Bachelor's Thesis
2022

Adversarial Attacks in German Hate Speech Detection

Author
Luisa Wunderlin
17-702-192

Supervisor
Janis Goldzycher

Department for Computational Linguistics
University of Zurich

Hand-In Date
01.12.2022

Abstract

Adversarial attacks are an ongoing challenge for hate speech detection. Adversarial attacks are alterations to correctly classified examples in an attempt to bring about a misclassification, often in an attempt to avoid detection. They are meant to be unintelligible to a model but easily readable for humans. In this thesis I test two German hate speech detection models on their robustness against 6 different character-level adversarial attacks. The experimental set-up follows the methods described by Gröndahl et al. (2018) in their "All You Need Is 'Love'" paper. While models have improved since Gröndahl et al. (2018) conducted their experiments, they are still strongly affected even by very simple attacks. I found that the models are most susceptible to severe attacks that strongly alter the entire sentence, such as removing all whitespaces, adding whitespaces between letters and leetspeak.

Zusammenfassung

Adversarial Attacks sind eine anhaltende Herausforderung für das Gebiet der Hate Speech Detection. Adversarial Attacks verändern korrekt klassifizierte Texte mit der Absicht, eine falsche Klassifizierung herbeizuführen. Sie sollen für ein Modell unverständlich, aber für Menschen gut leserlich bleiben. In dieser Arbeit teste ich zwei deutsche Hate Speech Detection Modelle auf ihre Robustheit gegenüber 6 verschiedenen Angriffen auf Zeichenebene. Der Versuchsaufbau folgt den Methoden, die von Gröndahl et al. (2018) in "All You Need Is 'Love'" beschrieben wurden. Obwohl die Modelle seit den Experimenten von Gröndahl et al. (2018) besser geworden sind, werden sie selbst von sehr einfachen Angriffen immer noch stark beeinträchtigt. Die Modelle sind am anfälligsten für schwere Angriffe, die den gesamten Satz stark verändern, wie z. B. das Entfernen aller Leerzeichen, das Hinzufügen von Leerzeichen zwischen Buchstaben und Leetspeak.

Acknowledgments

I would like to thank the authors of the 'All You Need is "Love"' paper, who generously made their codebase available to me and whose work formed the basis and starting point for my thesis: Tommi Gröndahl, Luca Pajola, Mika Juuti, Mauro Conti and N. Asokan.

Thank you to the Perspective team, who very kindly raised my API request limit. Thank you also to the HateMonitors team - Punyajoy Saha, Binny Mathew, Pawan Goyal and Animesh Mukherjee - whose model and code I tested and utilized in my experiments.

Lastly, I would like to thank my supervisor, Janis Goldzycher, who supported me throughout the process with exceptional expertise and compassion.

Contents

1 Introduction

Hate speech has become an unfortunate part of our online landscape.

The anonymity and global nature of social media has made it difficult for governments to take action against it. Platforms like Twitter, Facebook and YouTube all have policies against hate speech, but these often remain unenforced. (Espinosa Anke et al., 2019) Recently, the acquisition of Twitter through Elon Musk has brought about a new spike of hate speech and slurs being tweeted. Some slurs tripled in usage in the first week under new ownership, despite platform policies remaining unchanged. (counterhate.com, 2022)

Yet combating online hate speech is essential in order to limit the spread of harmful ideas and to keep hate speech from escalating into violence. (Guterres, 2019) As the volume of data has become unsustainable to parse manually, companies, governments and researchers have turned to automatic solutions.

In recent years, a lot of research has been conducted on automatic hate speech detection, meaning the field is constantly evolving. Modern machine learning approaches leverage large amounts of annotated data in order to build and fine-tune classification models. (Espinosa Anke et al., 2019) The goal of the majority of these models is to classify a text as either hate or non-hate, framing hate speech detection as a classification task. (Gröndahl et al., 2018)

One of the challenges that sets hate speech detection apart from other classification tasks is that the authors of hateful content have an interest in their text being misclassified.

Adversarial attacks are an attempt at bringing about such a misclassfication. They are texts which are created or altered with the intent to fool a model. (Goodfellow et al., 2015) Commonly used by users who are trying to bypass a hate speech filter or detection system, they are designed to appear innocent or unintelligible to an algorithm, but be clearly legible for a human user.

These adversarial attacks - and how German hate speech detection systems perform against them - are the subject of this thesis.

Gröndahl et al. (2018) found that several English hate speech detection systems were vulnerable against adversarial attacks, across model architectures. They employed 6 different mostly character-based adversarial attacks and compared their effects and effectiveness. Following their experimental set-up, I applied the same methods to three different datasets and tested two German hate speech detection models on the resulting adversarial examples.

Since 2018, much has happened in terms of hate speech detection models. Many authors have conducted similar studies and found adversarial attacks to be an ongoing challenge. However, the large majority of research on adversarial attacks focuses on English language models and datasets.

There have been efforts to close this gap between English and other languages and mul-

tilingual systems have become more common. (Röttger et al., 2022) I decided to focus on German instead of English, as Gröndahl et al. (2018) had done, because German is considerably less well-researched.

In this thesis, I aimed to answer the following research questions:

1. How do modern German hate speech detection systems perform against character-level adversarial attacks?
2. What types of attacks are the detection systems most vulnerable against?

In the following chapter, I will further introduce adversarial attacks and the paper by Gröndahl et al. (2018) that this thesis is based on. Chapters 3 and 4 give an overview over the models and datasets I tested. Chapter 5 introduces the experiment and data collection, its results are presented in Chapter 6. I present two follow-up experiments in Chapter 7, before discussing the results further in Chapter 8 and concluding the thesis in Chapter 9.

2 Background

2.1 Hate Speech

There exists no universally agreed upon definition of hate speech, which means each paper, dataset and organisation differ slightly in what they do or do not include. I will discuss the specific definitions used by the models and datasets I tested in Chapter 5.1, but for now I will introduce hate speech more generally.

After conducting a survey on different hate speech definitions, Fortuna and Nunes (2018) characterized the different definitions by four features: Hate speech has specific targets (1), is meant to incite violence or hate (2), attacks or diminishes (3) and affords humour a specific status (4). Not all of these features need to be present - in fact, none of the surveyed definitions had all four. The first feature, however, was present in every definition: Hate speech targets people based on specific group characteristics.

Fortuna and Nunes (2018) synthesized these results into a definition of their own: "Hate speech is language that attacks or diminishes, that incites violence or hate against groups, based on specific characteristics such as physical appearance, religion, descent, national or ethnic origin, sexual orientation, gender identity or other, and it can occur with different linguistic styles, even in subtle forms or when humour is used."

2.2 Adversarial Attacks

Adversarial attacks are examples which are created with the intent to fool a system or model. Goodfellow et al. (2015) define adversarial examples as "inputs formed by applying small but intentionally worst-case perturbations to examples from the dataset", with the intention of making the model output "an incorrect answer with high confidence". An adversarial attack can consist of slight or severe alterations to a previously correctly classified example of various types. The goal is to alter the examples in such a way that the model misclassifys them, while retaining their readability to human users.

Adversarial attacks are not exclusive to hate speech detection or even natural language processing in general. An example of an adversarial attack on an image recognition model would be to e.g. add a layer of noise to an image, so that it looks unaltered to a person but is classified differently by the model. (Goodfellow et al., 2015)

The successful deception of a hate speech detection model can go either way, with innocent examples being classified as hateful *or* hateful examples classified as innocent. However, the goal in this context is usually to *evade* detection. As such, the adversarial attacks have a desired targeted output, this being non-hateful. (Alsmadi et al., 2021)

There are different types of adversarial attacks in NLP, which Alsmadi et al. (2021) group into four categories:

1. **Character-level** attacks: Substituting, adding or deleting characters in a word (leetspeak, typos, adding whitespaces, etc.)
   Example: "*I want to k ! l l you.*"

2. **Word-level** attacks: Replacing words with adversarial synonyms which are harder to classify
   Example: "*I want to end you.*"

3. **Sentence-level** attacks: Adding, deleting or paraphrasing sentences
   Example: "*I'd love to send you to heaven today.*"

4. **Inter-level** attacks: Combination of character-, word- and sentence-level attacks

Semantic changes, such as word- and sentence level attacks, are complex, since they require an understanding of grammar and sentence structure. In many cases, human users might not even identify them as an attempt to evade detection. (Bitton et al., 2022) Character-level attacks are more straightforward in comparison, only making changes to individual letters.

Adversarial attacks leverage model weaknesses in order to produce faulty outcomes, and as such knowledge of model architecture, functions and parameters provides an advantage. (Alsmadi et al., 2021) In most real-world scenarios, however, attackers won't have insight into the model they are trying to trick. Therefore model-specific weaknesses can only be discovered and exploited through trial and error.

This doesn't mean that each system is completely individual in its weak points. As Goodfellow et al. (2015) observed, many systems share weaknesses for the same type of attack and even misclassify them in the same way. This happens across models and training data sets.

2.3 All You Need Is "Love"

My thesis was inspired by the 'All You Need Is "Love"' paper by Gröndahl et al. (2018), in which the authors tested several state-of-the-art hate speech detection models. They showed that models only perform well on the same type of data they were trained on and are brittle against automatic adversarial attacks.

As discussed in the previous section, there are many different types of adversarial attacks of varying degrees of complexity. Character-level attacks are the most straightforward and easily generated with an algorithm, since they don't require any understanding of the altered text.

Gröndahl et al. (2018) found that all models were vulnerable to adversarial attacks, across model architectures and training datasets, though to varying degrees. They implemented 6 mostly character-level attacks due to them being easy to automatically generate.

These are the 6 different types of adversarial attacks Gröndahl et al. (2018) employed:

0. Add typos
   Example: *i wnat to klil you*

1. Leetspeak
   Example: *1 w4nt t0 k1ll y0u*

2. Remove white spaces
   Example: *iwanttokillyou*

3. Add random white spaces
   Example: *i w ant to k ill you*

4. Add extreme white spaces
   Example: *i w a n t t o k i l l y o u*

5. 'All You Need Is Love': Remove white space and append love to end
   Example: *iwanttokillyou love*

The last type of attack also adds a word-level component by adding the positive word 'love' in order to mislead the classifiers. Gröndahl et al. (2018) argue that the effectiveness of this attack is indicative of a larger problem in hate speech detection systems: They treat the task as a classification task, instead of a detection task. This means that positive words are able to balance out hate speech. Instead, any amount of hate speech should make the example hate speech, since it is a *detection* task.

3 Models

I selected two hate speech detection systems to test on their robustness against adversarial attacks.

The first one is Perspective, a state-of-the art toxic language detection model. It was launched in 2017 and has been used commercially as a content-moderation tool (e.g. on the New York Times website). (*Perspective* 2021) I selected it because it is a commercial model which has received a lot of funding over the years and is actively being used in real-world applications.

