makes comments worth reading and worth replying to.

Outliers and Noise. Given the subjective nature of the task of replying to a reader comment, there are some outliers in our dataset that would be hard (if not impossible) to predict. Sometimes even very short comments gained the journalists' attention and resulted in a reply. For example, the short comment "Yay!" received a reply from a journalist:

reader A: Gunny would plug for Tom Tomorrow, and i'd plug for Jen Sorensen too.

[...]

reader A: Yay!

> **journalist:** Evidently, I should get acquainted with Jen Sorensen's work also [...]

The reason for this lies in the other comments and their context, specifically in a comment posted by the same user earlier. Actually, the journalist replied to both comments by the reader, but the dataset indicates it only as a reply to the shorter comment.[7] The comments are part of a discussion about layout and feature changes of the comment section, which explains why the journalist more actively joins the discussion: The discussion topic is THE GUARDIAN itself. The machine-labeled probabilities of persuasiveness and informativeness are low and fail to identify the reader comment as relevant to the journalist. Another example for a hard to predict reply stems from an article entitled "How to eat: beef stew".[8]

reader B: No mention of Yorkshire Pudding as an accompaniment? An absolute must in our house.

[7] www.gu.com/help/insideguardian/2012/apr/23/makeover-comment-is-free-america

[8] www.gu.com/lifeandstyle/wordofmouth/2014/feb/27/beef-stew-bread-dumplings-bowl-no-potatoes

reader C: Instead of dumplings or instead of bread?

journalist: There is, of course, nothing that Yorkshiremen wouldn't eat out of a pudding. Trifle, cereal, you name it.

The reply from reader C is machine-labeled with low persuasiveness (0.15) and informativeness scores (0.18), but, surprisingly, the journalist replied to it. This reply is unpredictable for our model, and it is a debatable point whether this sample helps the training process or rather is noise that should be removed in a pre-processing step.

5.5 Summary

In this chapter, we studied engaging comments and discussions as an antipole to toxic comments. To this end, we designed, implemented, trained, and tested machine learning models to predict reader engagement based on community preferences and individual preferences. Refraining from manually labeling large datasets, we leveraged the number of upvotes and replies that a comment receives as a measure of engagement. After correcting bias in these numbers, we analyzed the characteristics of the most and least engaging comments and revisited explanation methods for neural networks to identify words that strongly affect the classification results. Finally, we switched from reader engagement to journalist engagement and predicted which reader comments require the journalists' attention. We presented a dataset of interactions between journalists and their readership and demonstrated that it can readily be used for supervised machine learning.

6

Conclusion

In this thesis, we analyzed reader comments on online news platforms and developed machine learning models to foster respectful and engaging reader discussions. This final chapter concludes the thesis with a summary of the presented work and an outlook on directions for future research.

6.1 Summary

Online news platforms that provide comment sections are faced with an enormous challenge. Due to the misuse by spammers, haters, and trolls, the reader discussions require time-consuming moderation. The ever-increasing costs of this process have led many platforms to consider the discontinuation of their comment sections. With the goal of supporting them in keeping the comment sections open, we formulated the main research question: "How can we foster respectful and engaging online discussions?" and explored the potential of machine learning applications to address this question. To begin with, we defined the concepts of toxic comments, which make readers leave a discussion, and engaging comments, which make readers join a discussion. The two main chapters of this thesis, Chapter 4 and Chapter 5, focused on the classification of toxic comments and the recommendation of engaging comments and discussions.

While related work has already studied feature-based and deep-learning-based approaches to toxic comment classification, reader engagement in online discussions is a widely under-researched topic. Chapter 2 summarized related work and gave an overview of neural network architectures and published datasets. We identified the small size of labeled comment datasets as an obstacle to the training of complex neural network models. This insight motivated us to develop neural network architectures and training procedures that require fewer training samples and to collect and explore larger datasets. We described the process of collecting comments from Zeit Online, The Guardian, and Daily Mail and their integration into a unified data model in Chapter 3. By designing and implementing a process to measure and facilitate dataset reproducibility, we addressed the challenges of sharing comment datasets for research purposes. Our experiments demonstrated the practical feasibility of this process and confirmed that reproducible research does not require the direct sharing of data.

After evaluating current neural network models for toxic comment classification at the beginning of Chapter 4, we addressed the problem of limited training data by presenting methods for data augmentation and ensemble learning. Our ensemble of transformer-based neural network models outperformed various other approaches in shared task competitions, and we investigated the effect of random weight initialization on its performance. Working towards the integration of machine-learned models into the manual moderation process, we compared four methods to explain automatic classification results and collaborated with an online news platform to evaluate a feature-based classifier in a real-world application scenario.

Chapter 5 concentrated on highlighting and encouraging engaging comments and discussions. Based on an extensive set of features, we predicted which article discussions are engaging in the sense that they attract many commenters. Turning from community preferences to individual preferences, we then addressed the task of personalized discussion recommendation. To this end, we constructed a combination of content-based and user-co-occurrence-based neural network models and trained it on carefully selected training samples. As a result, we obtained a model that can estimate the probability that a particular user posts a comment in a particular discussion and enables platforms to rank and recommend the discussions accordingly. Further, we considered reader engagement on the level of individual comments and classified the most and the least engaging comments based on the number of received upvotes and replies. Extending the user group to journalists, we collected and analyzed a dataset of their interactions with readers through comments and demonstrated its usage for supervised machine learning. The publication of this dataset aims to stimulate and facilitate future research on this novel aspect of online discussions.

6.2 Outlook

The research results presented in this thesis point to different directions for future work. One possible direction is to further improve the proposed models' classification accuracy, but there are plenty of other interesting challenges. Four of them, which we consider particularly important, are: (1) taking into account a comment's context for its classification, (2) developing explanation methods for complex transformer-based neural network models, (3) analyzing cause-and-effect relationships between a comment's publication time, received upvotes and replies, etc., and (4) designing user interfaces beyond seemingly endless lists of comments spread across dozens of subpages. In the following, we briefly describe each of these research directions.

Comment classification typically only considers the comment text as input and is therefore context-agnostic. However, it would be interesting to develop context-aware classifiers that incorporate the text of surrounding comments, such as replies to that comment or previous comments in the same discussion. Further, the text of the corresponding news article or the user profiles of discussion participants could be taken into account. The goal here is to correct misclassifications due to the lack of understanding of the context, such as irony or sarcasm. A motivation for incorporating additional input is how humans read online comments. Because of the web page layout of social networks and news platforms, top-ranked comments, e.g., the earliest comments in a chronological ranking, receive the most attention. Users typically need to click through dozens of subpages to read lower-ranked comments and, in the process, might come across other comments that influence their perception of the discussion as a whole.

In this thesis, we evaluated attribution-based explanations for recurrent neural networks. One task for future research is to extend the explanation methods so that they become context-aware. In contrast to explicitly offensive language, implicitly offensive language cannot be explained by highlighting single words of the comment, such as swear words. As discussed in Section 4.3, BERT represents a word's context with its context-specific word embeddings. Therefore, a promising first step towards context-aware explanations is to improve explanation methods for BERT and other transformer-based models to build trust in these models and their application to semi-automatic comment moderation. The work by van Aken et al. [195] is a first approach in this research area.

The relations between comments and their number of received upvotes and replies are an interesting topic for research. Causal graphs allow expressing cause-effect hypotheses and testing their implications against the observational data, e.g., conditional independence among variables. A potential-outcome model could allow computing the effects of hypothetical interventions, and it would be interesting to apply this model to comment datasets. For example, the model could answer the question: What would be the effect on the number of received upvotes if a comment's position in the chronological ranking is changed? Counterfactuals could allow going backward in time and drawing explanations about the possible causes for a certain effect of interest. For example, given a comment in our dataset, what would have increased its number of replies? An earlier publication, a longer text, or more received upvotes?

