Efficient Transformers for Hate Speech Detection

Jesse Parvess
University of Pretoria
Department of Computer Science
Lynnwood Rd and Roper St Hatfield Pretoria 0028 Republic of South Africa
u15091644@tuks.co.za
https://github.com/15091644/COS-802

Abstract

Pivotal to automatic hate speech detection in online settings is the ability to separate it from instances of offensive language. Here, the contextual setting is important as the simple detection of certain terms associated with hate speech results in low precision, with many offensive phrases classified as hate speech. Recent work in masked language modeling using transformers potentially mitigates this issue. The bidirectional attention mechanism used during pretraining allows for these language models to be contextually aware. Various authors have investigated improving detection by fine-tuning BERT and RoBERTa in a hate speech multi-class classification setting. They have shown that these methods produce competitive results in separating hate speech from offensive language. However, since most instances of hate speech occurs in online settings there are size constraints of the models that can be used in deployment. Various contributions have looked to make these transformers more efficient so that they may be used in lower resource environments. However, none of these contributions have looked to investigate these improved architectures in hate speech detection. Therefore, various efficient transformer architectures are investigated in a hate speech classification task. Namely the more efficient architectures of DistilBERT, SqueezeBERT, MobileBERT, ALBERT are fine-tuned for hate speech detection and compared to BERT and RoBERTa in terms of efficiency and classification. It was found that DistilBERT, with approximately half the model size to BERT, obtains slightly better results compared to BERT.

Keywords: Transformers, Hate Speech Detection, Efficiency

1. Introduction

Social media has enabled the generation of massive amounts of text that represents people's opinions and views. This has also provided the opportunity for people and communities to openly engage in socially condemnable activity, behind a veil of anonymity. Therefore, the advent of social media has seen the rise of entire online communities, who actively engage in hate speech (Fekete, 2018). It is nearly impossible to manually curate these texts, therefore automatic detection of hate speech is required.

A key issue to automatic hate speech detection is the ability to separate it from offensive language (Davidson et al., 2017). Separating the two types of speech requires contextual understanding of the text in question. This is especially problematic as social media contains a high prevalence of "curse words" that conflate the distinction (Wang et al., 2014). Often models that are trained using lexical detection methods (that search for phrases or terms) fail to distinguish the two. Namely text containing these certain terms are classified as hate speech, but their context warrants classification as offensive language (Davidson et al., 2017). This issue rules out models like bag-of-words that use keywords as strong indicators in classification.

To improve detection, Mathew et al., (2020), Wullach et al., (2020) and Alonso et al., (2019) use deep learning approaches to cater for more nuanced separation. Particularly, they investigate using transformer models for their ability to cater for contextual settings. (Devlin et al., 2018). BERT (bidirectional encoder representations), uses masked language modeling in pretraining to fill the most probable word in a text, based on its context, these are the words left and right of the masked entity (Devlin et al., 2018). It is believed that this ability allows for BERT (and its likes) to be uniquely appropriate for distinguishing the difference between offensive language and hate speech. The authors present their highest performance on these types of models. However, they only present results for BERT and RoBERTa.

Despite improved results, BERT has millions of parameters that makes it inefficient to use these language models in high production environments (Sun et al., 2020). Therefore, there have been investigations to make BERT's architecture more efficient. Liu et al., (2019) optimized BERT's via longer pre-training, removing next sentence prediction, trained on longer sentences and changed the masking pattern used by Devlin et al., (2018). Although, this does not produce size reduction, they introduce RoBERTa, which still represents a first step in optimizing BERT's existing architecture. To reduce the size of BERT, Sanh et al., (2019) , presented DistilBERT and used knowledge distillation presented by Hinton et al., (2015) to compress BERT's size while maintaining task agnosticism and performance. To improve this, Sun et al., (2020) also used distillation to create MobileBERT, but via specially designed teacher and student networks that are equipped with architectural bottlenecks to reduce parameters. Furthermore, Iandola et al., (2020) use grouped convolutions in the self-attention layers to reduce BERT's size and computational cost in SqueezeBERT. Additionally, Lan et al., (2019) present ALBERT, that uses factorized embedding parameterization and cross-layer parameter sharing for parameter reduction for a more efficient architecture.

In this study we propose investigating these more efficient architectures by comparing them to BERT in a hate classification task. Each model is fine-tuned on the same set of training and validation examples, with the same number of training steps and batch size. Additionally, each is scored on the same set of test data. Most of these models were pretrained on data that uses formal writing (BookCorpus and English Wikipedia data sets). Since the hate speech data used for fine-tuning consists of Tweets, understanding the effect of informal language representation within

the models is required. Therefore, BERTweet was used as a benchmark to ascertain the effect of using a language model pretrained on tweets only.

For each model investigated, pre-trained parameters provided by the Hugging Face's Transformers library was used (Wolf et al., 2019).

The following investigation looks to answer the following questions:

1. Why is hate speech detection difficult, even for human annotators?
2. Can more efficient transformer architectures perform as well as larger capacity models in hate speech detection?
3. What effect does the type of data used in pre-training have on hate speech detection in online settings?
4. What type of words/phrases do these transformers models struggle with, when detecting hate speech?