The second model is HateMonitors, which was developed for the HASOC 2019 shared task. I selected HateMonitors because it is a highly ranked submission coming out of an academic competition. While HateMonitors is not as high-profile as Perspective, the developers have made all their code public and documented their process, which allows for a much better insight into the model's workings.

3.1 Perspective API

Perspective is a toxic language detection model created by Jigsaw and Google. It is freely available to use as an API.

The Perspective API takes text as input and returns a score from 0 to 1 for several categories. In the demo available on the Perspective website, anything above 0.7 is classified as toxic. The seven available attributes are Toxicity, Severe Toxicity, Insult, Profanity, Threat, Sexually Explicit and Identity Attack. The API is available for 18 languages, including German.

Perspective is currently in the process of being transitioned to a new model. Some newly introduced languages already operate on the new Unified Toxic Content Classification (UTC) model, while previously available languages still run on the old model. (Lees et al., 2022) German is among the languages which have yet to be transitioned to UTC.

While detailed documentation is available for UTC, little has been made public about Perspective's older model. Perspective's documentation simply describes the model as using "machine learning models to score the perceived impact a comment might have on a conversation". (*Perspective* 2021)

Rieder and Skop (2021) report that a production version of Perspective used the Global Vectors for Word Representation (GloVe) algorithm to encode texts. The resulting word embeddings were then passed to Google's TensorFlow framework and used to train neural networks. Perspective has received many updates over the years, so the model and preprocessing pipeline have likely changed to some extent. How much has changed is hard to tell, though Jigsaw has never announced a complete renewal of model architectures as they are in the process of doing now.

The data Perspective is trained on stems from various sources, using partners like newspapers and websites for data sourcing as well as publicly available datasets. (Rieder and Skop, 2021) By default, Perspective also stores all comments which are requested for scoring through their API, though it is possible to opt out of this. Unfortunately, which partners or datasets were used for German, Jigsaw never made public.

3.2 HateMonitors

The HateMonitors model was created by Saha et al. (2019) as a submission for the HASOC shared task at FIRE 2019. The objective of the task was to create a hate speech and offensive language detection model for English, Hindi and German.

The task was split into two subtasks. The goal of subtask A was to build a binary classifier, which sorted tweets into whether they contained hateful or offensive languages (HOF) or not (NOT). Subtask B aimed to refine the classification from subtask A into hate speech (HATE), offensive (OFFN) and profane (PRFN). An additional subtask was available for English and Hindi, but not German. (Mandl et al., 2019)

The HateMonitors team placed first for the German subtask A and second for subtask B.

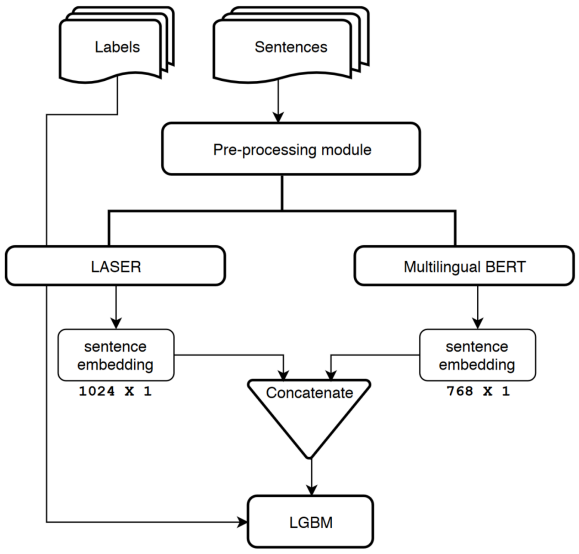

Figure 1: HateMonitors Model Architecture

A visualization of the model's pipeline can be seen in Figure 1, which was taken from the paper by Saha et al. (2019).

As a first preprocessing step, all text is converted to lowercase. Any numbers are normalized to a "number" string and all URLS are removed. Punctuation, stop-words and usernames are left untouched.

An other important part of HateMonitors' preprocessing - which Saha et al. (2019) did

not document, but can be seen in the code - is fastBPE. BPE, which stands for byte pair encoding, is a data compression technique. FastBPE makes use of the same algorithm applied to characters in order to segment rare and out-of-vocabulary words into subword units. (Sennrich et al., 2016) In HateMonitor's pipeline, these subwords are then passed on to be embedded.

Next, feature vectors for the posts are generated. The preprocessed text is passed through two embedding models, BERT (Bidirectional Encoder Representations from Transformers) and LASER (Language Agnostic Sentence Embeddings Representations).

Saha et al. (2019) used the BERT multilingual base model developed by Google, which was pretrained on 104 languages. It is case sensitive, though the text inputs are lowercased in preprocessing. The model output is a vector of length 768. The second embedding model, LASER, was developed by Facebook and trained on 93 languages, making both embedding models multilingual. The output consists of a vector of length 1024. The two resulting vectors are concatenated into a single feature vector of total length 1792, which is then passed as input to the classifier.

For the classifier, Saha et al. (2019) made use of Gradient boosted trees, specifically LGBM (Light Gradient Boosting Machine). LGBM is a framework based on decision trees and was selected by Saha et al. (2019) because of its efficient memory usage and because it had been successful in past competitions.

4 Datasets

I chose three different datasets to test Perspective and HateMonitors on. Two of the datasets contain Twitter data, with one being a pure Twitter dataset and the other mixed with Facebook comments. The third dataset is sourced from a newspaper's website.

4.1 HASOC 2019

The German HASOC 2019 dataset was published as part of the same shared task the HateMonitors model was developed for. The dataset comes pre-split into training and test set, adding up to a total of 4'199 social media posts from Twitter and Facebook.

Mandl et al. (2019) collected the data by filtering recent posts for topics and hashtags which typically draw a lot of hate. They also recorded the entire timeline of some of the users to ensure the model will not be trained on keywords only. While the authors did not disclose a specific list of keywords, some of the topics become apparent when manually exploring the dataset. The majority of the posts are in some way concerned with politics, often mentioning political parties (e.g. *Alternative für Deutschland, Grüne*). Keywords such as *Flüchtlinge* (refugees), *Migranten* (immigrants) and *Muslime* (muslims) occur frequently.

The collected data was annotated by eight amateur annotators for two different tasks. The annotators received brief guidelines including definitions for these labels. If the annotators disagreed on a case, they were able to discuss amongst each other, but were not required to reach a consensus. Wagner and Bumann (2020) note that cases including irony and in-group knowledge proved especially challenging for the annotators. The annotators reached an interrater agreement of 88% for the first and 86.51% for the second task. (Mandl et al., 2019)

Since the data was annotated for the shared task, it follows its split into two subtasks. For the first subtask, annotators were asked to decide if a post contained hateful, offensive or profane content (HOF) or not (NOT). For the second, the annotators then had to further specify if it included hate speech (HATE), offensive (OFFN) or profane (PRFN) content. I included two examples below for better visualization of the annotation structure.

text_id	text	task_1	task_2
hasoc_de_1205	Mein Freund in Südtirol hat seinen kleinen Weinkeller erweitert. Bin bis auf weiteres offline! :) Bis bald! [LINK]	NOT	NONE
hasoc_de_3252	@USER Vermutlich will eine bestimmte Gruppe diese Gegend für sich beanspruchen und die Einheimische dort verdrängen. Besonders in Rosengard wohnen überwiegend Muslime. Anders kann ich mir das nicht vorstellen bei dem Terror der dort herrscht.	HOF	OFFN

4.2 DeTox

The DeTox dataset is an extensively annotated Twitter dataset containing 10'278 German Tweets. (Demus et al., 2022)

The data was collected in the first half of 2021, which means the COVID-19 pandemic and the German elections were common topics of discussion. This is especially interesting, since it includes terms like *Querdenker* and COVID-related hate speech, which the HateMonitors model (and possibly the Perspective model) were not trained on.

Taking a similar approach to Mandl et al. (2019), Demus et al. (2022) used a keyword list in order to filter and find relevant posts. They used Google Trends to identify current topics that were likely to inspire hateful comments. As examples, Demus et al. (2022) name keywords such as the political hashtag *#merkelmussweg* ("Merkel must go") but also neutral words like *Jude* (jew), which are often encountered in a hateful context. Demus et al. (2022) also collected not only isolated comments but entire conversations in order to give a more varied view of how hate speech is used online. After data collection, the authors further filtered the data using hate speech and profanity corpora.

The annotators consisted of six students. They received detailed instructions and completed a training phase, which involved annotating comments and then discussing them if they showed a high inter-annotator disagreement. After the training, the annotators were split up in two groups as heterogeneously as possible, so that both groups were equally likely to label something as offensive. Following this split, each group reciecved half the dataset for annotation. The annotation process was supervised throughout and broken up by regular check-ins where the annotators discussed disagreements and unclear cases.

The dataset was annotated for 12 different categories, some binary, some with free text input or a selection of different options. First, the annotators had to decide whether the post was incomprehensible or not. If they answered yes, the rest of the questions were optional.

The other annotation categories were sentiment, hate speech, criminal relevance, expression (implicit/explicit), extremism, target (person/group/public), threat and a few subcategories asking for more detail on the mentioned categories.

For easier readability, I only included the first three categories in the table below.

c_id	c_text	incomp	sentiment	hate_speech
1398753952368369674	wieso het mir keine gseit dass es eh film git wo nazis ufem mond lebe und d erde überfalle	0.33	0.0	0.0
1398535540501368834	@USER Wen man die phsyche der Frauen bei den grünen linken und spd untersuchen würde 😂 😂 😂 😂	0.0	-1.0	0.66

Interestingly, the dataset evidently also includes Swiss German posts, such as the first tweet shown above (*1398753952368369674*), which translates to: "Why did no one tell me there's a movie where the Nazi's live on the moon and attack earth". I assume the collection of this tweet was not intentional, but simply unavoidable due to the similarity between Swiss German and High German. The tweet has a moderate incomprehensibility score, which means that only a part of the annotators understood what it was expressing.

4.3 RP-Crowd

The last dataset is the Moderator- and Crowd-Annotated German News Comment Dataset by Assenmacher et al. (2021), more succinctly named the RP-Mod & RP-Crowd dataset. The dataset includes two sets of annotations, one provided by moderators of a news website, the other by crowdworkers. I used the crowd annotated part for this thesis, since it includes more detailed and standardized labels.

RP stands for *Rheinische Post*, a major German newspaper and the source of the comments in this dataset. The comments were all posted to the newspaper's website between November 2019 and June 2020. Commenting requires a registered account, and each comment is checked by a moderator. The decisions by the moderators to publish the comment or not is what is makes up the RP-Mod portion of this dataset.

Assenmacher et al. (2021) sampled 85'000 comments from the data provided by the *Rheinische Post*. This includes all 7'141 comments the moderators decided not to publish, in order to boost the offensive and abusive categories. The rest of the entries were sampled randomly.

