Detecting Hate Speech with LSTMs

Tien Cheng Oh *Singapore Polytechnic, Diploma in Applied AI and Analytics*
tiencheng@pm.me

Abstract—The spread of hate speech through social media platforms has led to greater polarization in society, as hate groups are granted a platform to spread their extremist ideologies. While modern machine learning based moderation tools have been developed to detect explicit hate speech, implicit hate speech can remain undetected. In this paper, several Recurrent Neural Networks (RNNs) are trained and evaluated on a recently published implicit hate speech dataset to build a robust and effective model for detecting hate speech on social media.

Index Terms—Natural Language Processing, Recurrent Neural Networks, Long Short Term Memory Networks, Text Classification, Transfer Learning, Hate Speech

1 Introduction

Hate speech is speech that expresses hate or encourages violence towards a person or group based on race, religion, sex, or sexual orientation. [11] With the rise of social media platforms like Facebook and Twitter, there has been a rise in the spread of hate speech through these platforms. For instance, in the second quarter of 2021, Facebook removed 31.5 million pieces of hate speech on their platform. [5]

The spread of hate speech through social media platforms has led to greater polarization in society, as hate groups are granted a platform to spread their extremist ideologies. In a report on the consequences of hate speech during the COVID-19 pandemic, it was found that the rise in hate speech towards Asian and Black communities in the US and UK were correlated with reported hate crimes.[28] In the worse case scenarios, the spread of hate speech can lead to crimes against humanity, such as in 2018 where the United Nations found that the use of Facebook to spread hate speech against Rohingya Muslims has contributed to acts of genocide against the group. [27]

Traditionally, social media platforms have relied on community reporting and moderators to review and remove hateful and abusive content. However, the large volume and variety of content mean that moderation is difficult and can lead to mental health issues for content moderators who have to review hateful and abusive content. [1]

As such, companies are increasingly reliant on artificial intelligence solutions to automate removing hate speech from their platforms before other users see it. One such solution is Perspective API, a free machine learning service by Google to identify harmful content. [20] However, while these tools can catch a large proportion of explicit hate speech, they can struggle when dealing with implicit hate speech. [9]

1.1 Implicit Hate

Implicit hate speech is a type of hate speech defined by indirect language to express hatred towards a person or group. [9] Unlike explicit hate speech, which often uses slurs and profanity to attack a target group or individual directly, implicit speech is more subtle in using abstract terms and dog whistles to mask the intent behind the hate speech content [8]. This gives plausible deniability to the post, making it harder to report and takedown. As a result of the more subtle nature of implicit hate speech, traditional algorithms for detecting hate speech have been unable to do as well in this task. [9]

2 Related Works

Much work has been done to create machine learning models that can detect and classify hate speech in the past. In discussing related works, we will focus on two primary areas of interest:

- Models for Hate Speech Detection
- Datasets on Hate Speech

2.1 Past Models

Some of the earliest models for hate speech detection relied on classical machine learning. Yin et al. [31] extracted features from online harassment texts using a Term Frequency-Inverse Document Frequency (TF-IDF) algorithm, making use of the TF-IDF representation of a post along with contextual features like a nearest neighbors heuristic for prediction by a linear Support Vector Machine (SVM).

Then, deep learning algorithms, such as Recurrent Neural Networks(RNNs), gained prominence as they began achieving state of the art results in hate speech detection tasks. Badjatiya et al. [2] experimented with CNN and LSTM architectures, making use of both custom learned embeddings and GLoVe embeddings on a Twitter hate speech dataset [30]. They found that deep learning methods with word embeddings significantly outperformed classical machine learning methods like SVMs. They also found that between learned embeddings on the dataset vs GLoVe embeddings pre-trained on a Twitter corpus, the learned embeddings performed better due to them being more fine-tuned for the task of hate speech detection.