One reason that hinders in-depth discussions on online news platforms is the overwhelming number of comments and their confusing visualization as a long list split across multiple pages. From an applied research perspective, a future task is, therefore, to develop tools for readers to explore the comments. The goal is that potential discussion participants can get a quick overview and are not discouraged by an abundance of comments. The idea of visualizing comments of a single-platform could also be extended to a multi-platform scenario, where the comments from different platforms are shown in one joint visualization.

There is still much work to be done to ensure that readers can have respectful and engaging discussions on online news platforms without any obstacles. This thesis laid the foundation for tackling these tasks by presenting machine learning approaches for classifying toxic comments and recommending engaging comments and discussions. We hope that news platforms, first and foremost, the moderators and the journalists, but also their readers, will benefit from our research results. Our prototype of a semi-automatic comment moderation process has already successfully served as a blueprint and found its way into practice: The news platforms that we collaborated and exchanged ideas with have now integrated machine-learned models into their working routine. Thus, our contributions are not limited to theoretical scientific progress, but also advance practical applications.

References

[1] D. Agarwal, B.-C. Chen, and B. Pang. Personalized recommendation of user comments via factor models. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 571–582. ACL, 2011.

[2] M. Aharon, A. Kagian, R. Lempel, and Y. Koren. Dynamic personalized recommendation of comment-eliciting stories. In *Proceedings of the Conference on Recommender Systems (RecSys)*, pages 209–212. ACM, 2012.

[3] N. Albadi, M. Kurdi, and S. Mishra. Are they our brothers? analysis and detection of religious hate speech in the arabic twittersphere. In *Proceedings of the International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*, pages 69–76. ACM, 2018.

[4] K. K. Aldous, J. An, and B. J. Jansen. View, like, comment, post: Analyzing user engagement by topic at 4 levels across 5 social media platforms for 53 news organizations. In *Proceedings of the International Conference on Web and Social Media (ICWSM)*, pages 47–57. AAAI Press, 2019.

[5] M. Á. Álvarez-Carmona, E. Guzmán-Falcón, M. Montes-y Gómez, H. J. Escalante, L. Villasenor-Pineda, V. Reyes-Meza, and A. Rico-Sulayes. Overview of MEX-A3T at IberEval 2018: Authorship and aggressiveness analysis in Mexican Spanish tweets. In *Proceedings of the Workshop on Evaluation of Human Language Technologies for Iberian Languages (IberEval@SEPLN)*, pages 74–96. CEUR, 2018.

[6] C. Ambroselli. Quality management for online news comments. Master's thesis, University of Potsdam, Hasso Plattner Institute, Prof.-Dr.-Helmert-Str. 2-3, D-14482 Potsdam, 2018.

[7] C. Ambroselli, J. Risch, R. Krestel, and A. Loos. Prediction for the newsroom: Which articles will get the most comments? In *Proceedings of the North American Chapter of the Association for Computational Linguistics (NAACL)*, pages 193–199. ACL, 2018.

[8] A. Anagnostou, I. Mollas, and G. Tsoumakas. Hatebusters: A web application for actively reporting youtube hate speech. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, pages 5796–5798. IJCAI Organization, 2018.

[9] P. Aragón, V. Gómez, D. García, and A. Kaltenbrunner. Generative models of online discussion threads: state of the art and research challenges. *Journal of Internet Services and Applications*, 8(1):15, 2017.

[10] P. Aragón, V. Gómez, and A. Kaltenbrunner. To thread or not to thread: The impact of conversation threading on online discussion. In *Proceedings of the International Conference on Web and Social Media (ICWSM)*, pages 12–21. AAAI Press, 2017.

[11] L. Arras, F. Horn, G. Montavon, K.-R. Müller, and W. Samek. "What is relevant in a text document?": An interpretable machine learning approach. *PloS one*, 12 (8):1–23, 2017.

[12] L. Arras, G. Montavon, K.-R. Müller, and W. Samek. Explaining recurrent neural network predictions in sentiment analysis. In *Proceedings of the Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis (WASSA@EMNLP)*, 2017.

[13] N. X. Bach, N. D. Hai, and T. M. Phuong. Personalized recommendation of stories for commenting in forum-based social media. *Information Sciences*, 352-353:48–60, 2016.

[14] S. Bach, A. Binder, G. Montavon, F. Klauschen, K.-R. Müller, and W. Samek. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PloS one*, 10(7):1–44, 2015.

[15] L. Backstrom, J. Kleinberg, L. Lee, and C. Danescu-Niculescu-Mizil. Characterizing and curating conversation threads: expansion, focus, volume, re-entry. In *Proceedings of the International Conference on Web Search and Data Mining (WSDM)*, pages 13–22. ACM, 2013.

[16] P. Badjatiya, S. Gupta, M. Gupta, and V. Varma. Deep learning for hate speech detection in tweets. In *Companion Proceedings of the International Conference on World Wide Web (WWW Companion)*, pages 759–760. ACM, 2017.

[17] R. Bandari, S. Asur, and B. Huberman. The pulse of news in social media: Forecasting popularity. In *Proceedings of the International Conference on Web and Social Media (ICWSM)*, pages 26–33. AAAI Press, 2012.

[18] T. Bansal, M. Das, and C. Bhattacharyya. Content driven user profiling for comment-worthy recommendations of news and blog articles. In *Proceedings of the Conference on Recommender Systems (RecSys)*, pages 195–202. ACM, 2015.

[19] A. Baruah, K. Das, F. Barbhuiya, and K. Dey. Aggression identification in English, Hindi and Bangla text using BERT, RoBERTa and SVM. In *Proceedings of the Workshop on Trolling, Aggression and Cyberbullying (TRAC@LREC)*, pages 76–82. ELRA, 2020.

[20] V. Basile, C. Bosco, E. Fersini, D. Nozza, V. Patti, F. M. R. Pardo, P. Rosso, and M. Sanguinetti. Semeval-2019 task 5: Multilingual detection of hate speech against immigrants and women in twitter. In *Proceedings of the International Workshop on Semantic Evaluation (SemEval@NAACL)*, pages 54–63. ACL, 2019.

[21] G. Berry and S. J. Taylor. Discussion quality diffuses in the digital public square. In *Proceedings of the International Conference on World Wide Web (WWW)*, pages 1371–1380. ACM, 2017.

[22] S. Bhattacharya, S. Singh, R. Kumar, A. Bansal, A. Bhagat, Y. Dawer, B. Lahiri, and A. K. Ojha. Developing a multilingual annotated corpus of misogyny and aggression. In *Proceedings of the Workshop on Trolling, Aggression and Cyberbullying (TRAC@LREC)*, pages 158–168. ELRA, 2020.

[23] R. Blanco, H. Halpin, D. M. Herzig, P. Mika, J. Pound, H. S. Thompson, and T. Tran. Repeatable and reliable semantic search evaluation. *Web Semantics: Science, Services and Agents on the World Wide Web*, 21:14–29, 2013.

[24] P. Blandfort, T. Karayil, F. Raue, J. Hees, and A. Dengel. Fusion strategies for learning user embeddings with neural networks. In *Proceedings of the International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2019.

[25] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, 2003.

[26] H. Blockeel and J. Vanschoren. Experiment databases: Towards an improved experimental methodology in machine learning. In *European Conference on Principles of Data Mining and Knowledge Discovery (ECML PKDD)*, pages 6–17. Springer, 2007.

[27] T. Bogers, M. Gäde, M. Michael, L. Freund, M. Koolen, V. Petras, and M. Skov. Report on the workshop on barriers to interactive ir resources re-use (biirrr 2018). *SIGIR Forum*, 52(1):119–128, 2018.

[28] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics (TACL)*, 5(1):135–146, 2017.