Each entry was annotated by five annotators, which were recruited through the crowdworking platform Crowd Guru. The annotators went through a briefing, where they were informed about the task and subject. They also had to complete a short screening, where they had to correctly label 15 out of 20 gold standard comments. Each participant was allowed to label up to 1'000 comments. Every once in a while, they had to pass an attention check and a gold standard question.

The crowd annotation followed the schema of a comment moderation forum. First, the annotators decided whether or not the comment should be published. If not, they had to select why out of these labels: Sexism, racism, threats, insults and profane language. There were also two categories for meta comments (comments relating to the platform, e.g. asking why someone was blocked) and advertisements.

I only included the first four labels in the examples below and abbreviated them for space reasons. The labels are Reject Newspaper (RN), Reject Crowd (RC), Sexism Count Crowd (SCC) and Racism Count Crowd (RCC).

id	text	RN	RC	SCC	RCC
2149295	Kam mal aus der Kneipe und musste mich tierisch Übergeben. Zwei Straßen weiter kam ich in einer Polizeikontrolle.“ Fahrzeugschein und Führerschein bitte”.“Haben sie was getrunken ”. Ich “Nein” Dann kam der Spruch :“Hauchen Sie mich mal an” Also bei der Polizei,Traumjob und so ,NEIN DANKE	0	0	0.0	0.0
1929173	GENDER = Größte Erreichbare Nebensächliche Dummheit Einer Rasse	0	1	2.0	3.0

5 Experiment Structure

5.1 Mappings

One major issue when using models and datasets from different sources is consistency in labels and mappings.

Definitions and annotation processes can differ greatly between datasets. This makes the evaluation especially difficult when testing a model on data it was not trained on. It can be hard to tell whether the system is annotating 'correctly' according to the hate speech definition of its training set and performing badly because of differences in annotation, or whether it simply does not know how to handle diverse data.

As discussed in Chapter 2.1, hate speech does not have a universally accepted definition, nor do all the models and datasets even have a category labeled "hate speech". I summarized all the relevant definitions from the models and datasets in Table 1.

	Label	Definition
Perspective	Identity Attack	"Negative or hateful comments targeting someone because of their identity." (*Perspective* 2021)
HateMonitors & HASOC 2019	Hate Speech	"Describing negative attributes or deficiencies to groups of individuals because they are members of a group (e.g. all poor people are stupid). Hateful comment toward groups because of race, political opinion, sexual orientation, gender, social status, health condition or similar." (Mandl et al., 2019)
DeTox	Hate Speech	"Hate speech is defined as any form of expression that attacks or disparages persons or groups by characteristics attributed to the groups. Discriminatory statements can be aimed at, for example, political attitudes, religious affiliation or sexual identity of the victims." (Demus et al., 2022)
RP-Crowd	Sexism	"Attacks on people based on their gender (identity), often with a focus on women." (Assenmacher et al., 2021)
	Racism	"Attacks on people based on their origin, ethnicity, nation (typ. meant to incite hatred)." (Assenmacher et al., 2021)

Table 1: Model and Dataset Label Definitions and Mappings

Perspective lacks a hate speech attribute, but they do provide scores for so-called Identity Attacks. The definition of Identity Attacks includes what Fortuna and Nunes (2018)

identified as the core feature of hate speech: Hate speech has specific targets based on group characteristics and identity. Therefore Identity Attacks can be directly mapped to hate speech.

HateMonitors was developed for the HASOC 2019 Shared Task, and as such follows the hate speech definition of its training dataset, the HASOC 2019 dataset. The HASOC and DeTox dataset both include hate speech labels, whose definition largely overlap.

The most complex case is the RP dataset. It is extensively annotated, but lacks a general hate speech category. In order to meaningfully compare the dataset with the others, I combined the racism and sexism label into a general hate speech category.

The obvious issue with this is, of course, that hate speech includes more types of discrimination than just sexism and racism, e.g. based on sexuality, disabilities or religion. The definitions used by Assenmacher et al. (2021) include any discrimination based on gender for sexism, and they define racism as attacks based on "origin, ethnicity, nation". Combining the labels like this means that we will leave out any hate speech aimed at other groups, which might result in an under-representation of hate speech in the dataset.

An additional feature of the racist and sexists labels is that they are not binary, but consist of the count of annotators who voted for something to be either sexist or racist. If two or more annotators out of five agreed on it being hateful, I labeled the post as hateful. I elected to not use a plain majority vote since agreement between annotators was consistently low, and few comments passed the majority vote at all. Setting the threshold at 2 is also the recommendation given by Assenmacher et al. (2021).

5.2 Creating Adversarial Datasets

5.2.1 Selecting Datapoints

In order to test the models' performance against adversarial attacks, I created test sets of 850 entries from each of the three datasets. For the DeTox and RP-Crowd datasets, I selected the entries at random, to best represent the dataset in its entirety. The HASOC 2019 dataset already includes a test dataset, on which the HateMonitors model was not trained. It contains exactly 850 entries, so this is the data I used.

I was limited in the size of the test dataset mostly by how many Perspective API calls I was authorized to make and how fast the HateMonitors model was able to embed and evaluate the data. Since I was testing 850 posts for each of the 3 datasets and applying 6 different adversarial attacks each, I ended up evaluating 2'550 unique unaltered and 15'300 adversarially altered posts.

From the selected 2'550 unique posts, 213 were labeled as hate speech. Hate speech is a comparatively rare phenomena, which often leads to imbalanced datasets, with non-hate examples crowding out the hateful ones. This imbalance was present across the different datasets, though not to the same extent.

Dataset	Label	Amount	
HASOC	Hate Speech	41	4.8 %
DeTox	Hate Speech	133	15.6 %
RP	Sexism	6	0.7%
	Racism	34	4.0%
Total		213	8.4%

Table 2: Hate Speech Distribution in the Datasets

The HASOC and RP dataset contain 41 (4.8%) and 39 (4.5%) hate speech examples respectively, while the DeTox dataset contains almost three times as much with 133 (15.6%) hateful examples. This is representative of the distribution in the entire datasets, with some slight variations due to the random nature of the selection. These numbers are displayed in Table 2. The number of sexist and racist examples in the RP dataset add up to 40 instead of 39 because one of the examples was labeled as both.

5.2.2 Applying Adversarial Attacks

After compiling the unaltered test dataset, I applied 6 adversarial attacks to each of the entries, resulting in seven variations of the same text.

These attacks are the same as used by Gröndahl et al. (2018) and when possible, I used the code kindly provided to me by the authors. For the 5th attack, I substituted the original "love" with the German translation "Liebe".

The attacks are summarized in Table 3. The identifier is how they are referenced in the data and subsequent analysis, with *aa* standing for *adversarial attack*. The original text is marked *ua*, which stands for *unaltered* (data).

Identifier	Adversarial Attack	Example
aa_0	add typos	*was für ein kacklpapen*
aa_1	leetspeak	*w45 für 31n k4ckl4pp3n*
aa_2	remove whitespaces	*wasfüreinkacklappen*
aa_3	add random white spaces	*was f ür ei n kac klappen*
aa_4	add extreme white spaces	*w a s f ü r e i n k a c k l a p p e n*
aa_5	Remove whitespaces and append "Liebe"	*wasfüreinkacklappen Liebe*

Table 3: Overview of the Adversarial Attacks

All of the attacks also remove punctuation and lowercase the text. This does not affect the added "Liebe", which remains uppercase.

The attacks are non-deterministic, meaning re-running the code can yield slightly different results each time. This mostly affects attacks 0 (adding typos) and 3 (add random white spaces), which both include a random element. Random whitespaces inserts, as the name implies, whitespaces randomly into the word. Typos are generated by swapping characters with each other. Characters closer to the middle of a word are more likely to be swapped, as are characters close to each other.

5.3 Data Collection

The next step was having the models process the adversarial data.

For Perspective, this simply meant making an API call for each of the entries and requesting the corresponding Identity Attack score. Everything else was handled on Perspective's end. I submitted a text and the API returned a score from 0 to 1.

The HateMonitors model required more steps. I processed and embedded the data following the authors' process and code. As an intermediate result, I generated a file with entry ids and a large embedding vector for each of the entries. Using the embeddings, the model then classified the data on whether or not it contained any hateful, offensive or profane content. This represents subtask A of the shared task. Using this first classification as a baseline, the model then further classified the entries into hate speech, offensive and profane. As final output, the HateMonitors model returned a single label for each of the entries: HATE, OFFN, PRFN and None.

This concluded the raw data collection.

5.4 Perspective API Threshold

Unlike the HateMonitors model, the Perspective API does not return a label but a score. This means in order to map the score to binary labels, I had to settle on a threshold over which something was hateful with enough confidence.

While the Perspective team do not give any clear guidelines on what an appropriate score is, in the demo on their website the default threshold is 0.7. The score represents the likelihood that a post contains an Identity Attack, so 0.7 would be 70% confidence.

In order to give the model the best possible conditions for these specific datasets, I calculated the results for several thresholds from 0 to 1. In Figure 2, several metrics are plotted against the score thresholds.

As expected, the number of true positives keeps decreasing as the threshold rises. Notably, there's a sharp decrease of true positives around 0.7 in the RP dataset, which is reflected

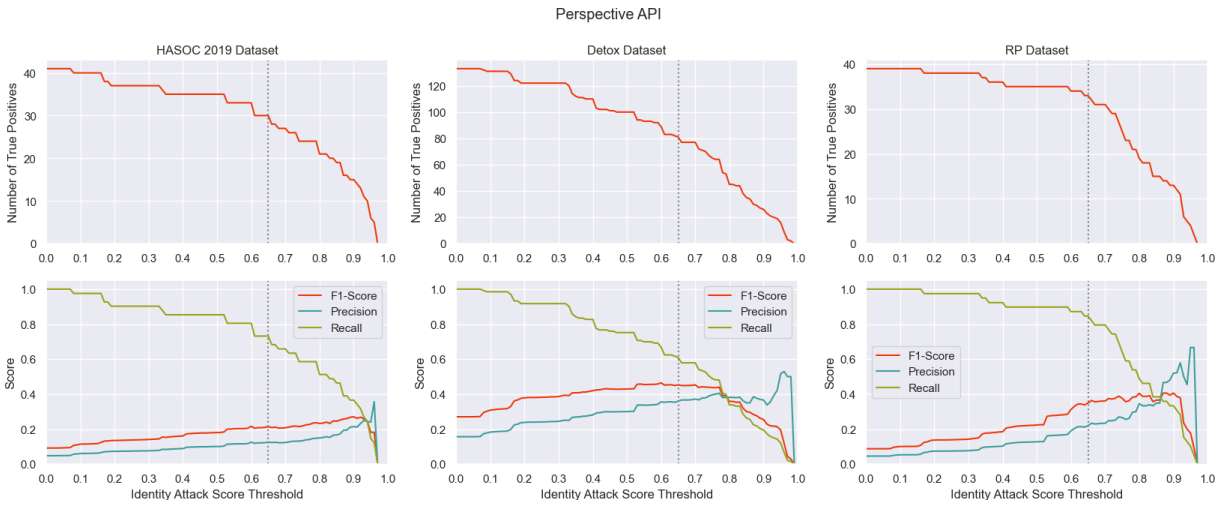

Figure 2: Metrics for different Identity Attack Score Thresholds

in the recall value. F1-score and Precision are generally quite low, but keep increasing steadily before dropping off above 0.8.