In recent times, more advanced deep learning algorithms based on Transformers have seen more use. For example,

Caselli et al. [3] made use of a Bidirectional Encoder Representations from Transformers (BERT) model to classify hate speech by performing pre-training on a Reddit Abusive Language corpus before tuning the model on other hate speech datasets. Similarly to the work of Badjatiya et al., they found that generating BERT embeddings using a hate speech dataset instead of a general-purpose corpus resulted in better performance in the task of detecting hate speech.

2.2 Past Datasets

Over the years, many datasets on hate speech have been collated. For example, in 2012, Warner and Hirschberg [29] collected data from Yahoo on anti-semitic hate speech, collecting 9000 examples. In 2016, Waseem and Hovy [30] annotated 16,914 publicly collected Tweets to create a hate speech dataset using criteria of hate speech based on Critical Race Theory. Further work was contributed by Davidson et al. [7]. They contributed a dataset with 24,802 labelled tweets by collecting tweets that contained terms from HateBase [12], a database of hate speech terms. In a follow-up paper [6], they found that their dataset and other hate speech datasets contain racial biases, as classifiers trained on their data were more likely to pick up Tweets written in African-American English as containing abusive language as compared to Tweets written in Standard American English.

A common trend throughout these datasets is that the hate speech collected consists mainly of explicit hate speech, often containing a high number of profanities [9]. In addition, most of these datasets do not cover a broad range of hate speech. All this means that classifiers trained on such data may have difficulty dealing with hate speech targeted towards groups or individuals not included in the data and may have difficulty classifying a post as hate speech if they do not include explicitly abusive text. Of the recent data sets on hate speech, two datasets contain examples of implicit hate speech. The first one is the Gab Hate Corpus by Kennedy et al. [14], released in 2020, consisting of 27,655 labelled posts, sampled randomly from Gab, a social media platform commonly associated with the Alt-Right in the United States [22]. However, only 9.1 per cent of examples contain examples of implicit hate speech, and 28.2 per cent contain profanities, which may bias models towards posts with profanities. Most recently, in 2021, ElSherief et al. [9] released the Implicit Hate Corpus, containing 22,584 tweets, including tweets taken from various hate groups labelled as implicit hate, explicit hate and non-hate speech. Compared to the Gab Corpus, it has a higher proportion of hate speech, and only 3.2 per cent of posts contain profanities. Their paper bench-marked several models like an SVM and a BERT model on classifying hate speech and the specific type of hate speech. They used a GPT model to generate an implicit hate speech post's implied statement and target group. They found that using a BERT model with data augmentation in the form of back-translation resulted in the best model for detecting hate speech. They also identified challenges faced in identifying implicit hate speech, such as the use of coded hate symbols, which are difficult for a model to understand without sufficient context.

Another common theme throughout these datasets is the imbalanced nature of the data. Regardless of whether it is implicit or explicit hate speech, hate speech is relatively uncommon in comparison to the volume of regular posts on social media. As such, of the datasets discussed above, the dataset with the largest proportion of hate speech, which is the Implicit Hate dataset used in this paper, only 39 per cent of posts contain hate speech.

2.3 Contributions

While Transformer based models like BERT have shown state of the art performance on hate speech detection tasks, these models have a high computational cost for both training and inference. [26] This can make the training and deployment of such models restricted to larger platforms. This paper hopes to contribute to the field by investigating the use of state-of–of-the–the-art architectures with lower resource utilization for implicit hate speech detection. In addition, the use of data preprocessing procedures like spelling correction are tested to improve the data quality.

3 Methodology

3.1 Dataset

The dataset used is the Latent Hatred dataset [9], which contains 22,584 tweets from prominent extremist groups in the United States of America. There are multiple variants of the data, with smaller subsets of the data having more fine-grained labels on the specific type of hate speech and the target of hate speech. This paper is mainly concerned with the task of binary classification: detecting if a tweet can be considered implicit hate speech or not. As such, we ignore the other variants of the dataset. There are three classes in the dataset, which are:

- Not Hate
- Implicit Hate
- Explicit Hate

Since the benchmark provided by the paper does binary classification, ignoring the third class (explicit hate), I follow the same practice to be able to conduct a comparison with the results found, and because there is a lack of sufficient examples of the third class to train a neural network model.