2. Literature Survey

2.1 Hate Speech Detection

Davidson et al., (2017) details the issues of automatically differentiating hate speech and offensive language. For instance, they state that some African Americans use the term "n*gga" as a colloquialism and words like "h*e" and "b*tch" are often quoted in rap lyrics by online users. The context in which these words lie does not necessarily warrant classification as hate speech and at most could be offensive. Furthermore, hate speech can be subtle with no explicit words used to express it. For instance, the phrase "No [insert group of people] deserves to live" contains no explicit accusatory language but can constitute hate speech. Therefore, the designation between the two types of language is subtle and a difficult task due to the context.

This task is further compounded by the nature of online language. Text from social media often lacks proper grammar and contains paralinguistic elements like URLs, emoticons, hashtags and emojis to convey a message (Alonso et al., 2019).

To make hate speech detection on social media more robust, Mathew et al., (2020) curate a data set composed from different sources (Twitter and Gab). Their data set (HateXplain) uses three classes (hate, offensive and neither) like that used by Davidson et al., (2017). Additionally, the authors fit several deep learning models, that represent an improvement to the SVM, and logistic regression approach used by Davidson et al., (2017). Their highest performance comes from BERT.

Alonso et al., (2019) builds on this by using an ensemble of transformers for hate speech detection. They use 5-fold ensemble training using RoBERTa. The average of the ensemble's output is used for the final prediction. They attain state-of-the-art performance on the HASOC benchmark.

Additionally, (Zhou et al., 2020) use the diversity of different deep learning models to create a voting ensemble for hate speech detection. They use several transformer models like ELMo and BERT along with CNNs to improve the diversity in the voting ensemble.

The above represents strides towards better detection of hate speech using the power of models that are contextually aware (BERT and Roberta). However, these models contain millions of parameters and are inefficient if placed in production.

2.2 Transformer Models the Road to Greater Efficiency

2.2.1 BERT – Introduced 2018

BERT was introduced by Devlin et al., (2018). It represents two self-supervised learning tasks. The first involves masked language modeling (MLM) and the second next sequence prediction (NSP). BERT only consists of an encoder architecture that uses bidirectionality to ascertain context. During pretraining 15% of the words in a sequence are randomly masked. The model then attempts to predict the words that were masked by using the sequences to the left and right of the masked words. Additionally, pretraining involves a binary NSP task. Here sentence pairs are predicted to be a pair of two following sentences or not. The above paradigm attempts to have a language representation that is contextually aware.

BERT-base consists of 110 million parameters. Additionally, it was pretrained on BookCorpus consisting of 800 million words and English Wikipedia consisting of 2500 million words.

2.2.2 RoBERTa – Introduced in 2019

RoBERTa was introduced by Liu et al., (2019). The authors postulate that BERT was significantly undertrained and that better hyperparameter choices could be used to significantly improve BERT's performance.

To generate further improvements the authors trained BERT for a longer period, with larger batches, more data, on longer sequences and with no next sequence prediction. RoBERTa produced state-of-the-art results when compared to BERT. Therefore, the authors optimized the existing architecture of BERT.

Like BERT, the authors trained RoBERTa on the BookCorpus and the English Wikipedia data sets. Additionally, they use the CC-NEWS data set consisting of 63 million English news articles. and the OPENWEBTEXT data set from Reddit.

Additionally, RoBERTa-base consists of 123 million parameters and is slightly larger than BERT-base but has a more optimized language representation because of the stated pretraining changes.

2.2.3 DistilBERT – Introduced in 2019

DistilBERT was introduced by Sanh et al., (2019). The authors looked to generate a task agnostic model that was smaller and can operate under constrained computational budgets during training and inference. The authors placed specific emphasis on the need for smaller models to reduce the environmental footprint during the training of large language models, which is computationally intensive and power hungry (Schwartz et al., 2019; Strubell et al., 2019).

To generate smaller models, they leverage knowledge distillation (Hinton et al., 2015) during pretraining to reduce the size of BERT by 40%. Here, a smaller student model is trained to reproduce the

outputs of a larger model, while maintaining general purpose task agnosticism.

DistilBERT has the same general architecture as BERT, except its token-type embeddings and pooler layers are removed and the number of layers were halved. Additionally, the optimizations introduced in RoBERTa (Liu et al., 2019) were used to improve training. Namely, larger batch sizes were used, and next sequence prediction was removed.

2.2.4 ALBERT – Introduced in 2019

ALBERT was introduced by Lan et al., (2019). The authors looked to introduce parameter reduction techniques to reduce the size of BERT.

Firstly, they factorized the large vocabulary embedding matrix into two small matrixes. This separates the size of the hidden layers from the size of the vocabulary embedding. This allows for the size of the hidden layer to grow without increasing the parameter size of the vocabulary embeddings.

Secondly, they use cross-layer parameter sharing. This prevents the number of parameters from growing as the network becomes deeper. It also was shown that parameter sharing can stabilize the network's parameters.

These techniques reduce ALBERT's parameters by 18 times compared to BERT-large and speeds up training. Furthermore, these techniques can be seen as a form of regularization that prevents overfitting due to high model capacity.