I decided to go with a threshold of 0.65, which is marked in Figure 2 as a dotted grey line. I chose this threshold because it strikes a balance between allowing F1-scores for the Detox and RP datasets to almost reach their peak, while retaining the majority of true positives. True positives are essential in order to determine the effectiveness of adversarial attacks, so I wanted to preserve as many of them as possible. Moving forward, all the Perspective results will be calculated with the threshold of 0.65.

5.5 Measuring Success

Before moving on to the results, I want to briefly discuss how I measured success and failure on the adversarial attacks.

F1-scores are very common in the evaluation of classifiers. (Harbecke et al., 2022) While Micro-F1-scores highlight the performance of a model on a particular class, Macro-F1 values give equal weight to all classes.

The problem with averaged F1-scores is that hate speech datasets tend to be highly imbalanced, with hate speech being relatively rare. (Espinosa Anke et al., 2019) Let us consider a dataset of 10 examples, with only one (10%) being hate speech. If the model classifies everything as non-hate speech, it will still achieve a Macro-F1-score of 0.47, despite the F1-score on the hate class being 0.

Hate speech detection is a detection task and adversarial attacks are an attempt to avoid this detection. (Gröndahl et al., 2018) Correctly detected hate speech, or true positives,

are therefore of special interest. If a true positive in the unaltered data is turned into a false negative through an adversarial attack, this means detection was evaded successfully.

I chose F1-scores on the hate class as the basis for my analysis in order to highlight the (un)successful detection of hate speech. This allows for a more intuitive comparison between the performance on the unaltered data and the effects of the adversarial attacks. I also reported other measures such as true/false positives, recall, precision or simply the raw labels and scores where I considered them to enhance the clarity of the attack's effect. For the base model performance, I also reported the Macro-F1-scores.

6 Results

6.1 F1-Scores & Base Performance

The Tables 4 and 5 contain all the F1-scores for the hate speech class, split up by model, datasets and adversarial attacks. In order to establish a baseline, I will first discuss the achieved results on the unaltered data (ua).

Dataset	ua	aa_0	aa_1	aa_2	aa_3	aa_4	aa_5
HASOC 2019	0.216	0.152	0.0	0.114	0.151	0.0	0.028
DeTox	0.465	0.334	0.0	0.267	0.306	0.0	0.156
RP	0.312	0.165	0.0	0.109	0.137	0.0	0.164
All	0.344	0.241	0.0	0.165	0.214	0.0	0.128

Table 4: F1-Scores of the Perspective API on the hate class

Over the entire dataset, Perspective (Table 4) achieved an F1-score of 0.344 on the hate class. Taking the both classes into account (positive and negative predictions), this comes out to a Macro-F1-score of 0.601.

	Dataset	ua	aa_0	aa_1	aa_2	aa_3	aa_4	aa_5
	HASOC	0.0	0.0	0.0	0.0	0.0	0.0	0.0
HATE	DeTox	0.014	0.0	0.0	0.0	0.0	0.0	0.0
	RP	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	All	0.009	0.0	0.0	0.0	0.0	0.0	0.0
HATE	HASOC	0.171	0.107	0.0	0.0	0.104	0.0	0.0
&	DeTox	0.317	0.262	0.0	0.028	0.1	0.0	0.0
OFFN	RP	0.179	0.083	0.0	0.0	0.066	0.0	0.0
	All	0.249	0.167	0.0	0.017	0.093	0.0	0.0

Table 5: F1-Scores for HateMonitors

I included two sets of F1-scores for the Hate Monitors model in Table 5. The first one is calculated as intended, with a true positive being what the model correctly classified as hate speech. Unfortunately, the number of true positives is only one, which results in a F1-score of 0.009 and a Macro-F1-score of 0.481.

Despite placing second in subtask B of the shared task, Saha et al. (2019) reported an F1-score of only 0.04 for the German hate class. While still better than my result, this is a low score, especially when compared with the 0.28 and 0.29 the model achieved for

English and Hindi respectively. Possible reasons for these low scores are discussed in Chapter 8.1.

Since the hate speech category was so underpopulated, I assumed that many hateful posts were instead classified as offensive. In an attempt to recover some of the performance, I treated what the model labeled as offensive as if it had been classified as hate speech. The F1-scores for this are the second set of values in Table 5.

This re-mapping brought up the F1-scores significantly, to 0.249 for the hate speech class and 0.568 for the Macro-average. This put the performance on the unaltered data in a range similar to the Perspective model, if still lower. Moving forward, I will be referencing this set of data when discussing results.