This dataset was developed in response to the small number of hate speech datasets that contained implicit hate speech compared to explicit hate speech. This data was collected over two years from 2015 to 2017 by scraping Twitter accounts identified as members of hate groups identified by the Southern Poverty Law Center. [25] The hate groups included consist of various ideologies, including White Nationalists, Neo-Nazi, Anti-Immigrants, Anti-Muslim, Anti-LGBTQ, and the Ku Klux Klan. [9] Finally, to increase the number of examples of implicit hate speech, the data curators added out of domain samples and performed bootstrapping. [9]

Examples of posts in the dataset are shown in Table 1 Note that although the last post comes from a Neo-Nazi poster, as can be seen by the hashtag supporting National Socialism, the post itself is not inherently hate speech, as it does not encourage hate or violence towards other groups or individuals.

Post	Class
video proof that everyone feels safe in majority white crowds.meanwhile white londoners can't even walk their own streets .	Implied Hate
its not a phobia if the islamic want to kill us ! better to kill them first! period !	Explicit Hate
white people have become slaves to gadgets ! #nationalsocialism #nationalism	Not Hate

TABLE 1: Example of Posts in the Dataset

3.2 Exploratory Data Analysis

Before performing modelling, an exploratory data analysis is conducted.

We begin by investigating the class distribution in the dataset, which is plotted in Figure 1. As can be observed, the implicit hate and explicit hate classes are in the minority, with most tweets being non-hate speech. The full distribution is shown in Table 2

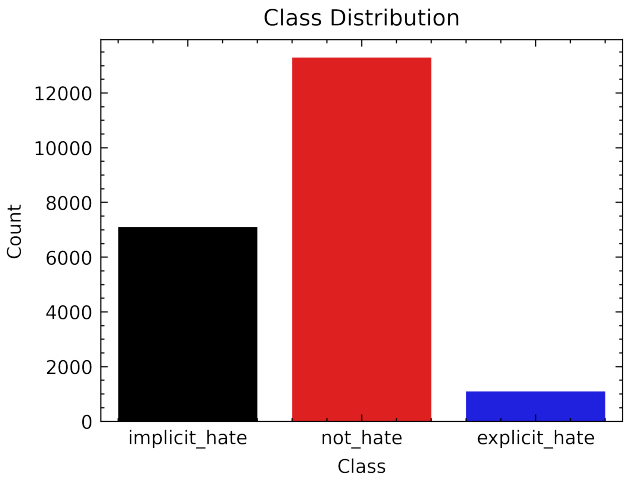

Fig. 1: Class Distribution

Class	Count
Not Hate	13,291
Implicit Hate	7,100
Explicit Hate	1,089

TABLE 2: Class Distribution

The distribution of post length is analyzed, as plotted out as a histogram in Figure 2 and with a per class analysis in Figure 3

Most posts are short and less than 50 words, and it can be observed from Figure 3 that there does not appear to be a significant difference in the distribution of post lengths between the different classes. However, there are some outliers in terms of post length for the Implicit Hate class, with a few posts exceeding 100 words in length.