[29] C. Bosco, D. Felice, F. Poletto, M. Sanguinetti, and T. Maurizio. Overview of the EVALITA 2018 hate speech detection task. In *Proceedings of the Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (EVALITA)*, volume 2263, pages 1–9. CEUR, 2018.

[30] C. Budak, R. K. Garrett, P. Resnick, and J. Kamin. Threading is sticky: How threaded conversations promote comment system user retention. *Human-Computer Interaction (HCI)*, 1(27):1–20, 2017.

[31] P. Burnap and M. L. Williams. Cyber hate speech on twitter : An application of machine classification and statistical modeling for policy and decision making. *Policy & Internet*, 7:223–242, 2015.

[32] P. Burnap and M. L. Williams. Us and them: identifying cyber hate on twitter across multiple protected characteristics. *EPJ Data Science*, 5:1–15, 2016.

[33] S. Carton, Q. Mei, and P. Resnick. Extractive adversarial networks: High-recall explanations for identifying personal attacks in social media posts. In *Proceedings of*

the Conference on Empirical Methods in Natural Language Processing (EMNLP), pages 3497–3507. ACL, 2018.

[34] T. Caselli, V. Basile, J. Mitrović, I. Kartoziya, and M. Granitzer. I feel offended, don't be abusive! implicit/explicit messages in offensive and abusive language. In *Proceedings of the Language Resources and Evaluation Conference (LREC)*, pages 6193–6202. ELRA, 2020.

[35] R. Catherine and W. Cohen. TransNets: Learning to transform for recommendation. In *Proceedings of the Conference on Recommender Systems (RecSys)*, pages 288–296. ACM, 2017.

[36] T. Chakrabarty, K. Gupta, and S. Muresan. Pay "attention" to your context when classifying abusive language. In *Proceedings of the Workshop on Abusive Language Online (ALW@ACL)*, pages 70–79. ACL, 2019.

[37] M. S. Charikar. Similarity estimation techniques from rounding algorithms. In *Proceedings of the Symposium on Theory of Computing*, pages 380–388. ACM, 2002.

[38] D. Chatzakou, N. Kourtellis, J. Blackburn, E. De Cristofaro, G. Stringhini, and A. Vakali. Mean birds: Detecting aggression and bullying on twitter. In *Proceedings of the International Web Science Conference (WebSci)*, page 13–22. ACM, 2017.

[39] Y. Chen, Y. Zhou, S. Zhu, and H. Xu. Detecting offensive language in social media to protect adolescent online safety. In *Proceedings of the International Conference on Social Computing and the International Conference on Privacy, Security, Risk and Trust (SOCIALCOM-PASSAT)*, pages 71–80. IEEE, 2012.

[40] F. Chirigati, R. Rampin, D. Shasha, and J. Freire. Reprozip: Computational reproducibility with ease. In *Proceedings of the International Conference on Management of Data (SIGMOD)*, pages 2085–2088. ACM, 2016.

[41] K. Cho, B. van Merrienboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio. Learning phrase representations using rnn encoder–decoder for statistical machine translation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734. ACL, 2014.

[42] Y.-L. Chung, E. Kuzmenko, S. S. Tekiroglu, and M. Guerini. CONAN - COunter NArratives through nichesourcing: a multilingual dataset of responses to fight online hate speech. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 2819–2829. ACL, 2019.

[43] K. B. Cohen, J. Xia, P. Zweigenbaum, T. J. Callahan, O. Hargraves, F. Goss, N. Ide, A. Névéol, C. Grouin, and L. E. Hunter. Three dimensions of reproducibility in natural language processing. In *Proceedings of the Language Resources and Evaluation Conference (LREC)*, pages 156–165. ELRA, 2018.

[44] D. Colla, T. Caselli, V. Basile, J. Mitrovic, and M. Granitzer. Grupato at semeval-2020 task 12: Retraining mBERT on social media and fine-tuned offensive language models. In *Proceedings of the International Workshop on Semantic Evaluation (SemEval@COLING)*. ACL, 2020.

[45] C. Collberg and T. A. Proebsting. Repeatability in computer systems research. *Communications of the ACM*, 59(3):62–69, 2016.

[46] Ç. Çöltekin. A corpus of Turkish offensive language on social media. In *Proceedings of the Language Resources and Evaluation Conference (LREC)*, pages 6174–6184. ELRA, 2020.

[47] G. V. Cormack, C. L. Clarke, and S. Buettcher. Reciprocal rank fusion outperforms condorcet and individual rank learning methods. In *Proceedings of the International Conference on Research and Development in Information Retrieval (SIGIR)*, pages 758–759. ACM, 2009.

[48] M. Dadvar, D. Trieschnigg, R. Ordelman, and F. de Jong. Improving cyberbullying detection with user context. In *Proceedings of the European Conference on Information Retrieval (ECIR)*, pages 693–696. Springer, 2013.

[49] C. Danescu-Niculescu-Mizil, G. Kossinets, J. Kleinberg, and L. Lee. How opinions are received by online communities: A case study on amazon.com helpfulness votes. In *Proceedings of the International Conference on World Wide Web (WWW)*, pages 141–150. ACM, 2009.

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

[51] O. de Gibert, N. Perez, A. García-Pablos, and M. Cuadros. Hate Speech Dataset from a White Supremacy Forum. In *Proceedings of the Workshop on Abusive Language Online (ALW@EMNLP)*, pages 11–20. ACL, 2018.

[52] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the North American Chapter of the Association for Computational Linguistics (NAACL)*, pages 4171–4186. ACL, 2019.

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

[54] N. Diakopoulos and M. Naaman. Towards quality discourse in online news comments. In *Proceedings of the Conference on Computer Supported Cooperative Work (CSCW)*, pages 133–142. ACM, 2011.

[55] K. Dinakar, B. Jones, C. Havasi, H. Lieberman, and R. W. Picard. Common sense reasoning for detection, prevention, and mitigation of cyberbullying. *Transactions on Interactive Intelligent Systems (TIIS)*, 2(3):18:1–30, 2012.

[56] N. Djuric, J. Zhou, R. Morris, M. Grbovic, V. Radosavljevic, and N. Bhamidipati. Hate speech detection with comment embeddings. In *Companion Proceedings of the International Conference on World Wide Web (WWW Companion)*, pages 29–30. ACM, 2015.

[57] F. K. Došilović, M. Brčić, and N. Hlupić. Explainable artificial intelligence: A survey. In *International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, pages 210–215. IEEE, 2018.

[58] C. Drummond. Finding a balance between anarchy and orthodoxy. In *Proceedings of the Workshop on Evaluation Methods for Machine Learning (ML Evaluation@ICML)*, pages 1–4. AAAI Press, 2008.

[59] M. ElSherief, V. Kulkarni, D. Nguyen, W. Y. Wang, and E. Belding. Hate lingo: A target-based linguistic analysis of hate speech in social media. In *Proceedings of the International Conference on Web and Social Media (ICWSM)*, pages 42–51. AAAI Press, 2018.

[60] E. V. Epure, B. Kille, J. E. Ingvaldsen, R. Deneckere, C. Salinesi, and S. Albayrak. Recommending personalized news in short user sessions. In *Proceedings of the Conference on Recommender Systems (RecSys)*, pages 121–129. ACM, 2017.

[61] E. Fersini, P. Rosso, and M. Anzovino. Overview of the task on automatic misogyny identification at IberEval 2018. In *Proceedings of the Workshop on Evaluation of Human Language Technologies for Iberian Languages (IberEval@SEPLN)*, pages 214–228. CEUR, 2018.

[62] P. Fortuna, J. Rocha da Silva, J. Soler-Company, L. Wanner, and S. Nunes. A hierarchically-labeled Portuguese hate speech dataset. In *Proceedings of the Workshop on Abusive Language Online (ALW@ACL)*, pages 94–104. ACL, 2019.