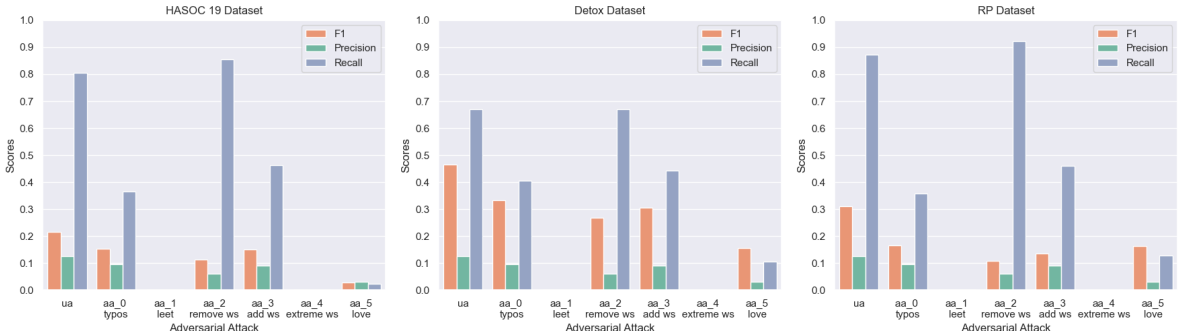

Figure 3: Perspective - F1-Score, Precision and Recall

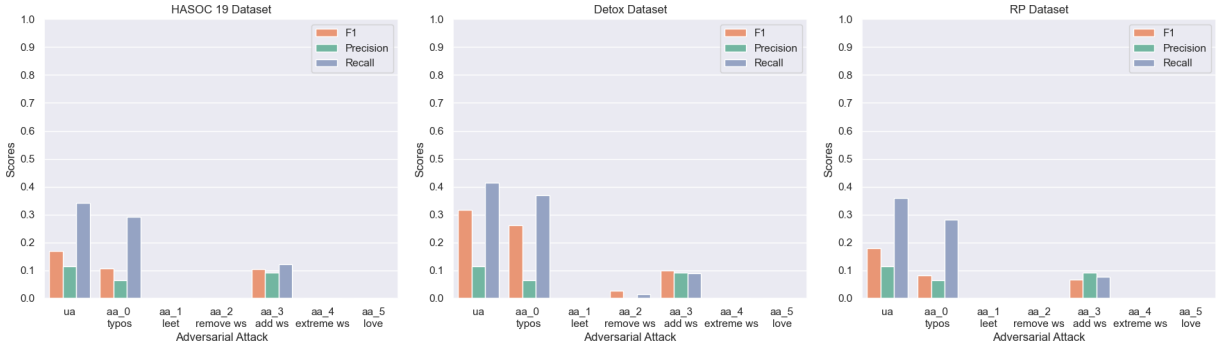

Figure 4: HateMonitors - F1, Precision and Recall

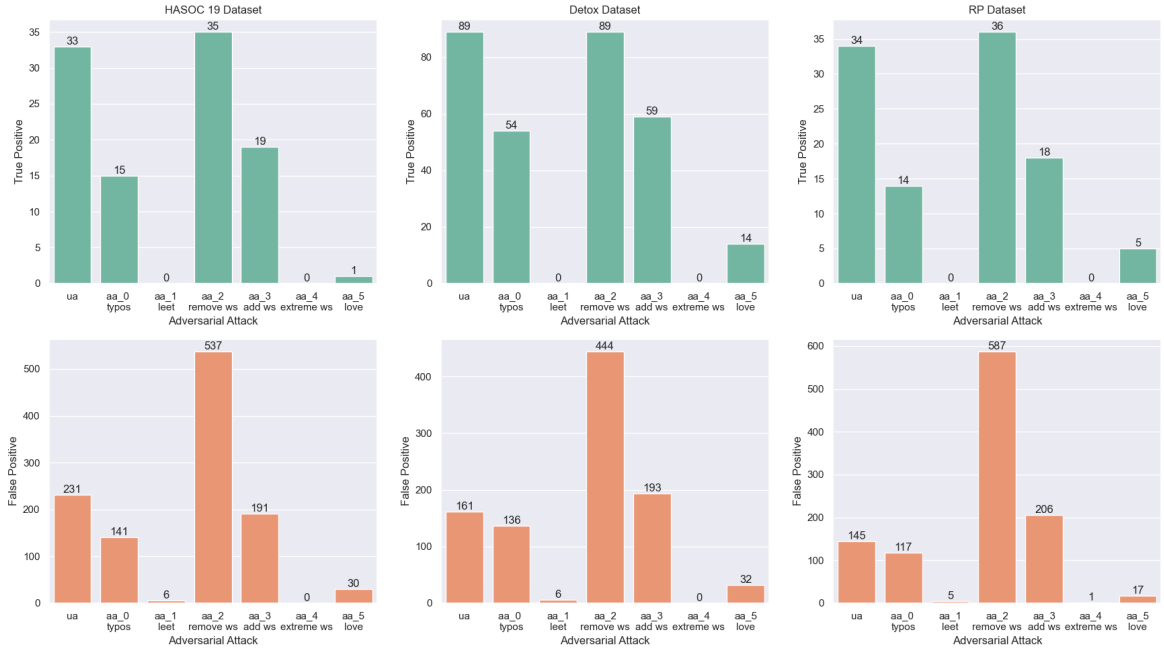

Figure 5: Perspective - Positive Predictions

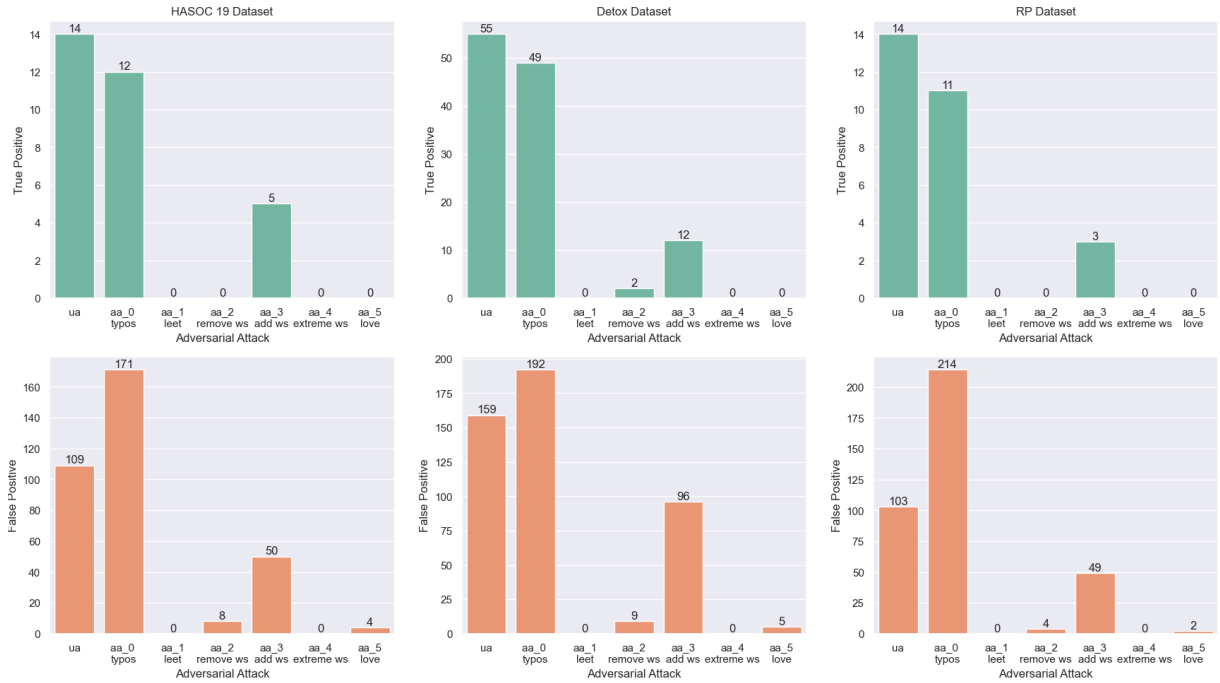

Figure 6: HateMonitors - Positive Predictions

6.2 Typos

Adding typos proved to be the least effective of the adversarial attacks. The attack reduced the F1-scores by 0.1 for Perspective and by 0.08 for HateMonitors.

In Figures 5 and 6, the examples predicted to be be hateful by the models (true positive and false positive) are displayed in barplots. Typos reduced the number of correctly identified true positives, with the effect being stronger for Perspective and more subtle for HateMonitors. Interestingly, they didn't generally make a post less likely to be classified as hate. While the number of false positives is slightly lowered for Perspective, too, (Figure 5), it is higher for HateMonitors (Figure 6).

Just from these figures, it is unclear as to what effect the typos are having on the predictions. In order to visualize the impact on individual examples, I plotted the difference between the Perspective Identity Attack score of the unaltered data and the score of the adversarial data (Figure 7).

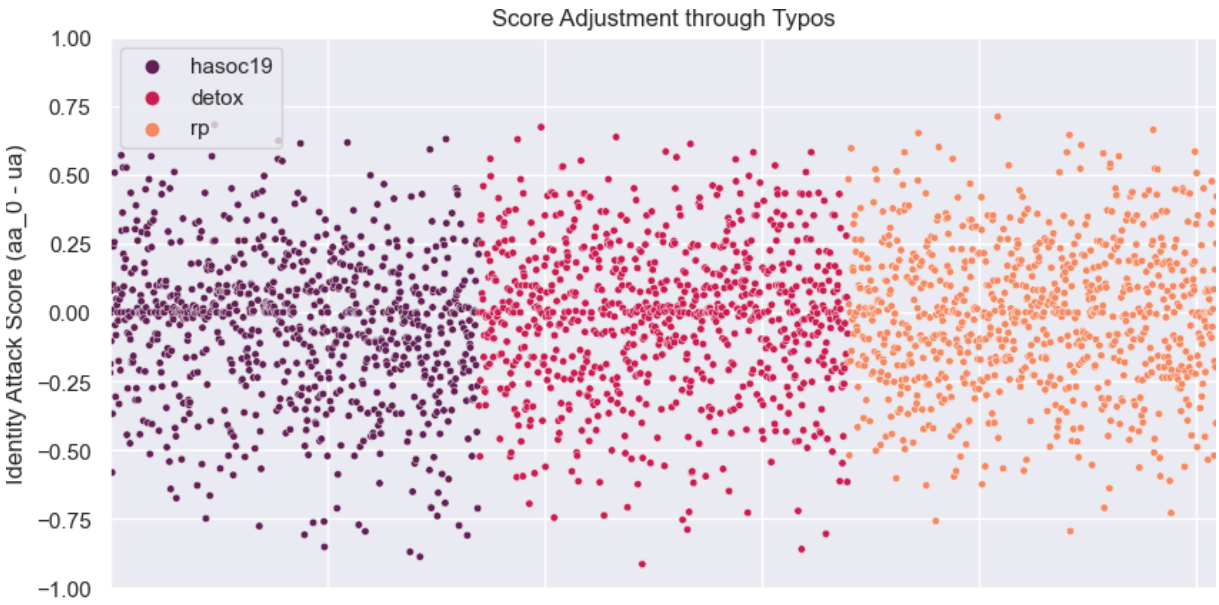

Figure 7: Perspective Identity Attack Score Adjustment through Typos

The score adjustments show no clear trend. There is a cluster of points around zero, for which the attack didn't alter the score, but then the adjustments are spread out in both directions, with a slight bias towards the negative. Generally, the attacks are almost equally as likely to raise the Identity Attack score of an entry as to lower it.

For HateMonitors the typos led to a change in label for 719 out of 2'550 entries (highlighted red in Table 6). Out of these 719 changes, 262 changed from being labeled hate speech (HS) to being labeled not hate speech (Non-HS).

Only 454 of 2'550 entries were labeled true in the unaltered data in the first place, which means over half of the true labels were changed to false. A true label had a 58% likelihood

of being flipped, but a false label only a 22% likelihood.

The increase of false positives compared to true positives in Figure 6 is due to the imbalance in the dataset. When a previously false entry is turned positive, it is much more likely to become a false positive than a true positive, because there are way fewer actually positive cases in the dataset.

		Label aa_0		
		HS	Non-HS	Total
Label ua	HS	192	262	454
	Non-HS	457	1'639	2'096
	Total	649	1'901	2'550

Table 6: HateMonitors Labels Before (ua) and After (aa_0) Applying Typos

6.3 Random Whitespaces

Three of the adversarial attacks utilized whitespaces in some way, of which adding random whitespaces was the least effective. It is also the least severe one, making only a few random alterations to the text. The attack reduced true positives for both models, but otherwise worked very differently on them.

On Perspective, the attack had a very similar effect as the typos. As can be seen in Figure 8, the score adjustments through the attack are spread evenly in both directions, creating a very similar pattern as in Figure 7.

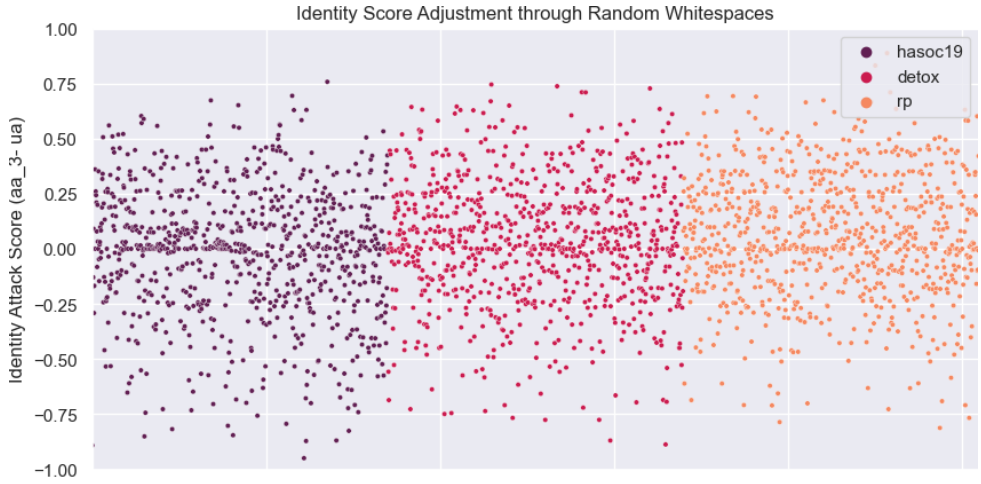

Figure 8: Perspective Identity Score Adjustment through Random Whitespaces

For HateMonitors, random whitespaces did not have the same effect as typos. While

typos had a largely unpredictable effect on the data, random whitespaces consistently turned true labels false.

As can be seen in Table 7, 395 out of 454 examples originally labeled as hate speech were flipped by the attack, which comes out to a 87% likelihood of a positive label being flipped. In contrast, only 7% of originally false labels were turned positive by the attack. This leads to both true and false positives being reduced through the attack.

		Label aa_3		
		HS	Non-HS	Total
Label ua	HS	59	395	454
	Non-HS	156	1'940	2'096
	Total	215	2'335	2'550

Table 7: HateMonitors Labels Before (ua) and After (aa_3) Adding Random Whitespaces

6.4 Suffering From Success: Leetspeak and Extreme Whitespaces

Two of the adversarial attacks proved to be so effective, not a single hateful tweet was detected correctly by either model. These two attacks are leetspeak and extreme whitespaces.

Both Perspective and HateMonitors marked almost all the examples employing leetspeak as non-hate speech, with only Perspective (wrongly) classifying 6 examples as hate. For extreme whitespaces, neither model made a single positive prediction for it.

In HateMonitor's case, the subtask B failed entirely for these two attacks. No entry was classified as hate speech, offensive and/or profane, making further fine-grained classification obsolete.

As for Perspective, many of the entries shared the exact same Identity Attack score of 0.070573. 1'155 of the leetspeak examples and 1'655 of the whitespace examples achieved this score. I assume it represent some type of default value Perspective falls back on, as it also appeared when evaluating the scores for aa_5 (Chapter 6.5).

6.5 Removing Whitespaces and Appending "Love"

Removing whitespaces was an attack that, like random whitespaces, affected Perspective and Hate Monitors differently.

For Perspective, the attack had a mixed effect, biased towards increasing the score. The majority of data points in Figure 9 fall between -0.25 and $+0.7$, which is the amount the attack altered the original Identity Attack score. Overall, removing whitespaces strongly raised positive predictions, the majority of them being false positives.

For HateMonitors, it led to almost all of examples being predicted to be non-hate speech. Only 23 out of 2'550 were classified as hate speech (Table 8). Furthermore, only 2 of these 23 were true positives, making the attack very effective.