Additionally, to improve performance, BERT's next sequence prediction was removed, and sentence order prediction (SOP) was used to focus on inter-sentence coherence. Here SOP is like BERT's NSP (using two consecutive segments of text from the same document to denote the positive class). However, the task is made more difficult by using the same two consecutive texts and swopping the order to denote the negative class. Whereas, in BERT's NSP a random sentence is used to the base sentence to denote the negative class. This trains the model to learn finer-grain discourse-level attributes.

Like BERT, ALBERT was trained using the BookCorpus and English Wikipedia data sets.

2.2.5 MobileBERT – Introduced in 2020

MobileBERT was introduced by Sun et al., (2020). The authors looked to introduce a model that compresses and accelerates BERT for mobile applications.

To train MobileBERT a teacher model is first trained. Here the teacher model has an inverted bottleneck incorporated in BERT-large. Following this, knowledge transfer occurs to the smaller MobileBERT model using distillation like that used in DistilBERT. MobileBERT is designed to be as deep as BERT but thinner as the size of its hidden layers are reduced laterally. The resulting model is 4.3x smaller and 5.5.x faster than BERT-base.

Like BERT, MobileBERT was trained using the BookCorpus and the English Wikipedia data set.

2.2.6 SqueezeBERT – Introduced in 2020

SqueezeBERT was introduced by Iandola et al., (2020). They looked to improve the efficiency of BERT by using principles found in computer vision. Namely, they replaced several operations in the self-attention layers with grouped convolutions.

They argue that convolutions are better operations because they are flexible and well-optimized in in software. Furthermore, they use grouped convolutions, proposed by Krizhevsky et al., (2012), to speed up convolution computation. Additionally, upon analyzing the latency in certain layers within BERT, they found that the position-wise fully connected (PFC) layers found in the self-attention and feed-forward modules contributes the majority to the model's latency (88% of compute time). Hence, it is these layers that are replaced with grouped convolutions.

Furthermore, they observe that the PFC operation is equivalent to a convolution with a kernel size of k=1. This implies that transformer models like GPT and BERT can replace PFC operations with grouped convolutions without changing any of the model's numerical properties.

Like with ALBERT, the authors use masked language modelling and sentence order prediction as pretraining tasks. SqueezeBERT is also pre-trained on the BookCorpus and English Wikipedia data sets.

2.3 BERTweet for Text on Social Media

From the above discussion, it is evident that many of the transformer models considered were pretrained on the BookCorpus and English Wikipedia datasets, which consists of formal language.

This presents a potential problem as automatic hate speech detection occurs online on social media settings. Here formal language is not used and often lacks proper grammar and contains paralinguistic elements like URLs, emoticons, hashtags and emojis (Alonso et al., 2019). Furthermore, these informal texts also change rapidly in style and content as new memes and trends emerge (Carrier, 2019). Specifically, Tweets differ because they are restricted in length and use irregular vocabulary like punctuations and abbreviations to convey a message (Eisenstein, 2013)

Therefore, the effect of using pretraining data that is more aligned to online settings is required. Specifically, since the data used for hate speech detection consists of Tweets, a model that was pretrained on Tweets is required.

Quoc Nguyen et al., (n.d.) present BERTweet, which essentially contains the same architecture as BERT but uses Tweet texts in pre-training. Their hope is to specialize BERT's pretraining data for tasks using BERT on Twitter.

Here, 80GB of 850 million English Tweets were used to pre-train a BERT model. Each Tweet consisted of at least 10 and at most 64 tokens. They also normalized the text by converting mentions to "@USER" and web/url links to "HTTPURL". Additionally, retweets are filtered out. Furthermore, Tweets form 2012 – 2019 were used with the addition of COVID-19 related tweets from 2020-01 – 2020-03 were used to make this language representation reasonably up to date.

BERTweet was found to outperform RoBERTa-base and XLM-base, thus outperforming these model's state-of-the-art results on three Tweet-based NLP tasks.

3. Methodology

3.1 Data Set

Data provided by Davidson et al., (2017) was used to fine-tune each of the transformer models. The authors begun by extracting tweets that contained hate speech lexicon (phrases and words identified as hate speech that were compiled by Hatebase.org.

CrowdFlower was then used for manual annotation of 25 000 extracted tweets. They were asked to label the text based on the context in which the tweet was used and not simply on the phrases or terms present within the tweet. Each annotator must vote as to whether a tweet is considered hate speech, offensive language or neither.

The authors then further filtered the data to only include annotations done by more than 2 individuals. This was done to reduce bias. This reduced the size of the data to 24783 instances. The majority vote was then used to designate the label for each tweet.

The data set has the following fields and explanation:

Table 1: The data used to fine-tune the transformer models.

Field	Explanation
count	Total number of annotators per tweet
hate_speech	Total number of votes for hate speech
offensive_language	Total number of votes for offensive language
neither	Total number of votes for neither
class	The class designation for the tweet
tweet	The text containing the tweet

The data was then split into a training set (17 349 Tweets), a validation set (3717 Tweets) and a test set (3717 tweets). These data sets were kept constant across all models during training and evaluation.

It is important to note, that the data set was imbalanced with 77% of labels being offensive language, 15% labels were neither, and 8% were classified as hate speech.