[63] A.-M. Founta, C. Djouvas, D. Chatzakou, I. Leontiadis, J. Blackburn, G. Stringhini, A. Vakali, M. Sirivianos, and N. Kourtellis. Large Scale Crowdsourcing and Characterization of Twitter Abusive Behavior. In *Proceedings of the International Conference on Web and Social Media (ICWSM)*, pages 491–500. AAAI Press, 2018.

[64] B. Gambäck and U. K. Sikdar. Using convolutional neural networks to classify hate-speech. In *Proceedings of the Workshop on Abusive Language Online (ALW@ACL)*, pages 85–90. ACL, 2017.

[65] J. Gao, X. Xin, J. Liu, R. Wang, J. Lu, B. Li, X. Fan, and P. Guo. Fine-grained deep knowledge-aware network for news recommendation with self-attention. In *Proceedings of the International Conference on Web Intelligence (WI)*, pages 81–88. IEEE, 2018.

[66] L. Gao and R. Huang. Detecting online hate speech using context aware models. In *Proceedings of the International Conference Recent Advances in Natural Language Processing (RANLP)*, pages 260–266. INCOMA Ltd., 2017.

[67] S. V. Georgakopoulos, S. K. Tasoulis, A. G. Vrahatis, and V. P. Plagianakos. Convolutional neural networks for toxic comment classification. In *Proceedings of the Hellenic Conference on Artificial Intelligence (SETN)*, pages 1–6. ACM, 2018.

[68] F. A. Gers, J. Schmidhuber, and F. Cummins. Learning to forget: Continual prediction with LSTM. *Neural Computation*, 12:2451–2471, 1999.

[69] C. H. E. Gilbert. VADER: A parsimonious rule-based model for sentiment analysis of social media text. In *Proceedings of the International Conference on Web and Social Media (ICWSM)*, pages 216–225. AAAI Press, 2014.

[70] D. N. Gitari, Z. Zuping, D. Hanyurwimfura, and J. Long. A lexicon-based approach for hate speech detection. *International Journal of Multimedia and Ubiquitous Engineering*, 10:215–230, 2015.

[71] S. Godbole, I. Bhattacharya, A. Gupta, and A. Verma. Building re-usable dictionary repositories for real-world text mining. In *Proceedings of the International Conference on Information and Knowledge Management (CIKM)*, pages 1189–1198. ACM, 2010.

[72] J. Golbeck, Z. Ashktorab, R. O. Banjo, A. Berlinger, S. Bhagwan, C. Buntain, P. Cheakalos, A. A. Geller, Q. Gergory, R. K. Gnanasekaran, R. R. Gunasekaran, K. M. Hoffman, J. Hottle, V. Jienjitlert, S. Khare, R. Lau, M. J. Martindale, S. Naik, H. L. Nixon, P. Ramachandran, K. M. Rogers, L. Rogers, M. S. Sarin, G. Shahane, J. Thanki, P. Vengataraman, Z. Wan, and D. M. Wu. A large labeled corpus for online harassment research. In *Proceedings of the International Web Science Conference (WebSci)*, pages 229–233. ACM, 2017.

[73] V. Gómez, H. J. Kappen, N. Litvak, and A. Kaltenbrunner. A likelihood-based framework for the analysis of discussion threads. *World Wide Web*, 16(5-6):645–675, 2013.

[74] E. Grave, P. Bojanowski, P. Gupta, A. Joulin, and T. Mikolov. Learning word vectors for 157 languages. In *Proceedings of the Language Resources and Evaluation Conference (LREC)*. ELRA, 2018.

[75] A. Graves. Generating sequences with recurrent neural networks. *arXiv preprint arXiv:1308.0850*, 2013.

[76] A. Grover and J. Leskovec. node2vec: Scalable feature learning for networks. In *Proceedings of the Conference on Knowledge Discovery and Data Mining (KDD)*, pages 855–864. ACM, 2016.

[77] J. Guberman, C. Schmitz, and L. Hemphill. Quantifying toxicity and verbal violence on twitter. In *Proceedings of the Conference on Computer Supported Cooperative Work (CSCW)*, pages 277–280. ACM, 2016.

[78] M. Häring, W. Loosen, and W. Maalej. Who is addressed in this comment?: Automatically classifying meta-comments in news comments. *Human-Computer Interaction (HCI)*, 2(67):1–20, 2018.

[79] R. He and J. McAuley. Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering. In *Proceedings of the International Conference on World Wide Web (WWW)*, pages 507–517. ACM, 2016.

[80] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997.

[81] B. Howe. Cde: A tool for creating portable experimental software packages. *Computing in Science & Engineering*, 14(4):32–35, 2012.

[82] C.-F. Hsu, E. Khabiri, and J. Caverlee. Ranking comments on the social web. In *International Conference on Computational Science and Engineering (CSE)*, pages 90–97. IEEE, 2009.

[83] M. Hsueh, K. Yogeeswaran, and S. Malinen. "Leave your comment below": Can biased online comments influence our own prejudicial attitudes and behaviors? *Human Communication Research*, 41(4):557–576, 2015.

[84] M. O. Ibrohim and I. Budi. Multi-label hate speech and abusive language detection in Indonesian Twitter. In *Proceedings of the Workshop on Abusive Language Online (ALW@ACL)*, pages 46–57. ACL, 2019.

[85] A. Jaech, V. Zayats, H. Fang, M. Ostendorf, and H. Hajishirzi. Talking to the crowd: What do people react to in online discussions? In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2026–2031. ACL, 2015.

[86] S. Jain and B. C. Wallace. Attention is not Explanation. In *Proceedings of the North American Chapter of the Association for Computational Linguistics (NAACL)*, pages 3543–3556. ACL, 2019.

[87] Y. Janin, C. Vincent, and R. Duraffort. Care, the comprehensive archiver for reproducible execution. In *Proceedings of the Workshop on Reproducible Research Methodologies and New Publication Models in Computer Engineering (Trust@PLDI)*, pages 1–7. ACM, 2014.

[88] A. Jha and R. Mamidi. When does a compliment become sexist? analysis and classification of ambivalent sexism using twitter data. In *Proceedings of the Workshop on Natural Language Processing and Computational Social Science (NLP+CSS@ACL)*, pages 7–16. ACL, 2017.

[89] A. Joulin, E. Grave, P. Bojanowski, and T. Mikolov. Bag of tricks for efficient text classification. In *Proceedings of the Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, pages 427–431. ACL, 2017.

[90] G. Kennedy, A. McCollough, E. Dixon, A. Bastidas, J. Ryan, C. Loo, and S. Sahay. Technology solutions to combat online harassment. In *Proceedings of the Workshop on Abusive Language Online (ALW@ACL)*, pages 73–77. ACL, 2017.

[91] G. Kennedy, A. McCollough, E. Dixon, A. Bastidas, J. Ryan, C. Loo, and S. Sahay. Technology solutions to combat online harassment. In *Proceedings of the Workshop on Abusive Language Online (ALW@ACL)*, pages 73–77. ACL, 2017.

[92] Y. Kim. Convolutional neural networks for sentence classification. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751. ACL, 2014.

[93] P.-J. Kindermans, K. T. Schütt, M. Alber, K.-R. Müller, D. Erhan, B. Kim, and S. Dähne. Learning how to explain neural networks: PatternNet and PatternAttribution. In *Proceedings of the International Conference on Learning Representations (ICLR)*, pages 1–16. OpenReview.net, 2018.

[94] R. Kobayashi and R. Lambiotte. TiDeH: Time-dependent hawkes process for predicting retweet dynamics. In *Proceedings of the International Conference on Web and Social Media (ICWSM)*, pages 191–200. AAAI Press, 2016.

