Classifying and Automatically Neutralizing Hate Speech with Deep Learning Ensembles

Ali Hindy, Varuni Gupta, John Ngoi
Stanford University CS224n Custom Project

Abstract

Hate speech is one of the most prevalent forms of polarizing language on the planet. This form of human language degrades and disrespects others, yet it is often difficult to detect automatically due to difficulties in understanding language context and bias (oftentimes, directed towards African American dialogue). The invention of social media has amplified hate speech to a magnitude never seen before in human history. To address this issue, we leverage **deep ensemble learning techniques** to classify and automatically neutralize hate speech. By leveraging the Hugging Face Twitter Hate Speech dataset, our sentiment analysis model is an ensemble system that utilizes a BERT encoder to identify hate speech words and phrases. In addition, we contribute a two-fold pipeline that can detect hate speech given the training samples on a word-by-word basis using a classification model, then replace hateful words with more neutral words using a per-word seq2seq model to generate the neutral word. We ran and evaluated baseline models such as Random Forest, Logistic Regression, Decision Trees, SVC, XGBoost for the classification tasks, yet our HateEnsemble-finetune model outperformed all of them with an F1 score of **99.36%**. Human evaluation and our perplexity scores suggest that these data and models are a first step towards the automatic identification and replacement of hate speech in text.

Introduction

Hate speech is any kind of communication that attacks or uses pejorative language with reference to an individual or group's religion, ethnicity, nationality, race, or other identifying factor. Hate speech often has more implications than just pejorative verbal language, as it perpetuates intolerance and bigotry and it can potentially lead to violence. For example, the sentence "We want the Arabs out of France" contains hate speech, since there is a derogatory meaning involving wanting an entire group of people out of France due to their identity. The contribution of our paper is as follows:

- We introduced a **novel ensemble model** consisting of pretrained and finetuned BERT models using an averaged softmax function on our dataset.
- We created a **novel dataset** ensemble from various datasets to assist in the pretrained BERT models finetuning. The datasets contain text that are labeled as hate-speech and not-hate-speech.
- We have also done some **initial analysis** to come up with the first end-to-end pipeline for hate speech classification and neutralization, where we suggest edits on a word-to-word basis to replace hate speech with neutral language until the classification model does not recognize the sentence as hate speech

We also provide recommendations for future work in improving our pipeline and suggestions for researchers interested in deep learning ensembles. In the following sections, we will review related work, provide experiments, results, and analysis.

Data

Dataset	Number of Samples	μ_{label}	σ_{label}
tweets_hate_speech_detection	31962	0.07	0.26
Davidson et al.	10944	0.13	0.33
UC Berkeley Design Lab	135556	0.35	0.49
HSLT Group at Vicomtech	23353	0.82	0.19
Dipartimento di Informatica	37281	0.91	0.24
hate_tweets	**207134**	**0.43**	**0.50**

Table 1: Description datasets in dataset ensemble (hatetweets)

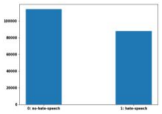


Figure 1: Label Distribution for hatetweets dataset

For our experiments, we used the Hugging Face dataset **tweets_hate_speech_detection** initially and a custom dataset, **hatetweets**. hatetweets is composed of 4 different hate speech datasets from the corpus made by Davidson et al. in their paper Automated Hate Speech Detection and the Problem of Offensive Language, the UC Berkeley Design Lab, the HSLT Group at Vicomtech, Donostia/San Sebastian, Spain, and the Dipartimento di Informatica, University of Turin.