Next, the most common n-grams in the posts are analyzed. An n-gram refers to a continuous sequence of items of text[17]. For example, a unigram would refer to individual words, while a tri-gram would refer to a three-word phrase. A search was conducted to find the most common uni, bi,

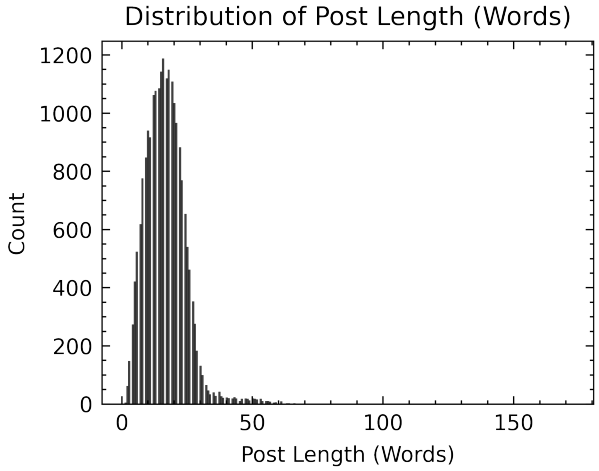

Fig. 2: Distribution of Post Lengths

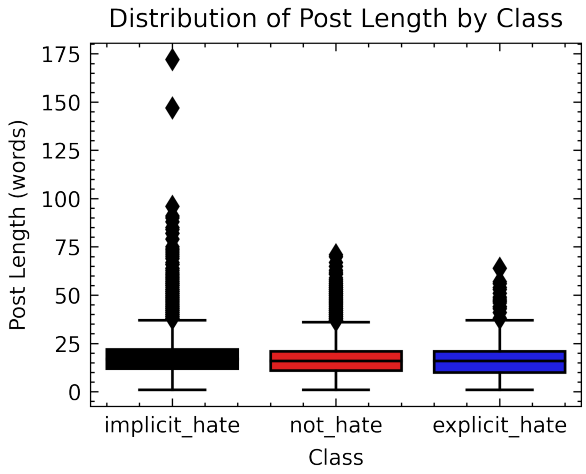

Fig. 3: Distribution of Post Lengths by Class

and tri-grams in the text (excluding English stopwords like "to" or "the"). When plotting out a summary of the most common uni, bi, and tri-grams (see Figure 4), we notice a few key insights.

The top words mainly refer to racial terms, reflecting that a significant amount of hate speech targets people of other races.

When analyzing the texts in the dataset, we can note that some of the posts appear to be mislabelled. For instance, the tweet "go away jew boy ! #jewishsupremacy #israeldid911 usury #istandwithsweden #whitegenocide" is labelled as not hate speech in the dataset, despite the fact that it contains implicit hate speech against Jews. This suggests that there are flaws in the data collection process, and which will make the task of detecting implicit hate speech using the data more difficult. In addition, the data quality of posts is inconsistent, with many spelling mistakes made and some posts being cut off mid sentence.

3.3 Data Preprocessing

To process the data, a set of common preprocessing steps were first followed, which are