[95] V. Kolhatkar and M. Taboada. Using new york times picks to identify constructive comments. In *Proceedings of the Natural Language Processing meets Journalism Workshop (NLPmJ@EMNLP)*, pages 100–105. ACL, 2017.

[96] V. Kolhatkar, H. Wu, L. Cavasso, E. Francis, K. Shukla, and M. Taboada. The sfu opinion and comments corpus: A corpus for the analysis of online news comments. *Corpus Pragmatics*, 4(2):155–190, 2019.

[97] J. Kovačević. How to encourage and publish reproducible research. In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1273–1276. IEEE, 2007.

[98] R. Kumar, M. Mahdian, and M. McGlohon. Dynamics of conversations. In *Proceedings of the Conference on Knowledge Discovery and Data Mining (KDD)*, pages 553–562. ACM, 2010.

[99] R. Kumar, A. K. Ojha, S. Malmasi, and M. Zampieri. Benchmarking Aggression Identification in Social Media. In *Proceedings of the Workshop on Trolling, Aggression and Cyberbullying (TRAC@COLING)*, pages 1–11. ACL, 2018.

[100] R. Kumar, A. N. Reganti, A. Bhatia, and T. Maheshwari. Aggression-annotated Corpus of Hindi-English Code-mixed Data. In *Proceedings of the Language Resources and Evaluation Conference (LREC)*, pages 1425–1431. ELRA, 2018.

[101] R. Kumar, A. K. Ojha, S. Malmasi, and M. Zampieri. Evaluating aggression and misogyny identification in social media. In *Proceedings of the Workshop on Trolling, Aggression and Cyberbullying (TRAC@LREC)*, pages 1–5. ELRA, 2020.

[102] V. Künstler. Modeling user behavior in online discussions on news platforms. Master's thesis, University of Potsdam, Hasso Plattner Institute, Prof.-Dr.-Helmert-Str. 2-3, D-14482 Potsdam, 2019.

[103] I. Kwok and Y. Wang. Locate the hate: Detecting Tweets Against Blacks. In *Proceedings of the Conference on Artificial Intelligence (AAAI)*, pages 1621–1622. AAAI Press, 2013.

[104] H. Lakkaraju, J. McAuley, and J. Leskovec. What's in a name? understanding the interplay between titles, content, and communities in social media. In *Proceedings of the International Conference on Web and Social Media (ICWSM)*, pages 1–10. AAAI Press, 2013.

[105] C. Lampe and P. Resnick. Slash (dot) and burn: distributed moderation in a large online conversation space. In *Proceedings of the Conference on Human Factors in Computing Systems*, pages 543–550. ACM, 2004.

[106] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. Backpropagation applied to handwritten zip code recognition. *Neural Computation*, 1(4):541–551, 1989.

[107] J. Li, W. Monroe, and D. Jurafsky. Understanding neural networks through representation erasure. *arXiv preprint arXiv:1612.08220*, 2016.

[108] Q. Li, J. Wang, Y. P. Chen, and Z. Lin. User comments for news recommendation in forum-based social media. *Information Sciences*, 180(24):4929–4939, 2010.

[109] N. Ljubešić, T. Erjavec, and D. Fišer. Datasets of Slovene and Croatian moderated news comments. In *Proceedings of the Workshop on Abusive Language Online (ALW@EMNLP)*, pages 124–131. ACL, 2018.

[110] T. Mandl, S. Modha, P. Majumder, D. Patel, M. Dave, C. Mandlia, and A. Patel. Overview of the hasoc track at fire 2019: Hate speech and offensive content identification in indo-european languages. In *Proceedings of the Forum for Information Retrieval Evaluation*, page 14–17. ACM, 2019.

[111] G. S. Manku, A. Jain, and A. Das Sarma. Detecting near-duplicates for web crawling. In *Proceedings of the International Conference on World Wide Web (WWW)*, pages 141–150. ACM, 2007.

[112] A. N. Medvedev, J.-C. Delvenne, and R. Lambiotte. Modelling structure and predicting dynamics of discussion threads in online boards. *Journal of Complex Networks*, 7(1):67–82, 2019.

[113] Y. Mehdad and J. R. Tetreault. Do characters abuse more than words? In *Proceedings of the Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL)*, pages 299–303. ACM, 2016.

[114] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In *Proceedings of the International Conference on Neural Information Processing Systems (NeurIPS)*, pages 3111–3119. Curran Associates Inc., 2013.

[115] J. Mitrović, B. Birkeneder, and M. Granitzer. nlpup at semeval-2019 task 6: a deep neural language model for offensive language detection. In *Proceedings of the International Workshop on Semantic Evaluation (SemEval@NAACL)*, pages 722–726. ACL, 2019.

[116] V. Mnih, N. Heess, A. Graves, and K. Kavukcuoglu. Recurrent models of visual attention. In *Advances in Neural Information Processing Systems (NIPS)*, pages 2204–2212. Curran Associates Inc., 2014.

[117] D. Monroe. AI, explain yourself. *Communications of the ACM*, 61(11):11–13, 2018.

[118] G. Montavon, S. Lapuschkin, A. Binder, W. Samek, and K.-R. Müller. Explaining nonlinear classification decisions with deep taylor decomposition. *Pattern Recognition*, 65:211–222, 2017.

[119] H. Mubarak, D. Kareem, and M. Walid. Abusive Language Detection on Arabic Social Media. In *Proceedings of the Workshop on Abusive Language Online (ALW@ACL)*, pages 52–56. ACL, 2017.

[120] H. Mulki, H. Haddad, C. Bechikh Ali, and H. Alshabani. L-HSAB: A Levantine Twitter dataset for hate speech and abusive language. In *Proceedings of the Workshop on Abusive Language Online (ALW@ACL)*, pages 111–118. ACL, 2019.

[121] C. Napoles, A. Pappu, and J. R. Tetreault. Automatically identifying good conversations online (yes, they do exist!). In *Proceedings of the International Conference on Web and Social Media (ICWSM)*, pages 628–631. AAAI Press, 2017.

[122] C. Napoles, J. Tetreault, A. Pappu, E. Rosato, and B. Provenzale. Finding good conversations online: The yahoo news annotated comments corpus. In *Proceedings of the Linguistic Annotation Workshop (LAW@EACL)*, pages 13–23, 2017.

[123] D. Nguyen. Comparing automatic and human evaluation of local explanations for text classification. In *Proceedings of the North American Chapter of the Association for Computational Linguistics (NAACL)*, pages 1069–1078. ACL, 2018.

[124] C. Nobata, J. R. Tetreault, A. Thomas, Y. Mehdad, and Y. Chang. Abusive language detection in online user content. In *Proceedings of the International Conference on World Wide Web (WWW)*, pages 145–153. ACM, 2016.

[125] N. Ousidhoum, Z. Lin, H. Zhang, Y. Song, and D.-Y. Yeung. Multilingual and multi-aspect hate speech analysis. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing and the International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4675–4684. ACL, 2019.

[126] H. Pandit, R. G. Hamed, S. Lawless, and D. Lewis. The use of open data to improve the repeatability of adaptivity and personalisation experiment. In *Proceedings of Workshop Towards Comparative Evaluation in User Modeling, Adaptation and Personalization (Eval@UMAP)*, pages 1–3. CEUR, 2016.

[127] D. Park, S. Sachar, N. Diakopoulos, and N. Elmqvist. Supporting comment moderators in identifying high quality online news comments. In *Proceedings of the Conference on Human Factors in Computing Systems*, pages 1114–1125. ACM, 2016.

[128] J. H. Park and P. Fung. One-step and two-step classification for abusive language detection on twitter. In *Proceedings of the Workshop on Abusive Language Online (ALW@ACL)*, pages 41–45. ACL, 2017.

[129] J. Pavlopoulos, P. Malakasiotis, and I. Androutsopoulos. Deeper attention to abusive user content moderation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1125–1135. ACL, 2017.