For the 4 datasets, we normalized the label by making a discrete binary categorization with 0: no-hate-speech, 1: hate-speech. The data preprocessing consisted of converting the categorical hate speech scores to a binary categorization, where the threshold score described by the data collectors as "hate speech" became the threshold for the binary categorization. Additionally, we used **conflation** to combine the datasets in order to minimize the loss of Shannon information when combining the distributions. Conflation is defined for if we have distributions $P_1, P_2, ..., P_n$ with probability mass functions $p_1, p_2, ..., p_n$ then the combined conflated distribution $\&P_1, P_2, ..., P_n$ is continuous with the equation

$$\&(P_1, P_2, ..., P_n) = \frac{\sum_{x \in A} \delta_x \Pi_{i=1}^{n} p_i(x)}{\sum_{y \in A} \Pi_{i=1}^{n} p_i(y)}$$

The hatetweets dataset contains data from a variety of sources, including tweets, speeches, web forums, and news articles. We intentionally created a diverse linguistic dataset in order to evaluate whether our model could detect hate speech in different scenarios with different diction and rhetoric. Additionally, all types of hate speech, including discrimination based off of age, ability, race, gender, religion, sexuality, and origin are included with a *roughly* equal split in the dataset.

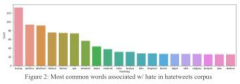


Figure 2: Most common words associated w/ hate in hatetweets corpus

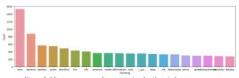
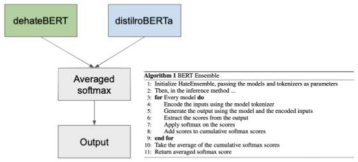


Figure 3: Most common words not associated w/ hate in hatetweets corpus

Experiments



Algorithm 1 BERT Ensemble

```
Initialize HateEnsemble, passing the models and tokenizers as parameters
Then, in the inference method ...
for Every model do
    Encode the inputs using the model tokenizer
    Generate the output using the model and the encoded inputs
    Extract the scores from the output
    Apply softmax on the scores
    Add scores to cumulative softmax scores
end for
Take the average of the cumulative softmax scores
Return averaged softmax score
```

We ran baseline evaluations using the RandomForest, Logistic Regression, Decision Trees, SVC and XGBoost models. For the ensembling, first, we measured the test F1 and Accuracy of the baselines for the dehateBERT, distilroBERTa, and HateEnsemble models. The HateEnsemble-baseline simply ensembled dehateBERT-pretrained-baseline and distilroBERTa-pretrained-baseline models. Next, we finetuned the dehateBERT-pretrained-baseline and distilroBERTa-pretrained-baseline against the hatetweets training dataset separately. Then we ensemble HateEnsemble-finetune from dehateBERT-pretrained-finetune and distilroBERTa-pretrained-finetune.

Results

PART A: CLASSIFICATION

Model	Test F1	Test Accuracy
dehateBERT-pretrained-baseline	0.7468	0.7507
distilroBERTa-pretrained-baseline	0.5621	0.5657
HateEnsemble-baseline	0.6305	0.6359
distilroBERTa-pretrained-finetune	0.9699	0.9703
dehateBERT-pretrained-finetune	0.9369	0.9377
HateEnsemble-finetune	0.9936	0.9937

Table 2: BERT and ensemble models F1 and Accuracy against hatetweets dataset

	Predicted: Hate Speech	Predicted: Not Hate Speech
Actual: Hate Speech	7361	35
Actual: Not Hate Speech	49	6111

Table 3: Confusion Matrix for hatetweets test set on Ensemble Finetune

Given the HateEnsemble-finetune performance is **99.36%** F1 Score, we present samples and analysis for the errors that our model made.

For this sample

"FUCK THIS BITCH"

the model predicted that this is not hate speech, whereas the labeled sentence is flagged as hate speech (False Negative (FN)). This false negative could occur because the sentence is all in UPPERCASE, but the training corpus had mix cases and few samples that were all UPPERCASE. One method to prevent this false negative is to preprocess the corpus to be entirely lowercase, or use a BERT uncased model.
Next, we consider the sample

"I would eat her pussy till ...."

the model predicted this sample as hate speech, whereas the labeled sentence is flagged as not hate speech (False Positive (FP)). Sexual acts and depiction of sexual parts is typical in hate speech, and the model could have mistakenly predicted this as hate speech. To solve this problem, the models could be finetuned against a corpus that includes these examples as not hate speech.