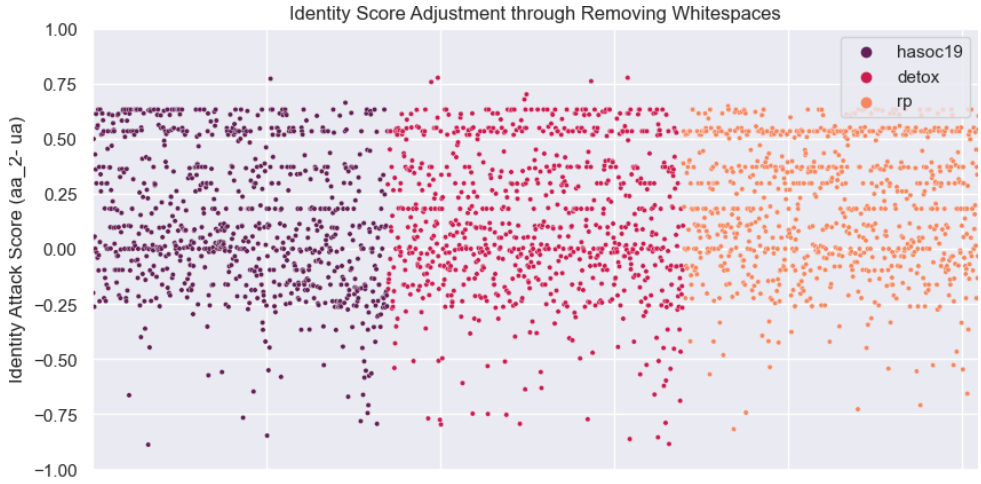

Figure 9: Perspective Identity Attack Score Adjustment through Removing Whitespaces

		Label aa_2		
		HS	Non-HS	Total
Label ua	HS	8	446	454
	Non-HS	15	2'081	2'096
	Total	23	2'527	2'550

Table 8: HateMonitors Labels Before (ua) and After (aa_2) Removing Whitespaces

The 5th attack, which additionally appends "Liebe", was significantly more effective than just removing whitespaces, especially for Perspective. In Figure 10, I plotted all the Identity Attack scores for the adversarial attacks 2 and 5. For many entries, the distribution is largely similar, showing a scattering of scores across the scale. For these points, appending "Liebe" seemed to have had little to no effect.

Around the 0.7 mark, however, over a 1'500 data points crowd together. In the graph for aa_5, (Figure 10) this line is moved down to almost zero. The values of these lines are very exact: The first one is at 0.702169, the second one at 0.070573, the same value that appeared in attack 1 and 4.

In order to show that these really are the same examples clustering together, I subtracted the score of attack 5 from the score of attack 2, leaving me with only the difference caused by "Liebe" alone. This is what the barplot in Figure 11 displays. As expected from Figure 10, "Liebe" had little effect for a number of entries. But for ca. 1'600 examples, "Liebe" adjusted the score by -0.7 to -0.6. This means attack 5 led to a drastic score reduction

not just in general, but specifically for examples that were already uniquely affected by whitespace removal.

For HateMonitors, since whitespace removal already resulted in very few positive predictions, the effect of appending "Love" was less drastic. But with 11 false positives and no true positive, it roughly cut the remaining positive predictions in half.

Figure 10: Identity Attack Scores for aa_2 and aa_5

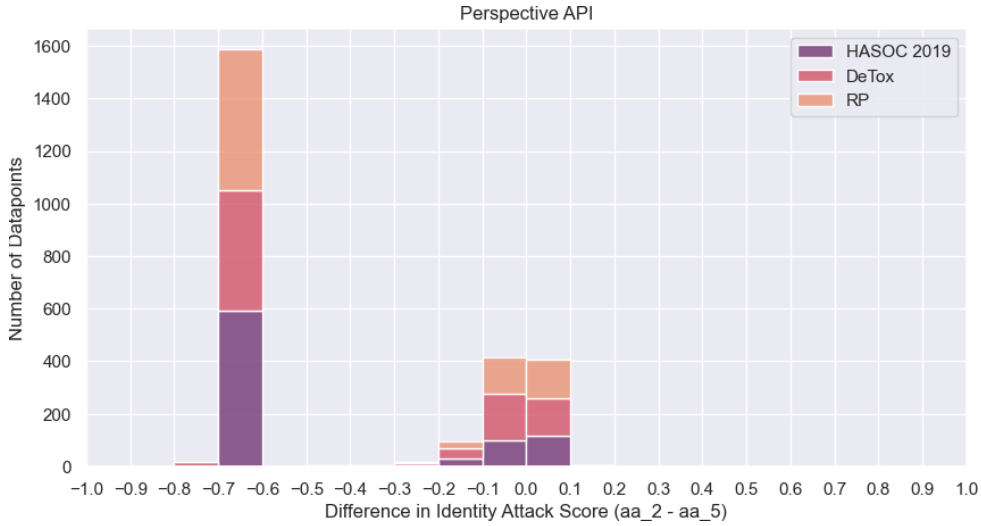

Figure 11: Identity Attack Score Adjustment through Appending "Liebe"

7 Follow-Up Experiments

While analyzing the results of my main experiment, several minor questions arose. In order to better answer them, I conducted two small qualitative experiments. They are meant to further the understanding of the previously achieved results, which is why I chose to separate them from the main experiment and results.

I will present both set-up and results in this chapter, and then discuss them together with the main results in Chapter 8.

7.1 Severity

Many of the adversarial attacks employed alter the text quite severely. The attacks are applied indiscriminately to all words, regardless of their meaning or relevance.

Looking at the results, I wondered if the success of adversarial attacks like typos, random whitespaces and leetspeak could partially be attributed to their extreme severity. Would the attack work equally as well if it was only applied to parts of the texts, specifically potential keywords?

I had Perspective score several versions of a hate speech example on Identity Attacks. The entry, which comes from the RP dataset, was annotated as both sexist and racist by the annotators and its unaltered text received a high Identity Attack score, which is why I selected it. The text translates to "GENDER = The Biggest Achievable Minor Stupidity Of A Race".

	Text	Score
ua	GENDER = Größte Erreichbare Nebensächliche Dummheit Einer Rasse	0.951
aa_1	gender = größte erreichbare nebensächliche dummhe1t einer r4sse	0.966
	g3nd3r = größte erreichbare nebensächliche dummhe1t einer r4ss3	0.609
	g3nd3r = größt3 3rr31chb4r3 n3b3n5ächl1ch3 dummh31t 31n3r r4553	0.331
aa_0	gedner = größte ereichbare nebensächliche dummehit einer rasse	0.959
	gedner = größte ereichbare nebensächliche dummehit einer rsase	0.680
	gedner = grßöte erierchbare nebänsechliche dummehit eienr rsase	0.914
aa_3	gender = größte erreichbare neben sächliche dummheit einer ras se	0.966
	gen der = größte erreichbare neb ens ächliche dummhe it einer ras se	0.865
	gen der = g rößte err eichbar e neb ens ächliche dummh e it ei ner ras se	0.865

Table 9: Identity Attack Scores of Increasingly Severe Adversarial Attacks

I altered the text with three adversarial attacks: leetspeak, typos and random whitespaces. The attacks increase in severity. The texts and associated Identity Attack scores can be

seen in Table 9. I altered potential keywords like *Gender*, *Dummheit* and *Rasse* first, theorizing that their obfuscation would have the biggest impact on the score.

Neither typos nor random whitespaces managed to lower the Identity Attack score below the threshold, independent of severity. For these two attacks, increasing the severity didn't improve the attack's effectiveness, at least for this example. In fact, for typos, the second attack was more successful than the most severe one, while for random whitespaces they achieved the exact same score.

For leetspeak on the other hand, the score dropped with increasing severity. Only the most severe attack lowered the score below the threshold.

The results of this experiment are further discussed in Chapter 8.6.

7.2 How Perspective has improved

While testing the adversarial attacks, I was repeatedly surprised by how strongly the models were affected by them. This raised the question: Have the models not gotten more robust since Gröndahl et al. (2018) tested them, or are German models just lagging behind?

In order to to explore this question, I re-ran a qualitative experiment outlined by Gröndahl et al. (2018) in both English and German.

When Gröndahl et al. (2018) conducted their experiments, Perspective had not yet come out with any other scores except Toxicity. Since Toxicity is not exactly the same as hate speech, Gröndahl et al. (2018) did not include Perspective in their main evaluation. Instead they tested the Toxicity score with a few examples.

A limiting factor for this experiment is, of course, that my main experiment did not evaluate Toxicity but Identity Attack. Considering Toxicity is both Perspective's flagship score and the category first introduced, it is entirely possible that it works differently (and better) than Identity Attack. The function of this experiment is mainly to gauge if Perspective's robustness to adversarial attacks differs significantly between language models.

Gröndahl et al. (2018) used 9 sentences provided as example sentences on the Perspective website at the time, to which they applied their most successful attack, the 'All You Need is "Love"'-approach (aa_5). I translated the nine sentences as closely as possible into German and then had them scored by Perspective for their toxicity.

1. Climate change is happening and it's not changing in our favor. If you think differently you're an idiot.
   Der Klimawandel passiert, und er wird sich nicht zu unseren Gunsten verändern. Wenn du etwas anderes glaubst, bist du ein Idiot.

2. They're stupid, it's getting warmer, we should enjoy it while it lasts

Sie sind dumm, es wird wärmer, wir sollten es genießen, solange es anhält

3. They are liberal idiots who are uneducated
   Sie sind liberale Idioten, die ungebildet sind.

4. idiots. backward thinking people. nationalists. not accepting facts. susceptible to lies.
   Idioten. rückständig denkende Leute. Nationalisten. Fakten nicht akzeptieren. anfällig für Lügen.

5. They are stupid and ignorant with no class
   Sie sind dumm, ignorant und haben kein Niveau

6. It's stupid and wrong
   Es ist dumm und falsch

7. If they voted for Hilary they are idiots
   Wenn sie für Hilary gestimmt haben, sind sie Idioten

8. Anyone who voted for Trump is a moron
   Jeder der für Trump gestimmt hat ist ein Schwachkopf

9. Screw you trump supporters
   Scheiss auf euch Trump-Anhänger

Sentence	Gröndahl et al.		English		German	
	ua	aa_5	ua	aa_5	ua	aa_5
S1	0.95	0.35	0.85	0.60	0.87	0.03
S2	0.92	0.37	0.72	0.50	0.84	0.69
S3	0.98	0.37	0.85	0.57	0.96	0.62
S4	0.95	0.37	0.83	0.50	0.93	0.94
S5	0.97	0.37	0.87	0.62	0.99	0.49
S6	0.88	0.35	0.64	0.36	0.73	0.49
S7	0.99	0.15	0.90	0.64	0.95	0.54
S8	0.96	0.35	0.90	0.43	0.92	0.22
S9	0.90	0.35	0.74	0.57	0.95	0.73

Table 10: Toxicity Scores

The adversarial attack was effective for both English and German. Averaged over all sentences, it lowered the Toxicity score by 0.28 for English and by 0.37 for German.