[130] J. Pavlopoulos, P. Malakasiotis, J. Bakagianni, and I. Androutsopoulos. Improved abusive comment moderation with user embeddings. In *Proceedings of the Natural Language Processing meets Journalism Workshop (NLPmJ@EMNLP)*, pages 51–55. ACL, 2017.

[131] T. Pedersen. Empiricism is not a matter of faith. *Computational Linguistics*, 34 (3):465–470, 2008.

[132] J. Pennington, R. Socher, and C. D. Manning. Glove: Global vectors for word representation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543. ACL, 2014.

[133] Q. Pham, T. Malik, and I. T. Foster. Using provenance for repeatability. In *Proceedings of the Workshop on the USENIX Theory and Practice of Provenance*, pages 1–4, 2013.

[134] T. Pires, E. Schlinger, and D. Garrette. How multilingual is multilingual BERT? In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 4996–5001. ACL, 2019.

[135] Z. Pitenis, M. Zampieri, and T. Ranasinghe. Offensive language identification in Greek. In *Proceedings of the Language Resources and Evaluation Conference (LREC)*, pages 5113–5119. ELRA, 2020.

[136] G. K. Pitsilis, H. Ramampiaro, and H. Langseth. Effective hate-speech detection in twitter data using recurrent neural networks. *Applied Intelligence*, 48(12):4730–4742, 2018.

[137] M. Ptaszynski, J. K. K. Eronen, and F. Masui. Learning deep on cyberbullying is always better than brute force. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, pages 3–10. IJCAI Organization, 2017.

[138] M. Ptaszynski, A. Pieciukiewicz, and P. Dybała. Results of the poleval 2019 shared task 6: First dataset and open shared task for automatic cyberbullying detection in polish twitter. In *Proceedings of the PolEval Workshop*, pages 89–110. Polish Academy of Sciences, 2019.

[139] J. Qian, M. ElSherief, E. M. Belding-Royer, and W. Y. Wang. Leveraging intra-user and inter-user representation learning for automated hate speech detection. In *Proceedings of the North American Chapter of the Association for Computational Linguistics (NAACL)*, pages 118–123. ACL, 2018.

[140] J. Qian, A. Bethke, Y. Liu, E. Belding, and W. Y. Wang. A benchmark dataset for learning to intervene in online hate speech. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing and the International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4755–4764. ACL, 2019.

[141] H. Rahmatizadeh Zagheli, H. Zamani, and A. Shakery. A semantic-aware profile updating model for text recommendation. In *Proceedings of the Conference on Recommender Systems (RecSys)*, pages 316–320. ACM, 2017.

[142] A. H. Razavi, D. Inkpen, S. Uritsky, and S. Matwin. Offensive language detection using multi-level classification. In *Proceedings of the Canadian Conference on Advances in Artificial Intelligence (Canadian AI)*, pages 16–27. Springer, 2010.

[143] S. Rendle. Factorization machines. In *Proceedings of the International Conference on Data Mining (ICDM)*, pages 995–1000. IEEE, 2010.

[144] M. Rezvan, S. Shekarpour, L. Balasuriya, K. Thirunarayan, V. L. Shalin, and A. Sheth. A quality type-aware annotated corpus and lexicon for harassment research. In *Proceedings of the International Web Science Conference (WebSci)*, page 33–36. ACM, 2018.

[145] M. H. Ribeiro, P. H. Calais, Y. A. Santos, V. A. F. Almeida, and W. M. Jr. Characterizing and detecting hateful users on twitter. In *Proceedings of the International Conference on Web and Social Media (ICWSM)*, pages 676–679. AAAI Press, 2018.

[146] M. T. Ribeiro, S. Singh, and C. Guestrin. "Why should I trust you?": Explaining the predictions of any classifier. In *Proceedings of the Conference on Knowledge Discovery and Data Mining (KDD)*, pages 1135–1144. ACM, 2016.

[147] J. Risch and R. Krestel. What should i cite? cross-collection reference recommendation of patents and papers. In *Proceedings of the International Conference on Theory and Practice of Digital Libraries (TPDL)*, pages 40–46. Springer, 2017.

[148] J. Risch and R. Krestel. Aggression identification using deep learning and data augmentation. In *Proceedings of the Workshop on Trolling, Aggression and Cyberbullying (TRAC@COLING)*, pages 150–158. ACL, 2018.

[149] J. Risch and R. Krestel. My approach = your apparatus? entropy-based topic modeling on multiple domain-specific text collections. In *Proceedings of the Joint Conference on Digital Libraries (JCDL)*, pages 283–292. ACM, 2018.

[150] J. Risch and R. Krestel. Delete or not delete? semi-automatic comment moderation for the newsroom. In *Proceedings of the Workshop on Trolling, Aggression and Cyberbullying (TRAC@COLING)*, pages 166–176. ACL, 2018.

[151] J. Risch and R. Krestel. Learning patent speak: Investigating domain-specific word embeddings. In *Proceedings of the International Conference on Digital Information Management (ICDIM)*, pages 63–68. IEEE, 2018.

[152] J. Risch and R. Krestel. Domain-specific word embeddings for patent classification. *Data Technologies and Applications*, 53(1):108–122, 2019.

[153] J. Risch and R. Krestel. Measuring and facilitating data repeatability in web science. *Datenbank-Spektrum*, 19(2):117–126, 2019.

[154] J. Risch and R. Krestel. Toxic comment detection in online discussions. In B. Agarwal, R. Nayak, N. Mittal, and S. Patnaik, editors, *Deep Learning-Based Approaches for Sentiment Analysis*, Algorithms for Intelligent Systems, pages 85–109. Springer, first edition, 2020.

[155] J. Risch and R. Krestel. Bagging BERT models for robust aggression identification. In *Proceedings of the Workshop on Trolling, Aggression and Cyberbullying (TRAC@LREC)*, pages 55–61. ELRA, 2020.

[156] J. Risch and R. Krestel. Top comment or flop comment? predicting and explaining user engagement in online news discussions. In *Proceedings of the International Conference on Web and Social Media (ICWSM)*, pages 579–589. AAAI Press, 2020.

[157] J. Risch and R. Krestel. A dataset of journalists' interactions with their readership: When should article authors reply to reader comments? In *Proceedings of the International Conference on Information and Knowledge Management (CIKM)*. ACM, 2020. in print.

[158] J. Risch, S. Garda, and R. Krestel. Book recommendation beyond the usual suspects: Embedding book plots together with place and time information. In *Proceedings of the International Conference On Asia-Pacific Digital Libraries (ICADL)*, pages 227–239. Springer, 2018.

[159] J. Risch, A. Stoll, M. Ziegele, and R. Krestel. hpiDEDIS at GermEval 2019: Offensive language identification using a German BERT model. In *Proceedings of the Conference on Natural Language Processing (KONVENS)*, pages 403–408. German Society for Computational Linguistics and Language Technology (GSCL), 2019.

[160] J. Risch, S. Garda, and R. Krestel. Hierarchical document classification as a sequence generation task. In *Proceedings of the Joint Conference on Digital Libraries (JCDL)*, pages 147–155. ACM, 2020.

[161] J. Risch, V. Künstler, and R. Krestel. HyCoNN: hybrid cooperative neural networks for personalized news discussion recommendation. In *under review*, 2020.

[162] J. Risch, R. Ruff, and R. Krestel. Explaining offensive language detection. *Journal for Language Technology and Computational Linguistics (JLCL)*, 34(1):29–47, 2020.

[163] J. Risch, R. Ruff, and R. Krestel. Offensive language detection explained. In *Proceedings of the Workshop on Trolling, Aggression and Cyberbullying (TRAC@LREC)*, pages 137–143. ELRA, 2020.