3.2 Preprocessing

To normalize the tweets, hashtags and mentions were processed to state “HASHTAGHERE” and “MENTIONHERE”. Additionally, ampersands were removed, and “RT” was replaced with “RETWEET” to designate a retweet. Furthermore, common English contractions were expanded and emojis were removed from the text.

3.3 Algorithms

BERT, RoBERTa, DistilBERT, ALBERT, MobileBERT, SqueezeBERT, and BERTweet were fine-tuned on the same training and validation training.

Each model was trained for two epochs, with batches of 16 tweets. This represented a total of 2170 training steps.

Once the model was fine-tuned to multi-class hate speech detection the test data was used to score each model. Here, a confusion matrix was used to understand the model’s biases. Additionally, the models were scored using precision, recall, F1 and accuracy. Here, the highest scoring model checkpoint was used in each case.

Furthermore, each model’s size and evaluation time was registered to understand what efficiency gains could be found per model architecture.

4. Results

4.1 Hate Speech Detection a Difficult Task

To investigate the difficulty of the task, the distribution of the annotator’s labelling was investigated.

4.1.1 Neither Class

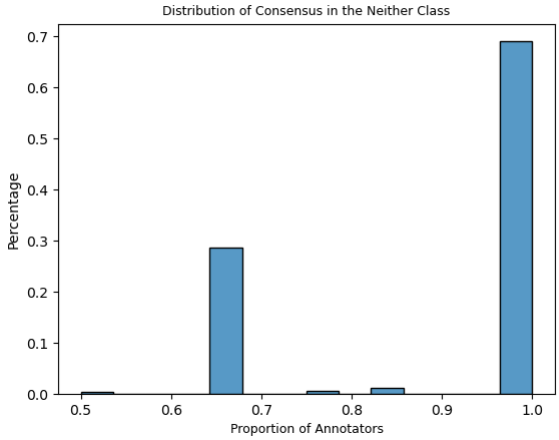

Figure 1: Distribution of consensus of the neither class.

Figure 1 indicates that of the Tweets labelled as neither, 70% had a 100% consensus on the Tweet’s nature. This indicates the majority of annotators agreed to the Tweet’s classification as neither, and that there was little ambiguity in its classification.

4.1.2 Offensive Language

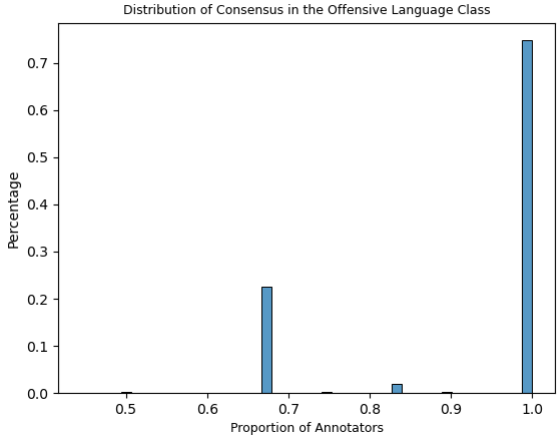

Figure 2: Distribution of consensus of the offensive speech class

Figure 2 indicates that of the Tweets labelled as offensive, 75% had a 100% consensus of the Tweet's nature. This indicates the majority of annotators agreed to the Tweet's classification as offensive, and that there was little ambiguity in its classification.

4.1.3 Hate Speech

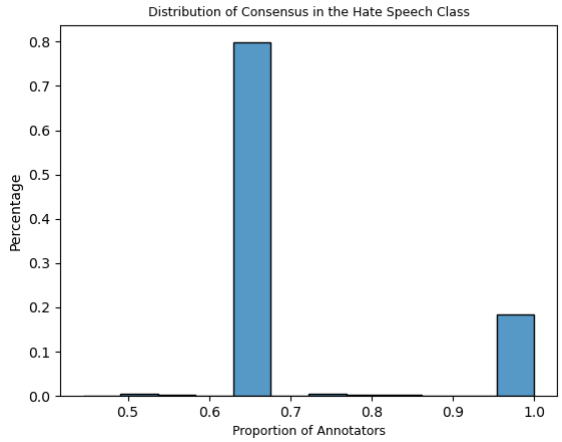

Figure 3: Distribution of consensus of the hate speech class

Figure 3 indicates that of the Tweets labelled as hateful, 20% had a 100% consensus of the Tweet's nature. This a reversion on the trend observed in figures 1 and 2. This indicates that the majority (80%) of annotators did not have full agreement (67% consensus) on the nature of the Tweets labelled as hate speech.

This demonstrates that the task of hate speech detection is contextually difficult even for human annotators. To further illustrate this, the examples of Tweets with lowest hate speech consensus are shown:

Table 2: Tweets with Low Annotator Consensus

tweet	hate speech consensus
Drakes new shoes that will be released by Nike/Jordan.... Yes, there's glitter on the shoes ....dudes a fag	0.5
Apaches' 30 MM rounds turn Jihadis into tomato soup	0.5
Getting hit off a cunt thats about 6 foot sideways and a cunt that was in the army lol, i dini care, yous are fuckin laughable	0.5
Shawty said you not gone understand america unless you went to a PWI school, crackers already brainwashed her smh lol	0.44
I don't support that ya idiot. And if I'm a teabagger you're being teabagged.	0.5
Mine too This ma niggah	0.5
Uncle Tom Hypocrite!	0.5
When it rains it's good "go to the mall weather" all the coons be at home	0.5

Table 2 indicates that the Tweets classified as hate speech, with the lowest annotator consensus are ambiguous. Some do not have offensive phrases while others use hateful language in a contextual setting that makes it difficult to differentiate.