German was affected slightly more than the English model, but not by much. S1's Toxicity score was reduced to almost 0 by the adversarial attack, which had a big impact on the average. Excluding S1, the attack lowered the score by 0.32.

When Gröndahl et al. (2018) tested the model, the attack lowered all the scores below 0.4. Now, only one of the English sentences was reduced to below 0.4.

8 Discussion

8.1 HateMonitors' Performance

As seen in Chapter 6.1, the HateMonitors model performed very poorly in regards to the hate speech class, even on the unaltered data. In the following sections, I will be discussing possible reasons.

The first explanation is an imbalance in the training dataset. Saha et al. (2019) explain the low performance of the German model partially with the low percentage of hate speech in the training dataset. In the German training dataset, hate speech made up only 2.9%, as opposed to the 19% and 12% in the English and Hindi datasets, which both performed significantly better on the hate class. (Mandl et al., 2019)

This does not explain, however, the drop from the authors' F1-score of 0.04 to 0.009 in my results. Since one of my datasets was the HASOC test dataset, I had a direct comparison between my results and the results by Saha et al. (2019).

Even when working with the original code and datasets, reproducing exact results is often difficult. (Wieling et al., 2018) Small differences in the pipeline can have a large impact on the results. In my case, the discrepancy was likely due to the LASER embeddings. When using the pre-generated embeddings provided by Saha et al. (2019), the model achieved the exact same results as they reported, which leads me to believe the model itself functioned as intended. Once I re-ran the entire pipeline, however, the results differed. When manually comparing the embeddings by the authors with the ones generated by me, I found that while the BERT embeddings were completely identical, the LASER embeddings differed slightly in their numbers. I was ultimately neither able to identify the cause of this nor was I able to fix it.

Another factor are the blurred boundaries between offensive and hate speech. Saha et al. (2019) remark that the HateMonitors model had trouble distinguishing between offensive, hateful and profane language, suggesting that "these models are not able to capture the context in the sentence".

Manual exploration of the results confirmed this. The model only classified 15 posts as hate speech. While it is difficult to reach any clear conclusion with so few data points, the model appeared to rely on keywords like *Nazi*, *Antisemitismus* and *Muslime* in combination with profane language to classify hate, which lead to several examples of highly emotional counterspeak being wrongly classified.

8.2 Perspective's Improvement

In Chapter 7.2, I repeated an experiment by Gröndahl et al. (2018) on Perspective's Toxicity Score. I found that Perspective had become significantly more robust against adversarial attacks since 2018. Additionally, the English and German model performed

equally well against the attacks. This refuted my theory that the German model is simply less advanced than the English one.

What stood out to me about these results, however, is that Toxicity was significantly more robust against the attacks than the Identity Attack score. Attack 5 reduced the majority of Identity Attack scores by -0.7, which in this experiment was only the case for S1. Additionally, for Toxicity, no equivalent to the 'default' value, which the Identity Attack scores clustered around for attack 5, appeared.

8.3 Differences between Datasets

While the datasets differed in many regards, as laid out in Chapter 4, the adversarial attacks had very similar effects on performance across datasets.

I expected HateMonitors to perform better on the HASOC 2019 dataset, since models tend to perform better on the datasets they are trained on, but this was not the case. While there are some variations in F1-scores and true/false positive rates between datasets, most of them are not very pronounced.

The most notable difference in performance was that both models performed significantly better on the DeTox dataset than the RP or HASOC datasets, with higher F1-scores on both the unaltered and adversarial data. This can be explained by DeTox's higher ratio of hate speech examples.

8.4 Typos and Random Whitespaces

Typos and random whitespaces are unique among the adversarial attacks in that they both had a very unpredictable effect on the model predictions. This was especially true for Perspective and to a lesser extend for HateMonitors. The attacks both lowered and raised Identity Attack scores and flipped HateMonitors' predictions both ways.

This does not necessarily mean that typos and random whitespaces are ineffective adversarial attacks, since any misclassification represents a successful attack. But for reliably avoiding detection, typos and random whitespaces are too undirected and random. When looking at true positives turned negative, they were the two least effective adversarial attacks.

Random whitespaces reduced HateMonitors' positive predictions more strongly than typos. I believe this is the case because adding random whitespaces alters word boundaries, creating a challenge for embeddings.

HateMonitors is not completely vulnerable to such word boundary changes, however. As discussed in Chapter 3.2, HateMonitors embedds subword units. The benefit of separating words into subwords is the reduction of out-of-vocabulary words. Since words with adversarially altered word boundaries are also out-of-vocabulary, further segmenting them into

better processable subwords likely improves their chances of being embedded correctly.

I suspect this subword tokenization also made the model slightly more robust against typos. The model was affected by them, but still made both correct and false positive predictions. This is significant because all attacks except typos and random whitespaces resulted in almost zero positive predictions from HateMonitors. From the code we know Saha et al. (2019) did not use a spell checker, so either the attack was not severe enough or the model architecture is in some way robust against typos.

Since Perspective's preprocessing and model architecture are not publicly known, it is more difficult to reason why the model was or was not affected by the attacks. Interestingly, typos and random whitespaces had a very similar effect on its predictions, in both F1-scores and Identity Attack score adjustment.

8.5 Tried and True: Leetspeak

Leetspeak proved to be exceptionally effective, lowering the F1-score to 0 for both Perspective and HateMonitors.

Since it was so effective, I was interested in the 17 examples that were (wrongly) classified as hate, all of them by Perspective.

When looking at the examples manually, I found that not every word in them had been altered by the leetspeak attack. This is, of course, to be expected, since the algorithm only substitutes 5 characters. Any word not containing either a, e, i, o or s will not be altered. In some of the cases, this however led to potential keywords not being obfuscated, as can be seen in the two examples below. For these examples, the words "trump" and "lgbtq" remained unaltered. Both terms are often present in heated political discussions, and the association of neutral terms with hate speech soley due to co-occurence is a well-known phenomena. Perspective's model is however likely more complex than just a collection of words associated with a positive or negative score. When requesting the Identity Attack scores of "Trump" and "LGBTQ" as isolated words, neither of them showed a particularly high score, nor did simple sentences containing them.

id IA score	text ua	text aa_1
2080162	Trump und Kompetenz - Contradictio in adiecto!	trump und k0mp3t3nz c0ntr4d1ct10 1n 4d13ct0
hasoc_de_6008	@Uwe_Junge_MdL Uns wurde es eher mit der AfD schlecht. Was euresgleichen von LGBTQ Menschen hält habt ihr ja schon oft genug deutlich gemacht.	uw3jung3mdl un5 wurd3 35 3h3r m1t d3r 4fd 5chl3cht w45 3ur35gl31ch3n v0n lgbtq m3n5ch3n hält h4bt 1hr j4 5ch0n 0ft g3nug d3utl1ch g3m4cht

Leetspeak is one of only two adversarial attacks resulting in zero true positives, the other being extreme whitespaces. Its effectiveness surprised me, since it is an old and well-known obfuscation technique.

8.6 Severity and Context

In Chapter 7.1, I tested if increasing the severity of an adversarial attack also increased its effectiveness. Both typos and leetspeak are regularly used by human users trying to avoid detection, but usually not to the degree of severity employed in the described adversarial attacks.

A person trying to insult someone might type: "youre such an a$$hole, I can't believe it." The focus of the obfuscation lies on the offensive term, not the words surrounding it. In contrast, the adversarial attacks apply alterations indiscriminately to all the words, sometimes making them exhausting to read even for humans ("yorue such an asohsle i cnat beileve it" and "y0ur3 5uch 4n 455h0l3 1 c4nt b3l13v3 1t").

When testing the different severities on Perspective, I found that random whitespaces and typos did not have a significant effect regardless of severity. This is not an unexpected result, since I found both of these attacks to have very unpredictable effects on Perspective.

Leetspeak however, which I found to be a very successful attack in its most severe form, proved to be significantly less effective when applied only to certain words. Even when all nouns (and potential keywords) were obfuscated in some way, the score did not fall below the threshold. This, as well as the linear relationship between severity and score, imply that the obfuscation of the majority of words really is necessary.

When analyzing how attention models identify hate speech, Kumar et al. (2019) found that the models do not only rely on the offensive term itself, but the term in combination with certain pronouns, verbs, etc. Different models focused on different words in the same sentence. This suggests that altering non-offensive words would also negatively impact the prediction results. If a model can not make sense of a sentence at all, it also will not be able to detect any hate speech in it.

Similar results were found by the Perspective team themselves when testing their newest model, UTC, which is not currently deployed for German.

The developers anticipated that users might try to deceive the model by misspelling words. (Lees et al., 2022) They tested the UTC model on English comments, which were obfuscated to varied degrees of severity. The obfuscation methods included intentional typos *("kil")*, leet speak *("k1ll")* and vowel substitution *("k*ll", "k ll")*. To measure the severity of the alteration, they introduced the obfuscation rate, which is the probability any letter will be replaced by a human-readable substitution.

In their tests, the Perspective API performed significantly worse on obfuscated data. The AUC-ROC score fell mostly linear from 86.1 on the unaltered data to 58.7 on data with

a 50% obfuscation rate. The more characters were altered, the harder the texts were to parse for the model.

8.7 The Power of Love

On the effectiveness of attack 5, Gröndahl et al. (2018) write: "Whitespace removal turns the sentence into a single *<unk>*, making the classification entirely dependent on the model's prediction of this particular token. Models might behave differently with respect to it, and hence the effects of whitespace removal can be uncertain. This problem can be remedied by adding words strongly indicative of the non-hate class, effectively forcing the model to prefer it."

Despite none of the models I tested being strictly word-based, my experiments replicated the effect described by Gröndahl et al. (2018) exactly.

For the HateMonitors model, whitespace removal was already highly effective, reducing the number of true positives to zero. Adding "Liebe" to this did not alter the result in any way. Since the model only gives a binary output, it is difficult to assess if "Liebe" had any additional effect on the final prediction.

As for Perspective, the majority of Identity Attack scores for texts with whitespaces removed lie above the threshold of 0.65. Appending "Liebe" brings them down to almost 0. This is especially true for the cluster of of entries sharing the exact same value.

While I can not claim that Perspective and HateMonitors operate as the word-based model tested by Gröndahl et al. (2018), it is striking how closely my results mirrored theirs. This suggests that the equivalent of an *<unk>* token still exists in some form in these models, with the incomprehensible sentence being classified as hate speech by Perspective and non-hate speech by HateMonitors. Therefore adding "Liebe" only altered Perspective's prediction.

8.8 Preparing for Adversarial Attacks

I have discussed at length how adversarial attacks can reduce model performance. This leaves the question: How can we mitigate the effects of these attacks? I will discuss two different approaches: Catching the attacks during preprocessing and training the models on adversarial data.