[164] G. Rizos, S. Papadopoulos, and Y. Kompatsiaris. Predicting news popularity by mining online discussions. In *Companion Proceedings of the International Conference on World Wide Web (WWW Companion)*, pages 737–742. ACM, 2016.

[165] D. Robinson, Z. Zhang, and J. Tepper. Hate speech detection on twitter: Feature engineering v.s. feature selection. In *Proceedings of the Extended Semantic Web Conference (ESWC)*, pages 46–49. Springer, 2018.

[166] S. Rosenthal, P. Atanasova, G. Karadzhov, M. Zampieri, and P. Nakov. A large-scale semi-supervised dataset for offensive language identification. *arXiv preprint arXiv:2004.14454*, 2020.

[167] L. Rösner, S. Winter, and N. C. Krämer. Dangerous minds? effects of uncivil online comments on aggressive cognitions, emotions, and behavior. *Computers in Human Behavior*, 58:461–470, 2016.

[168] B. Ross, M. Rist, G. Carbonell, B. Cabrera, N. Kurowsky, and M. Wojatzki. Measuring the Reliability of Hate Speech Annotations: The Case of the European Refugee Crisis. In *Proceedings of the Workshop on Natural Language Processing for Computer-Mediated Communication (NLP4CMC@KONVENS)*. University Frankfurt, 2016.

[169] K. Y. Rozier and E. W. D. Rozier. Reproducibility, correctness, and buildability: The three principles for ethical public dissemination of computer science and engineering research. In *Proceedings of the International Symposium on Ethics in Engineering, Science, and Technology (ETHICS)*, pages 1–13. IEEE, 2014.

[170] R. Ruff. Explanations for text categorization. Bachelor's thesis, University of Passau, Innstraße 41, D-94032 Passau, 2019.

[171] J. Salminen, H. Almerekhi, M. Milenković, S.-g. Jung, J. An, H. Kwak, and B. J. Jansen. Anatomy of online hate: developing a taxonomy and machine learning models for identifying and classifying hate in online news media. In *Proceedings of the International Conference on Web and Social Media (ICWSM)*, pages 330–339. AAAI Press, 2018.

[172] N. S. Samghabadi, S. Maharjan, A. Sprague, R. Diaz-Sprague, and T. Solorio. Detecting nastiness in social media. In *Proceedings of the Workshop on Abusive Language Online (ALW@ACL)*, pages 63–72. ACL, 2017.

[173] M. Sanguinetti, F. Poletto, C. Bosco, V. Patti, and M. Stranisci. An Italian Twitter corpus of hate speech against immigrants. In *Proceedings of the Language Resources and Evaluation Conference (LREC)*, pages 2798–2805. ELRA, 2018.

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

[175] A. Schmidt and M. Wiegand. A survey on hate speech detection using natural language processing. In *Proceedings of the International Workshop on Natural Language Processing for Social Media (SocialNLP@EACL)*, pages 1–10. ACL, 2017.

[176] S. Seo, J. Huang, H. Yang, and Y. Liu. Interpretable convolutional neural networks with dual local and global attention for review rating prediction. In *Proceedings of the Conference on Recommender Systems (RecSys)*, pages 297–305. ACM, 2017.

[177] E. Shmueli, A. Kagian, Y. Koren, and R. Lempel. Care to comment?: Recommendations for commenting on news stories. In *Proceedings of the International Conference on World Wide Web (WWW)*, pages 429–438. ACM, 2012.

[178] S. Siersdorfer, S. Chelaru, W. Nejdl, and J. San Pedro. How useful are your comments? analyzing and predicting youtube comments and comment ratings. In *Proceedings of the International Conference on World Wide Web (WWW)*, pages 891–900. ACM, 2010.

[179] G. I. Sigurbergsson and L. Derczynski. Offensive language and hate speech detection for Danish. In *Proceedings of the Language Resources and Evaluation Conference (LREC)*, pages 3498–3508. ELRA, 2020.

[180] K. Simonyan, A. Vedaldi, and A. Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. In *Workshop Proceedings of the International Conference on Learning Representations (ICLR)*, pages 1–8. OpenReview.net, 2014.

[181] S. Sonnenburg, M. L. Braun, C. S. Ong, S. Bengio, L. Bottou, G. Holmes, Y. LeCun, K.-R. Müller, F. Pereira, C. E. Rasmussen, et al. The need for open source software in machine learning. *Journal of Machine Learning Research*, 8(Oct):2443–2466, 2007.

[182] E. Spertus. Smokey: Automatic recognition of hostile messages. In *Proceedings of the National Conference on Artificial Intelligence and Conference on Innovative Applications of Artificial Intelligence (AAAI /IAAI)*, pages 1058–1065. AAAI Press, 1997.

[183] J. T. Springenberg, A. Dosovitskiy, T. Brox, and M. A. Riedmiller. Striving for simplicity: The all convolutional net. In *Workshop Proceedings of the International Conference on Learning Representations (ICLR)*, pages 1–14. OpenReview.net, 2015.

[184] R. Sprugnoli, S. Menini, S. Tonelli, F. Oncini, and E. Piras. Creating a WhatsApp dataset to study pre-teen cyberbullying. In *Proceedings of the Workshop on Abusive Language Online (ALW@EMNLP)*, pages 51–59. ACL, 2018.

[185] N. J. Stroud, E. Van Duyn, and C. Peacock. News commenters and news comment readers. *Engaging News Project*, pages 1–21, 2016.

[186] J. M. Struß, M. Siegel, J. Ruppenhofer, M. Wiegand, and M. Klenner. Overview of GermEval task 2, 2019 shared task on the identification of offensive language. In *Proceedings of the Conference on Natural Language Processing (KONVENS)*, pages 352–363. German Society for Computational Linguistics and Language Technology (GSCL), 2019.

[187] M. Sundararajan, A. Taly, and Q. Yan. Axiomatic attribution for deep networks. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 3319–3328. JMLR.org, 2017.

[188] A. Tatar, J. Leguay, P. Antoniadis, A. Limbourg, M. D. de Amorim, and S. Fdida. Predicting the popularity of online articles based on user comments. In *Proceedings of the International Conference on Web Intelligence, Mining and Semantics (WIMS)*, pages 67:1–67:8. ACM, 2011.

[189] A. Tatar, P. Antoniadis, M. D. de Amorim, and S. Fdida. Ranking news articles based on popularity prediction. In *Proceedings of the International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*, pages 106–110. IEEE, 2012.

[190] M. Thelwall, K. Buckley, G. Paltoglou, D. Cai, and A. Kappas. Sentiment strength detection in short informal text. *Journal of the Association for Information Science and Technology*, 61(12):2544–2558, 2010.

[191] M. Tsagkias, W. Weerkamp, and M. de Rijke. Predicting the volume of comments on online news stories. In *Proceedings of the International Conference on Information and Knowledge Management (CIKM)*, pages 1765–1768. ACM, 2009.

[192] M. Tsagkias, W. Weerkamp, and M. de Rijke. News comments: Exploring, modeling, and online prediction. In *Proceedings of the European Conference on Information Retrieval (ECIR)*, pages 191–203. Springer, 2010.

[193] S. Tulkens, L. Hilte, E. Lodewyckx, B. Verhoeven, and W. Daelemans. The automated detection of racist discourse in dutch social media. *Computational Linguistics in the Netherlands Journal*, 6:3–20, 2016.

[194] B. van Aken, J. Risch, R. Krestel, and A. Löser. Challenges for toxic comment classification: An in-depth error analysis. In *Proceedings of the Workshop on Abusive Language Online (ALW@EMNLP)*, pages 33–42. ACL, 2018.

[195] B. van Aken, B. Winter, A. Löser, and F. A. Gers. How does BERT answer questions? a layer-wise analysis of transformer representations. In *Proceedings of the International Conference on Information and Knowledge Management (CIKM)*, page 1823–1832. ACM, 2019.