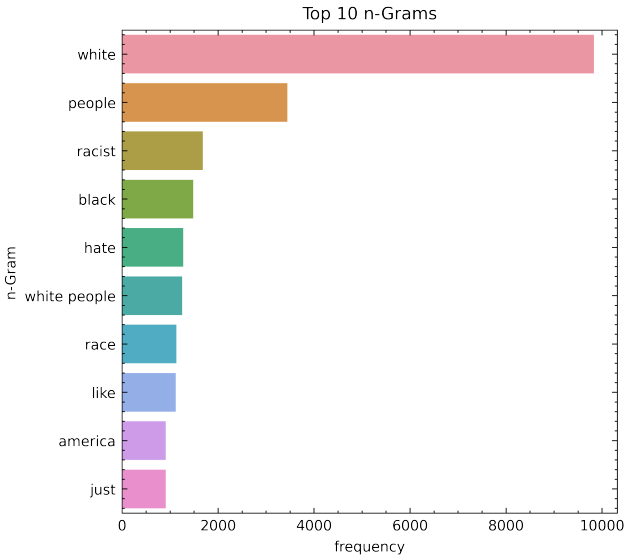

Fig. 4: Top 10 N Grams

1) The removal of the explicit hate class from the dataset
2) Standardization of posts. This involves steps like converting the sentences to the lower case, removing punctuation, and the removal of stopwords. In addition, a spell checking algorithm is used to correct spelling mistakes.
3) The text is then tokenized. The tokenization method used differs according to the model used, and so will be specified later when describing each model tested
4) Finally, the tokenized text is then represented in the form of learned word embeddings. Word embeddings are a dense vector representation of words, which allows words with similar meanings to be closer together in the word embedding space. This allows for better generalization (in comparison to a sparse one hot encoded representation which suffers from the curse of dimensionality), as the model is able to transfer information from each training sentence to other semantically related sentences. [10] The ability for word embeddings to capture the relationships between words and understand word analogies can be seen in Figure 5, where pairs of words with similar relationship are shown to have vectors that point in similar directions.

To process the dataset, the *Pandas*, *Tensorflow*, and *NLTK* Python libraries are utilized.

3.4 Experimental Setup

When training and evaluating the neural networks, the following set-up is used:

- As per the original benchmark [9], a 60-20-20 split into a training, validation, and testing set is done on the data. The split is stratified such that the class distribution is the same in all three splits. This leaves 12234 training examples, 4078 validation examples and 4079 testing examples.

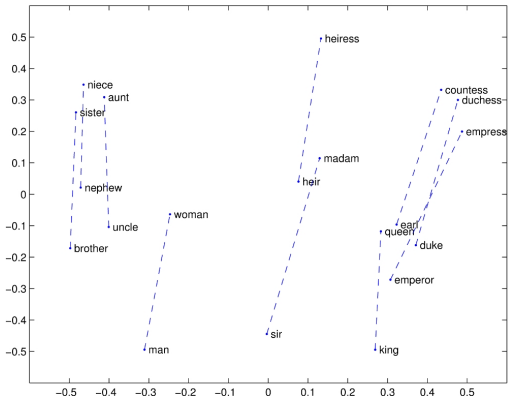

Fig. 5: GLoVe Word Embeddings Projected onto Two Dimensions [19]

- A batch size of 64 is used
- For optimization, the Adam optimizer is used with the default parameters
- To record the details of the experiments conducted, the Weights and Biases (WANDB) tool is used.

Using the validation scores as a guide, each model is tuned, and the validation and testing metrics are recorded using WANDB.

3.5 Metrics

When comparing the various models, several metrics are used, which are

- Precision
- Recall
- F1 Score
- Accuracy

The primary evaluation metric used will be the F1 score, due to the imbalanced nature of the data. However, the recall of a model is also a useful metric which will also be looked at as it relates to the ability of the model to find all posts of the positive class (hate speech).

3.5.1 F1 Score

In a binary classification task, the F1 score is the harmonic mean of precision and recall, and can be calculated as in (1)

$$F_1 = \frac{2 * (precision * recall)}{precision + recall} \tag{1}$$

F1 score is used as in the case where an imbalanced classification problem occurs in the dataset, the F1 score is a better metric for evaluating the performance of a model as compared to a metric like accuracy.

3.6 Modelling

In building a model for implicit hate speech detection, I evaluate a series of models, which in a brief summary are:

- Support Vector Machine (SVM) using Term Frequency-Inverse Document Frequency (TF-IDF) representation

- Bidirectional Deep Long Short Term Memory (LSTM) Model with GLoVe Embeddings
- ASGD Weight-Dropped LSTM (AWD-LSTM) Model [16] Trained Using ULMFiT [13]

To train these models, a variety of libraries are used, namely *Scikit-Learn* (SVM), *Keras* (LSTM) and *Fastai* (ULMFiT).

3.6.1 Baseline SVM with TF-IDF

To provide a non deep learning baseline for comparison, I constructed a simple baseline using the *scikit-learn* library. The text data was first encoded using TF-IDF, before being fed into a support vector classifier.