8.8.1 Preprocessing

Most classifiers include a preprocessing step, in which the input is normalized in some way. HateMonitors e.g. lowercased the text and normalized URLs and numbers. While there are benefits to normalizing non-adversarial data, text normalization can also help

mitigate character-level adversarial attacks. (Bitton et al., 2022)

Since typos frequently occur in any dataset, many models will include a way to deal with them. One possibility is employing a spell checker during preprocessing. Gröndahl et al. (2018) implemented spell checkers as part of their preprocessing and found that it helped to recover some of the performance lost to the typos attack.

A potential drawback of spell checkers is that they sometimes incorrectly correct a word, turning an out-of-vocabulary word into the wrong in-vocabulary word. While this can lead to problems in all NLP tasks, it is heightened for hate speech detection. As Kumar et al. (2019) write: "In the context of hate speech detection, a problem with standard spell checkers is with their handling of profanity. For example, "sh*t" is corrected to "shot" and "b*tch" to "batch"."

One big advantage of utilizing preprocessing steps to mitigate attacks is that they can be implemented without altering or re-training the model itself. Developers can quickly react to new adversarial attacks as they arise. (Bitton et al., 2022) Gröndahl et al. (2018) recommend a counter-algorithm as a possible solution for leetspeak attacks, since leetspeak is largely formulaic.

Of course there are limits to how severely text can and should be altered in a preprocessing step. There are arguments for not removing attacks like leetspeak in a preprocessing step, such as the information contained in "bad" data. Kumar et al. (2019) argue that intentional obfuscation can be a strong indicator for hate speech, and therefore correcting it is not always advantageous.

8.8.2 Adversarial Training

Another mitigation method is to include adversarial data in the training set or finetune the model on it. The idea behind it is straightforward: If a model has been confronted with a certain type of data before, it is better equipped to handle it. Many model creators have implemented this successfully.

Gröndahl et al. (2018) found that retraining the models on data altered by the leetspeak and typos attack had a strong positive impact on the model's performance. It worked especially well for leetspeak, since it is a deterministic attack. Typos could, in contrast, be generated in any number of ways and therefore the model was only familiar with certain variations.

The developers of the UTC model have also worked with adversarial training. After conducting their experiments on how obfuscation negatively impacts the UTC model (described in Chapter 8.6), Lees et al. (2022) finetuned the model on 30% obfuscated data. As a result, they were able to recover the lost performance almost entirely.

The biggest drawback of adversarial training is that it is very expensive in both time and resources. (Bitton et al., 2022) Every new adversarial attack demands the creation of an appropriate dataset and a potentially lengthy and expensive retraining and deployment

process. The ideal solution is likely a middle ground between adversarial training and preprocessing.

8.9 Further Research

There are many interesting questions still left to explore and many ways in which one could expand and improve upon my thesis. Below are five topics that, had they not been completely out of scope for this project, I would have liked to investigate further.

- An obvious next step is to expand the scope of this thesis and test more models, more datasets and more attacks. How do models perform against word- and sentence-level attacks?
- How can models be made robust against adversarial attacks? What are the best mitigation tactics?
- The attacks in this thesis were generated algorithmically. How do models perform against adversarial attacks created by people?
- My thesis focused on German, and I briefly touched on how the German and English model differed. Are low-resource language models more fragile against adversarial attacks? How do different languages compare against each other?
- In my follow-up experiments I tested the influence of severity on an adversarial attack's success. My experiment consisted only of a single example. It would be interesting to further expand the scope and repeat the experiment with a statistically significant number of examples.

9 Conclusion

The aim of this thesis was to test German hate speech detection systems against character-level adversarial attacks and see what type of attacks they were most vulnerable against. Initially, I expected hate speech detection models to have improved significantly since Gröndahl et al. (2018) conducted their experiments. While models have improved, they are still strongly affected even by very simple attacks.

I theorized on the reasons why some of the attacks were more successful than others and behaved differently for the two models, but was ultimately restricted in my analysis due to my limited knowledge and understanding of Perspective's and HateMonitors' model architectures. However, while the models' susceptibility to certain attacks varied slightly, they still largely shared weaknesses. The different datasets had no significant impact on the effectiveness of attacks.

Out of the tested attacks, typos and random whitespaces were the least effective. They also happened to be the least extreme ones and those that mimicked accidental user behaviour. Removing whitespaces and adding "Liebe" proved very effective, replicating the results by Gröndahl et al. (2018) almost exactly. Extreme whitespaces and leetspeak were very successful on both models. I also found that adversarial attacks work best if they affect the entirety of a sentence instead of just certain words. The most effective adversarial attacks were simple but severe.

References

Alsmadi, Izzat et al. (Oct. 2021). "Adversarial Attacks and Defenses for Social Network Text Processing Applications: Techniques, Challenges and Future Research Directions". In: *arXiv preprint* abs/2110.13980. URL: https://arxiv.org/abs/2110.13980?context=cs.

Assenmacher, Dennis et al. (Aug. 2021). *RP-Mod & RP-Crowd: Moderator- and Crowd-Annotated German News Comment Datasets*. Version v1. The research leading to these results received funding from the federal state of North Rhine- Westphalia and the European Regional Development Fund (EFRE.NRW 2014-2020), Project: MODERAT! (No. CM-2-2-036a). Zenodo. URL: https://doi.org/10.5281/zenodo.5242916.

Bitton, Joanna, Maya Pavlova, and Ivan Evtimov (July 2022). "Adversarial Text Normalization". In: *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies: Industry Track*. Hybrid: Seattle, Washington + Online: Association for Computational Linguistics, pp. 268–279. DOI: 10.18653/v1/2022.naacl-industry.30. URL: https://aclanthology.org/2022.naacl-industry.30.

counterhate.com (Nov. 2022). *Fact check: Musk's claim about a fall in hate speech doesn't stand up to scrutiny*. https://counterhate.com/blog/fact-check-musks-claim-about-a-fall-in-hate-speech-doesnt-stand-up-to-scrutiny/. Accessed: 2022-11-17.

Demus, Christoph et al. (July 2022). "DeTox: A Comprehensive Dataset for German Offensive Language and Conversation Analysis". In: *Proceedings of the Sixth Workshop on Online Abuse and Harms (WOAH)*. Seattle, Washington (Hybrid): Association for Computational Linguistics, pp. 143–153. URL: https://aclanthology.org/2022.woah-1.14.

Espinosa Anke, Luis et al. (Jan. 2019). "Hate Speech Detection: A Solved Problem? The Challenging Case of Long Tail on Twitter". In: *Semant. Web* 10.5, pp. 925–945. ISSN: 1570-0844. URL: https://doi.org/10.3233/SW-180338.

Fortuna, Paula and Sérgio Nunes (July 2018). "A Survey on Automatic Detection of Hate Speech in Text". In: vol. 51. 4. New York, NY, USA: Association for Computing Machinery. URL: https://doi.org/10.1145/3232676.

Goodfellow, Ian J., Jonathon Shlens, and Christian Szegedy (2015). "Explaining and Harnessing Adversarial Examples". In: *International Conference on Learning Representations*. URL: https://arxiv.org/abs/1412.6572.

Gröndahl, Tommi et al. (2018). "All You Need is "Love": Evading Hate Speech Detection". In: *Proceedings of the 11th ACM Workshop on Artificial Intelligence and Security*. AISec '18. Toronto, Canada: Association for Computing Machinery, pp. 2–12. ISBN: 9781450360043. URL: https://doi.org/10.1145/3270101.3270103.

Guterres, Antonio (2019). "United Nations Strategy and Plan of Action on Hate Speech". In: URL: https://www.un.org/en/genocideprevention/documents/advising-and-mobilizing/Action_plan_on_hate_speech_EN.pdf.

Harbecke, David et al. (May 2022). "Why only Micro-F1? Class Weighting of Measures for Relation Classification". In: *Proceedings of NLP Power! The First Workshop on Efficient Benchmarking in NLP*. Dublin, Ireland: Association for Computational Linguistics, pp. 32–41. URL: https://aclanthology.org/2022.nlppower-1.4.

Kumar, Dhruv, Robin Cohen, and Lukasz Golab (June 2019). "Online abuse detection: the value of preprocessing and neural attention models". In: *Proceedings of the Tenth Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*. Minneapolis, USA: Association for Computational Linguistics, pp. 16–24. URL: https://aclanthology.org/W19-1303.

Lees, Alyssa et al. (2022). "A New Generation of Perspective API: Efficient Multilingual Character-level Transformers". In: *KDD '22: Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. New York, NY, USA: Association for Computing Machinery. URL: https://arxiv.org/abs/2202.11176.

Mandl, Thomas et al. (2019). "Overview of the HASOC Track at FIRE 2019: Hate Speech and Offensive Content Identification in Indo-European Languages". In: *Proceedings of the 11th Forum for Information Retrieval Evaluation*. FIRE '19. Kolkata, India: Association for Computing Machinery, pp. 14–17. ISBN: 9781450377508. URL: https://doi.org/10.1145/3368567.3368584.

Perspective (2021). https://perspectiveapi.com/. Accessed: 2022-11-5.

Rieder, Bernhard and Yarden Skop (2021). "The fabrics of machine moderation: Studying the technical, normative, and organizational structure of Perspective API". In: *Big Data & Society* 8.2, p. 20539517211046181. URL: https://doi.org/10.1177/20539517211046181.

Röttger, Paul et al. (2022). "Multilingual HateCheck: Functional Tests for Multilingual Hate Speech Detection Models". In: *Proceedings of the Sixth Workshop on Online Abuse and Harms (WOAH)*. Seattle, Washington (Hybrid): arXiv, pp. 154–169. URL: https://arxiv.org/abs/2206.09917.

Saha, Punyajoy et al. (Dec. 2019). "HateMonitors: Language Agnostic Abuse Detection in Social Media". In: *Forum for Information Retrieval Evaluation*. Kolkata, India. URL: https://arxiv.org/abs/1909.12642.

Sennrich, Rico, Barry Haddow, and Alexandra Birch (Aug. 2016). "Neural Machine Translation of Rare Words with Subword Units". In: *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Berlin, Germany: Association for Computational Linguistics, pp. 1715–1725. DOI: 10.18653/v1/P16-1162. URL: https://aclanthology.org/P16-1162.

Wagner, Kristina and Cassandra Bumann (2020). "Challenges in Annotating a Corpus for Automatic Hate Speech Detection". In: *BOBCATSSS 2020 Paris - Information Management, Fake News and Disinformation*. Paris, France. URL: https://bobcatsss2020.sciencesconf.org/data/pages/BoA_final.pdf.

Wieling, Martijn, Josine Rawee, and Gertjan van Noord (Dec. 2018). "Reproducibility in Computational Linguistics: Are We Willing to Share?" In: *Computational Linguistics* 44.4, pp. 641–649. ISSN: 0891-2017. eprint: https://direct.mit.edu/coli/article-pdf/44/4/641/1809901/coli_a_00330.pdf. URL: https://doi.org/10.1162/coli%5C_a%5C_00330.