[196] C. Van Hee, E. Lefever, B. Verhoeven, J. Mennes, B. Desmet, G. De Pauw, W. Daelemans, and V. Hoste. Detection and fine-grained classification of cyberbullying events. In *Proceedings of the International Conference Recent Advances in Natural Language Processing (RANLP)*, pages 672–680. INCOMA Ltd., 2015.

[197] P. Vandewalle, J. Kovacevic, and M. Vetterli. Reproducible research in signal processing. *Signal Processing Magazine*, 26(3):37–47, 2009.

[198] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems (NIPS)*, pages 5998–6008. Curran Associates Inc., 2017.

[199] F. D. Vigna, A. Cimino, F. Dell'Orletta, M. Petrocchi, and M. Tesconi. Hate me, hate me not: Hate speech detection on facebook. In *Proceedings of the Italian Conference on Cybersecurity (ITASEC)*, pages 86–95. CEUR, 2017.

[200] J. Vitek and T. Kalibera. Repeatability, reproducibility, and rigor in systems research. In *Proceedings of the International Conference on Embedded Software (EMSOFT)*, pages 33–38. ACM, 2011.

[201] C. Wang and D. M. Blei. Collaborative topic modeling for recommending scientific articles. In *Proceedings of the Conference on Knowledge Discovery and Data Mining (KDD)*, pages 448–456. ACM, 2011.

[202] C. Wang, M. Ye, and B. A. Huberman. From user comments to on-line conversations. In *Proceedings of the Conference on Knowledge Discovery and Data Mining (KDD)*, pages 244–252. ACM, 2012.

[203] W. Warner and J. Hirschberg. Detecting hate speech on the world wide web. In *Proceedings of the Workshop on Language in Social Media (LSM@ACL)*, pages 19–26. ACL, 2012.

[204] Z. Waseem. Are you a racist or am i seeing things? annotator influence on hate speech detection on twitter. In *Proceedings of the Workshop on NLP and Computational Social Science (NLP+CSS@EMNLP)*, pages 138–142. ACL, 2016.

[205] Z. Waseem and D. Hovy. Hateful symbols or hateful people? predictive features for hate speech detection on twitter. In *Proceedings of the Student Research Workshop@NAACL*, pages 88–93. ACL, 2016.

[206] Z. Waseem, T. Davidson, D. Warmsley, and I. Weber. Understanding abuse: A typology of abusive language detection subtasks. In *Proceedings of the Workshop on Abusive Language Online (ALW@ACL)*, pages 78–84. ACL, 2017.

[207] M. Wiegand, M. Siegel, and J. Ruppenhofer. Overview of the GermEval 2018 shared task on the identification of offensive language. In *Proceedings of the Conference on Natural Language Processing (KONVENS)*, pages 1–10. Austrian Academy of Sciences, 2018.

[208] S. Wiegreffe and Y. Pinter. Attention is not not explanation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing and the International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 11–20. ACL, 2019.

[209] E. Wulczyn, N. Thain, and L. Dixon. Ex machina: Personal attacks seen at scale. In *Proceedings of the International Conference on World Wide Web (WWW)*, pages 1391–1399. ACM, 2017.

[210] G. Xiang, B. Fan, L. Wang, J. I. Hong, and C. P. Rosé. Detecting offensive tweets via topical feature discovery over a large scale twitter corpus. In *Proceedings of the International Conference on Information and Knowledge Management (CIKM)*, pages 1980–1984. ACM, 2012.

[211] Z. Yang, D. Yang, C. Dyer, X. He, A. Smola, and E. Hovy. Hierarchical attention networks for document classification. In *Proceedings of the North American Chapter of the Association for Computational Linguistics (NAACL)*, pages 1480–1489. ACL, 2016.

[212] D. Yin, Z. Xue, L. Hong, B. D. Davison, A. Kontostathis, and L. Edwards. Detection of harassment on web 2.0. *Proceedings of the Workshop on Content Analysis in the WEB (CAW@WWW)*, pages 1–7, 2009.

[213] T. Yoneda, S. Kozawa, K. Osone, Y. Koide, Y. Abe, and Y. Seki. Algorithms and system architecture for immediate personalized news recommendations. In *Proceedings of the International Conference on Web Intelligence (WI)*, pages 124–131. ACM, 2019.

[214] M. Zampieri, S. Malmasi, P. Nakov, S. Rosenthal, N. Farra, and R. Kumar. Predicting the type and target of offensive posts in social media. In *Proceedings*

of the North American Chapter of the Association for Computational Linguistics (NAACL), pages 1415–1420. ACL, 2019.

[215] M. Zampieri, S. Malmasi, P. Nakov, S. Rosenthal, N. Farra, and R. Kumar. SemEval-2019 task 6: Identifying and categorizing offensive language in social media (OffensEval). In *Proceedings of the International Workshop on Semantic Evaluation (SemEval@NAACL)*, pages 75–86. ACL, 2019.

[216] M. Zampieri, P. Nakov, S. Rosenthal, P. Atanasova, G. Karadzhov, H. Mubarak, L. Derczynski, Z. Pitenis, and c. Çöltekin. SemEval-2020 Task 12: Multilingual Offensive Language Identification in Social Media (OffensEval 2020). In *Proceedings of the International Workshop on Semantic Evaluation (SemEval@COLING)*. ACL, 2020. in print.

[217] M. D. Zeiler and R. Fergus. Visualizing and understanding convolutional networks. In *European Conference on Computer Vision (ECCV)*, pages 818–833. Springer, 2014.

[218] Q. Zhang, Y. Gong, J. Wu, H. Huang, and X. Huang. Retweet prediction with attention-based deep neural network. In *Proceedings of the International Conference on Information and Knowledge Management (CIKM)*, pages 75–84. ACM, 2016.

[219] Z. Zhang and L. Luo. Hate speech detection: A solved problem? the challenging case of long tail on twitter. *Semantic Web Journal*, pages 1–21, 2018.

[220] Z. Zhang, D. Robinson, and J. A. Tepper. Detecting hate speech on twitter using a convolution-gru based deep neural network. In *Proceedings of the Extended Semantic Web Conference (ESWC)*, pages 745–760. Springer, 2018.

[221] L. Zheng, V. Noroozi, and P. S. Yu. Joint deep modeling of users and items using reviews for recommendation. In *Proceedings of the International Conference on Web Search and Data Mining (WSDM)*, pages 425–434. ACM, 2017.

[222] H. Zhong, H. Li, A. C. Squicciarini, S. M. Rajtmajer, C. Griffin, D. J. Miller, and C. Caragea. Content-driven detection of cyberbullying on the instagram social network. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, pages 3952–3958. AAAI Press, 2016.

[223] M. Zignani, C. Quadri, A. Galdeman, S. Gaito, and G. P. Rossi. Mastodon content warnings: Inappropriate contents in a microblogging platform. In *Proceedings of the International Conference on Web and Social Media (ICWSM)*, pages 639–645. AAAI Press, 2019.

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

Declaration

I hereby confirm that

- this dissertation is the result of my own work, it was prepared without unauthorized help and using only the given literature,
- this dissertation has not been previously submitted, in part or whole, to any other university,
- I am aware of the doctorate regulations of the Digital Engineering Faculty of the University of Potsdam from November 27, 2019.

Ich erkläre hiermit, dass

- ich die vorliegende Dissertationsschrift selbständig und ohne unerlaubte Hilfe angefertigt sowie nur die angegebene Literatur verwendet habe,
- die Dissertation keiner anderen Hochschule in gleicher oder ähnlicher Form vorgelegt wurde,
- mir die Promotionsordnung der Digital Engineering Fakultät der Universität Potsdam vom 27. November 2019 bekannt ist.

Julian Risch – October 5, 2020