The difficulty of this task warrants the use of contextually aware models.

4.2 Confusion Matrix

To visualize results across model architectures, the models with the highest correctly classified instances are shaded in green. Additionally, the models with the highest misclassified examples are shaded in red.

Table 3: Confusion matrices for the various models tested

Confusion Matrix for BERT

Class	Hate Speech	Offensive Language	Neither
Hate Speech	90	101	22
Offensive Language	81	2757	67
Neither	10	38	552

Confusion Matrix for RoBERTa

Class	Hate Speech	Offensive Language	Neither
Hate Speech	65	122	26
Offensive Language	70	2770	65
Neither	17	38	545

Confusion Matrix for DistilBERT

Class	Hate Speech	Offensive Language	Neither
Hate Speech	90	104	19
Offensive Language	75	2754	76
Neither	15	28	557

Confusion Matrix for ALBERT

Class	Hate Speech	Offensive Language	Neither
Hate Speech	38	132	43
Offensive Language	42	2755	108
Neither	5	29	566

Confusion Matrix for MobileBERT

Class	Hate Speech	Offensive Language	Neither
Hate Speech	76	108	29
Offensive Language	74	2756	75
Neither	8	37	555

Confusion Matrix for SqueezeBERT

Class	Hate Speech	Offensive Language	Neither
Hate Speech	86	105	22
Offensive Language	84	2748	73
Neither	8	47	545

Confusion Matrix for BERTweet

Class	Hate Speech	Offensive Language	Neither
Hate Speech	78	113	22
Offensive Language	65	2776	64
Neither	14	28	558

Table 3 indicates that all models are biased towards classifying hate speech as offensive language. This could be due to the similarity between text types and the class imbalance within the training data set.

DistilBERT and BERT perform equally in correctly classifying hate speech. However, BERTweet performs the best out of all the models in classifying offensive language and neither, indicating improved language representation from the Tweet pretraining data.

ALBERT performs poorly in classifying hate speech and struggles to distinguish offensive language from neither.

Additionally, SqueezeBERT conflates offensive language with hate speech and neither with offensive language.

4.3 Efficiency

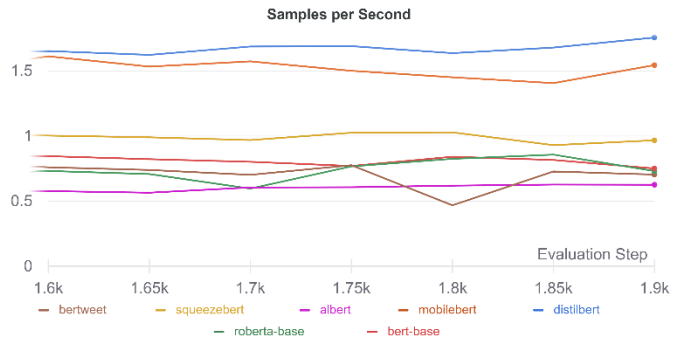

Figure 4: Samples per second at each evaluation step per model.

Figure 1 indicates that the most efficient model for this task is DistilBERT, with MobileBERT closely following. They both have similar inference times of approximately 1.5 samples per second. Furthermore, SqueezeBERT is the third most efficient model, with approximately 1 sample evaluated per second. The above indicates that the efficiency measures used in these models have made them faster.

BERT, RoBERTa and BERTweet have similar inference times. This is because each model has similar architecture sizes and are only optimized in terms of data (BERTweet) or hyperparameter considerations (RoBERTa).

4.4 Evaluation

Each model is evaluated across several metrics, such as its size, F1 score, precision and recall for each class. Additionally, the macro average is chosen for to summarize each metric. This was done because the classes in the test set were imbalanced, and a global average would mask poor performance on underrepresented classes.

Models that scored the best were bolded in green and those that performed the worse were bolded in red.

If a set of models had the same score the scores were bolded for all models.

Table 10: Evaluation of each model

Model	Label	Size (GB)	F1	Recall	Precision
BERT	Global	0.43	**0.77**	0.76	**0.77**
	Hate	-	**0.46**	**0.42**	**0.5**
	Offensive	-	**0.95**	0.95	**0.95**
	Neither	-	0.89	0.92	**0.86**
BERTweet	Global	0.53	0.76	0.75	**0.77**
	Hate	-	0.42	0.37	**0.5**
	Offensive	-	**0.95**	**0.96**	**0.95**
	Neither	-	**0.9**	0.93	**0.87**
RoBERTa	Global	0.49	0.73	0.72	0.74
	Hate	-	0.36	0.31	**0.43**
	Offensive	-	**0.95**	0.95	**0.95**
	Neither	-	0.88	0.91	0.86
DistilBERT	Global	0.26	**0.77**	**0.77**	**0.77**
	Hate	-	**0.46**	**0.42**	**0.5**
	Offensive	-	**0.95**	0.95	**0.95**
	Neither	-	0.89	0.93	0.85
ALBERT	Global	**0.045**	**0.69**	**0.69**	**0.73**
	Hate	-	**0.26**	**0.18**	0.45
	Offensive	-	**0.95**	0.95	**0.94**
	Neither	-	**0.86**	**0.94**	**0.79**
MobileBERT	Global	0.1	0.75	0.74	0.76
	Hate	-	0.41	0.36	0.48
	Offensive	-	**0.95**	0.95	**0.95**
	Neither	-	0.88	0.93	0.84
SqueezeBERT	Global	0.2	0.76	0.75	0.76
	Hate	-	0.44	0.4	0.48
	Offensive	-	**0.95**	0.95	**0.95**
	Neither	-	0.88	0.91	0.85