3.6.2 Custom Bidirectional LSTM

For this paper, a custom bidirectional LSTM architecture is created. It consists of three bidirectional LSTM layers followed by a dropout and a dense layer for classification. For the word embeddings, pre-trained GLoVe embeddings are used. A diagram of the model architecture is shown in Figure 6

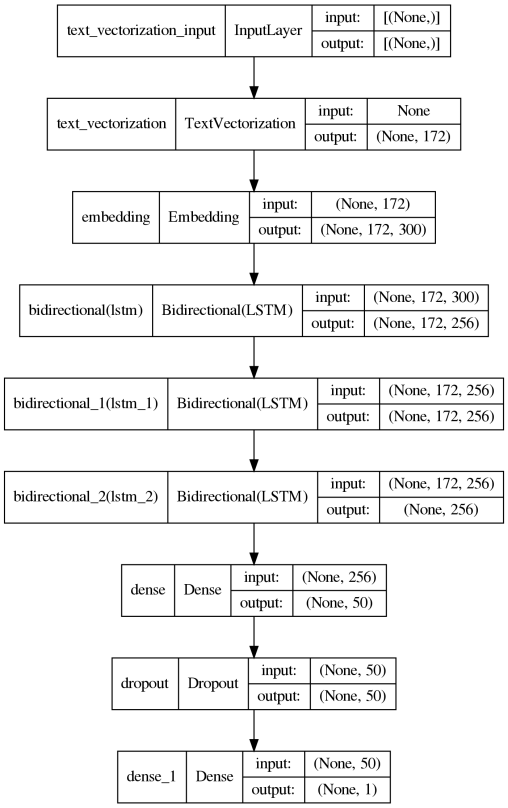

Fig. 6: Custom LSTM Architecture

For the GloVe word embeddings, a 300 dimensional pretrained embedding on the Common Crawl corpus [4] is used, as it was found to perform better than pretrained embeddings on a Twitter corpus.

3.6.3 AWD-LSTM with ULMFiT

Universal Language Model Fine-tuning for Text Classification (ULMFiT) is a state-of–of-the–the-art transfer learning method for sequence models [13], built on the backbone of a pretrained AWD-LSTM model [16].

The AWD-LSTM model is a 3 layer LSTM model, with heavy regularization techniques applied to reduce overfitting. This includes Embedding Dropout, DropConnect, Variational Dropout and Activation/Temporal Activation Regularization. These regularization methods help the LSTM model to achieve competitive performance, despite not having a more complex architecture like attention mechanisms.

In ULMFiT, a AWD-LSTM model is trained using a three-stage process. In the first two phases, the model is trained for language modelling instead of text classification. Firstly, the model is trained on a large corpus of data (in this case, Wikitext-103). This helps the model learn high level features of the English language. Then, the language model is fine tuned to the Hate Speech dataset, so as to help the model adapt to the target dataset. Finally, the model is then fine tuned to be used for classification, using the language model trained as an encoder.

During training, several methods are used to improve the performance of the fine-tuned model. For instance, a one-cycle policy is used during training, introduced by Smith [23], makes use of a slanted triangle learning rate where the learning rate increases linearly to a high learning rate before decaying to a lower learning rate. This allows for the network to converge extremely quickly and reduces overfitting as the high learning rate reached means the model is more likely to reach a flatter minima of the loss function [24], which is known as Super-Convergence . Other techniques include gradual unfreezing, which involves gradually unfreezing the model starting from the last layer during training, and discriminative fine tuning, where updates to earlier layers are given a lower learning rate.

When training the model in the FastAI library, the following steps are done:

1) The training, validation and testing data are loaded, and processed as described in Section 3.3. The tokenization method used is the spaCy tokenizer [15]. We utilize different data loader for the Language Model encoder and the Classifier model
2) A pre-trained AWD-LSTM network on Wikitext is loaded, and trained for language modelling (predicting the next word given the previous sequence), using the slanted triangular learning rate, gradual unfreezing and discriminative learning rate for several epochs.
3) The model is then saved, adapted for text classification, using the same techniques used in the previous step to fine tune the model. Early stopping is applied to stop the model training when the validation F1 score stops improving.