One fundamental limitation or these models is that they are unable to dynamically determine "who" wrote the sample. For example, if a black person uses the N-word against another black person, it may not always be considered offensive. If a white person uses the N-word in almost any context, that white person would be considered a racist, and the speech that contains the N-word would be deemed hateful. Hence, the determination here of whether or not an utterance is considered hate speech is entirely dependent on the speaker. This nuance of hate speech detection has been explored in prior literature, yet it is important to point out in the context of our paper, since despite our high accuracy on a large dataset, there still exist speaker-dependent limitations to hate speech detection.

PART B: NEUTRALIZATION

For Text Neutralization we followed the Pryzant et al. paper which leverages a novel corpus containing 180,000 sentence pairs of subjective bias webscraped from Wikipedia
We ran both models on 10% of our hatetweets dataset (due to compute restraints) and the below table shows our results:

Model	BLEU Score
Concurrent Model	0.4743
Modular Model	0.5122

Table 4: Inference Results for Text Neutralization

As steps to run our end-to-end pipeline we implemented the following steps:

- Tagger Model: The Part Of Speech (POS) tagger model labels the part of speech of each word and tags biased words in the corpus. This process completes in around 2 hours on Google Colab Pro+.
- Concurrent Model: The Concurrent Model converts biased sentences into neutral form and creates sentence pairs in the form (biased, neutral) using a BERT encoder. This process completes in around 40 hrs in Google Colab Pro+ on only 10% of our data. This step led to a performance bottleneck due to compute resources, as it took the original authors 300+ hours to run this step.
- Modular Model: The Modular Model contains both Tagger and Concurrent Models. It is a BERT-based classifier to identify hate words and has a novel Join-Embedding through which the classifier can edit the hidden states. We ran it on 10% of our corpus as well.
- Inference: We ran inference to assess the performance of both the Concurrent and the Modular models. Performance in terms of BLEU scores (score for comparing a candidate translation of text to one or more reference translations) is shown in Table 4.

We do some concrete results generated after the neutralization part. For example the sentence *"Mark (born 8 march 1964, Watford) is a* ***disgrace*** *liberal democrat politician in the United Kingdom , and member of parliament for the Winchester constituency."* gets converted to *"Mark (born 8 march 1964, Watford) is a liberal democrat politician in the United Kingdom , and member of parliament for the Winchester constituency."* with the removal of the "***disgrace***" word.

Conclusion / Future Works

In our experiments, we have proven that the HateEnsemble is able to achieve an F1 score of 99.36% finetuned against our dataset ensemble. Due to the large computing infrastructure required to train the Concurrent Model and our ensemble, we were unable to train our neutralization model on our full dataset, nor were we able to run more experiments on our ensemble model.

With scaling laws, we believe that our ensemble model would perform better if the batch size were larger or if we had the compute to use large pretrained models like the byT5 models. For example, increasing the training batch size from 8 to 16 generates a CUDA out of memory error on a Microsoft Azure NC6s_v3 Ubuntu Linux instance.

Evaluating the qualitative examples can be challenging to review and discuss, as these samples do contain some very hateful content. This also poses a challenge for human turks and deep learning practitioners working on similar tasks, and their willingness to work on these tasks long term due to the emotional damage it may cause.

As good as the intentions are with detecting hate speech and neutralizing them, we also have to be aware that this can also be viewed as a form of censorship and in itself, may be biased on the team or researchers who are evaluating the models performance.

Acknowledgements

We would like to thank our mentor, Manan Rai, for his guidance during this process. Manan helped tremendously in alignment and making sure we were on the right track with our experiments. We would also like to thank Reid Pryzant, author of the original text neutralization paper, for his advice and help adapting his model to our data.