Table 10 indicates that BERTweet outperforms in identifying offensive language and neither. It is speculated that the reason for this is because this model was pretrained on Tweet data and thus has a representation of informal writing on Twitter.

Furthermore, DistilBERT is the best performing model, and marginally outperforms BERT in hate speech detection. This indicates that with half the memory, DistilBERT can perform as well as BERT for this task.

MobileBERT and SqueezeBERT do not outperform the other models. It is speculated that the reduction in model capacity could be the reason for lower results. Comparing between the two models, we find that SqueezeBERT outperforms MobileBERT. This could be because the convolutions in SqueezeBERT reduce

parameterization but do not numerically change the model from BERT's operations. Therefore, SqueezeBERT is closer in structure to the best performing models (BERT, DistilBERT, BERTweet) and thus better compared to MobileBERT's different architecture.

ALBERT performed the worst compared to all models. The change in the pretraining regime, namely the use of parameter sharing as a form of regularization, may have resulted in a model that lacked the capacity to generalize well for this task (underfitting).

4.5 Consistent Error Analysis

To understand the types of speech that the models struggled to classify, the examples for which the ensemble of models made the same collective mistakes are extracted. It is assumed that if all the models made the same error, then there is something inherent within the example that makes it difficult to classify.

To detect if the model was anchored on particular words, a count of the vocabulary in the misclassified examples were performed. If multiples of the same word were found across the misclassified Tweets, then it is assumed that the model is anchored on those terms, with these terms potentially being the source of the error.

4.5.1 Labelled ad Hate Speech, Classified as Offensive

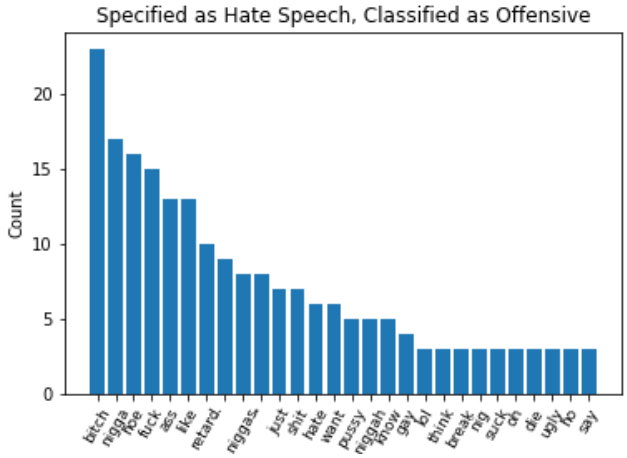

Figure 5: Word counts for Tweets of hate speech, where the models all predicted offensive language.

Figure 5 indicates that the models were anchored on curse words like 'b*tch', 'n*gga', 'f*ck' and 'sh*t'. These words are probably common in most of the offensive Tweets and thus make it difficult for the models to differentiate between hate speech and offensive language in these instances.

4.5.2 Labelled as Hate Speech, Classified as Neither

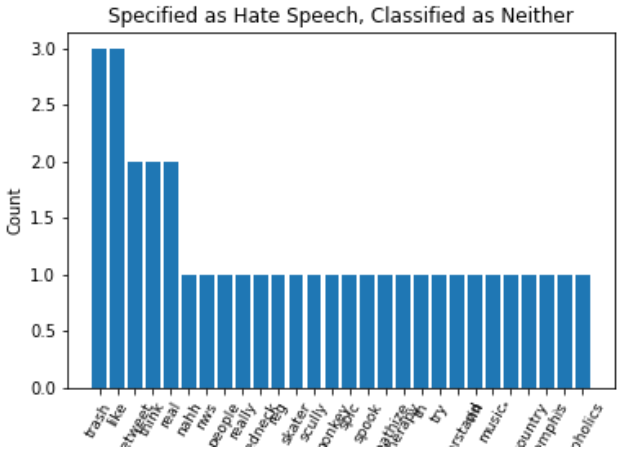

Figure 6: Word counts for Tweets of hate speech where the models all predicted neither.

Figure 6 indicates words like 'trash', 'redneck', 'scully' and 'monkey' were conflated as neither, due to their ability to be used colloquially in informal language in a manner that is not hate speech. However, these instances used the terms in a hateful manner when placed in context.

4.5.3 Labelled as Offensive, Classified as Hate Speech

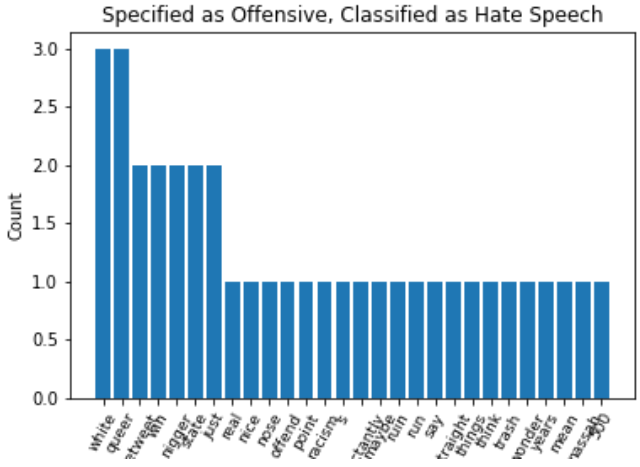

Figure 7: Word counts for Tweets of offensive language, where the models all predicted hate speech.

Figure 7 indicates that words used in Tweets like 'white', 'queer' and 'n*gger' potentially caused misclassification as hate speech due to the prominence of these terms being used in this type of speech.

4.5.4 Labeled as Offensive, Classified as Neither

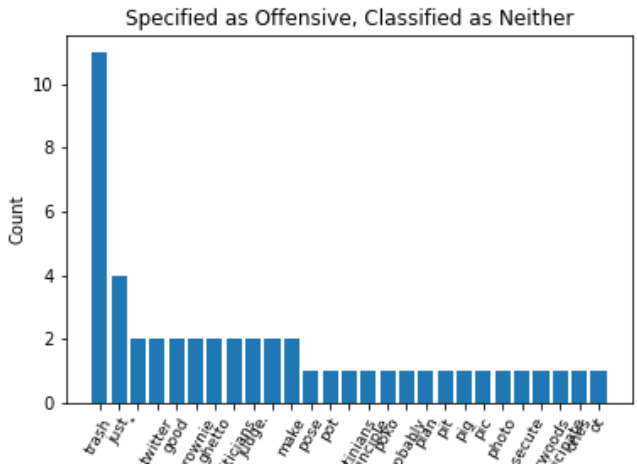

Figure 8: Word counts for Tweets of offensive language where the models all predicted neither.

Figure 8 indicates that the models were anchored on the terms 'trash', 'brownie' and 'ghetto'. These words can be used in a neutral context. In the context of the tweets, these terms were used in an offensive manner.

4.5.5 Neither Labelled as Offensive Language

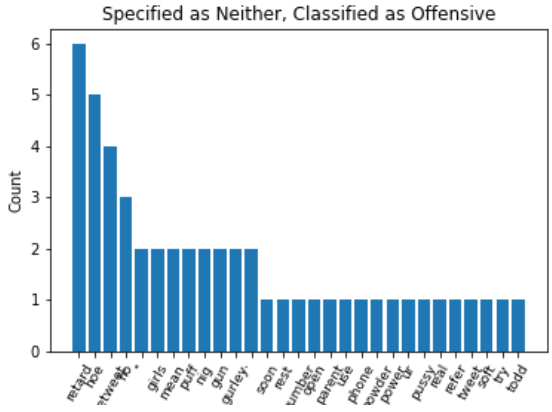

Figure 9: Word counts for Tweets of neither where the models all predicted offensive language.

Figure 9 indicates that the models were anchored on the terms 'retard', 'h*e', 'mean' and 'puff'. These terms can be used in a colloquial manner. However, their prevalence in offensive language biased the models to classify tweets with these terms as offensive.

4.5.6 Neither Labelled as Hate Speech

There were no instances where all the models misclassified the neutral Tweets as hate speech. This indicates that the dissimilarity between these types of speech makes this task easy to separate.

4.5.7 Discussion

The above discussion indicates that these models are heavily influenced by certain terms within a Tweet. This biases the models and is a major source of error. This could be due to the small amount of training data available for the models during fine-tuning.

Additionally, the class imbalance within the training set also biases these models towards certain terms seen in the majority class (offensive language).

This indicates the importance of curating a better hate detection data set that contains many examples of the target classes.

5. Future Work and Conclusion

5.1.1 Future Work

This investigation indicated that the pre-training data used has a significant impact on the fine-tuning classification task. Therefore, an investigation as to whether pre-training smaller models, like DistilBERT, MobileBERT and SqueezeBERT, on Tweets exclusively might create performance gains in hate speech detection.

Furthermore, the above investigation showed that these models suffer with bias when there are few fine-tuning examples. Therefore, a larger hate speech data set must be curated, and training should occur with a balanced data set.

Additionally, the investigation showed that certain sub-tasks are easier than others (differentiating neither from offensive language). There may be performance gains if the problem is broken down into a series of binary tasks that first look to do high level speech differentiation and then use a specialized model on harder more nuanced tasks. For instance a model that differentiates neither (labelled 0) and hate speech with offensive language (labelled 1) as a first task specializes this model for this separation. Secondly an additional model that then separates hate speech from offensive language can then be tuned. This may be easier for a series of models compared to having a single model that looks to generalize in detection.

5.2 Conclusion

Hate speech detection is a difficult task that even human annotators struggle to do. Therefore, contextually aware transformer models are appropriate for this task. However, they are large and not practical for deployment. Smaller transformer architectures were investigated, and it was found that DistilBERT performers as well as BERT. Additionally, it was found that the pre-training data used is a significant factor for applications where the deployment environment is vastly different to the data used in pre-training (formal language versus informal language used on Twitter). Furthermore, looking at the examples where all the transformer

models made the same mistake, it was found that these models struggle with certain terms that can be used non-exclusively in different types of speech. Enriching the data set with these types of difficult examples may improve performance further and improve model detection.

6. Available Model

Core to Hugging Face's philosophy is the democratization of artificial intelligence, especially with regard to transform models.

Following in line with this philosophy a DistilBERT model trained on the same set of data is available in their hub: https://huggingface.co/JesseParvess/autonlp-hate_speech_detection-32207676

7. References

Alonso, P., Saini, R., & Kovács, G. (n.d.). *Hate Speech Detection using Transformer Ensembles on the HASOC dataset*. https://t.co/e2C48U3pss

Carrier, M. (2019). Because Internet: Understanding the new rules of language (a review). *Training Language and Culture*, *3*, 107–111. https://doi.org/10.29366/2019tlc.3.3.8

Davidson, T., Warmsley, D., Macy, M., & Weber, I. (2017). *Automated Hate Speech Detection and the Problem of Offensive Language*. www.aaai.org

Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2018). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *CoRR*, *abs/1810.04805*. http://arxiv.org/abs/1810.04805

Eisenstein, J. (2013). *What to do about bad language on the internet*. Association for Computational Linguistics.

Fekete, L. (2018). Alt-America: the rise of the radical Right in the age of Trump. *Race & Class*, *60*(1), 107–109. https://doi.org/10.1177/0306396818771235

Hinton, G., Vinyals, O., & Dean, J. (2015). *Distilling the Knowledge in a Neural Network*. http://arxiv.org/abs/1503.02531

Iandola, F. N., Shaw, A. E., Krishna, R., & Keutzer, K. (2020). SqueezeBERT: What can computer vision teach NLP about efficient neural networks? *CoRR*, *abs/2006.11316*. https://arxiv.org/abs/2006.11316

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (n.d.). *ImageNet Classification with Deep Convolutional Neural Networks*. http://code.google.com/p/cuda-convnet/

Lan, Z., Chen, M., Goodman, S., Gimpel, K., Sharma, P., & Soricut, R. (2019). ALBERT: A Lite BERT for Self-supervised Learning of Language Representations. *CoRR*, *abs/1909.11942*. http://arxiv.org/abs/1909.11942

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., & Stoyanov, V. (2019). RoBERTa: A Robustly Optimized BERT Pretraining Approach. *CoRR*, *abs/1907.11692*. http://arxiv.org/abs/1907.11692

Mathew, B., Saha, P., Yimam, S. M., Biemann, C., Goyal, P., & Mukherjee, A. (2020). *HateXplain: A Benchmark Dataset for Explainable Hate Speech Detection*. http://arxiv.org/abs/2012.10289

Quoc Nguyen, D., Vu, T., Tuan Nguyen, A., & Research, V. (n.d.). *BERTweet: A pre-trained language model for English Tweets*. https://pypi.org/project/emoji

Sanh, V., Debut, L., Chaumond, J., & Wolf, T. (2019). DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. *CoRR*, *abs/1910.01108*. http://arxiv.org/abs/1910.01108

Schwartz, R., Dodge, J., Smith, N. A., & Etzioni, O. (2019). *Green AI*. http://arxiv.org/abs/1907.10597

Strubell, E., Ganesh, A., & McCallum, A. (2019). *Energy and Policy Considerations for Deep Learning in NLP*. http://arxiv.org/abs/1906.02243

Sun, Z., Yu, H., Song, X., Liu, R., Yang, Y., & Zhou, D. (2020). MobileBERT: a Compact Task-Agnostic BERT for Resource-Limited Devices. *CoRR*, *abs/2004.02984*. https://arxiv.org/abs/2004.02984

Wang, W., Chen, L., Thirunarayan, K., & Sheth, A. P. (2014). Cursing in English on Twitter. *Proceedings of the 17th ACM Conference on Computer Supported Cooperative Work & Social Computing*, 415–425. https://doi.org/10.1145/2531602.2531734

Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M., Davison, J., Shleifer, S., von Platen, P., Ma, C., Jernite, Y., Plu, J., Xu, C., Scao, T. le, Gugger, S., … Rush, A. M. (2019). *HuggingFace's Transformers: State-of-the-art Natural Language Processing*. http://arxiv.org/abs/1910.03771

Wullach, T., Adler, A., & Minkov, E. (2020). *Towards Hate Speech Detection at Large via Deep Generative Modeling*. http://arxiv.org/abs/2005.06370

Zhou, Y., Yang, Y., Liu, H., Liu, X., & Savage, N. (2020). Deep Learning Based Fusion Approach for Hate Speech Detection. *IEEE Access*, *8*, 128923–128929. https://doi.org/10.1109/ACCESS.2020.3009244