4 Discussion

4.1 Learning Curves

4.1.1 Custom LSTM

When analyzing the loss (see Figure 7) and F1 score curve (see Figure 8) of the Custom LSTM model, it is observed that the model begins to overfit to the training data,

Fig. 7: Loss Curve for Custom LSTM

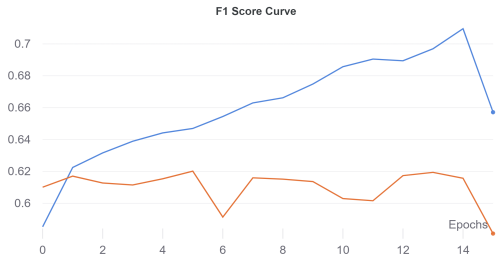

Fig. 8: Training and Validation F1 Score

Fig. 9: Loss Curve for AWD-LSTM

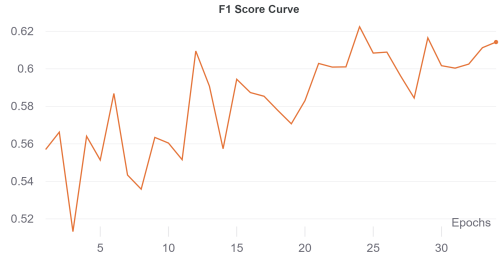

Fig. 10: Training and Validation F1 Score

4.1.2 AWD-LSTM

The loss curve for the AWD-LSTM trained using ULMFiT is shown in Figure 9, as well as that of the F1 score plot in Figure 10. It can be seen that the model begins to overfit after fifteen epochs. It reaches the highest F1 score near epoch 24, achieving an F1 score of slightly above 62.

4.2 Results

4.2.1 Validation Data

The performances of the models evaluated on the validation set are shown in Table 3.

From the validation metrics, we can see that the strongest model in terms of F1 score, precision, and accuracy is the AWD-LSTM model trained using ULMFiT. However, the custom LSTM model has a much stronger recall on the validation set.

4.2.2 Testing Data

The performances of the models evaluated on the final testing set are shown in Table 4. As a point of comparison with the reported results of the original paper, the results reported for the Linear SVM and BERT model are also shown. [9].

As can be seen, although the validation set metrics suggest that the AWD-LSTM is better, the testing set metrics suggest that the Custom LSTM generalizes better as it achieves a superior F1 and recall. Nevertheless, the metrics on the testing set still fall short of those reported of the BERT model by the original authors. However, in terms of recall, the Custom LSTM does outperform that of the BERT model. This suggests that the Custom LSTM could be a good candidate model for hate speech detection, as a high recall means that the model is able to catch a higher proportion of implicit hate speech posts.

4.3 Limitations

One of the limitations of this experiment was the data utilized. As discovered in the Exploratory Data Analysis, the line between implicit hate speech and non-hate speech is ambiguous, and hence there exist mislabelled posts in the dataset. As such, the ability of a model to learn and generalize from the data is limited by the data quality.

4.4 Future Work

One area which could see room for improvement is in taking a data-centric approach to the problem. A data centric approach is a paradigm of machine learning that moves away the focus from the machine learning models towards improving the underlying data used to train these models [18]. As was discussed in the limitations, data labelling issues show that there is room for improvement in improving the quality of the data, which may yield more benefits in terms of performance gained than compared to attempting incremental improvements on the models used.

Another area worth exploring is the use of alternative models for word embedding. For instance, the use of ELMo word representation [21] likely would be an improvement on the the current word embedding methods being used thus far by providing contextual and character-based word representation.

5 Conclusion

In conclusion, several Long Short Term Memory networks are trained on an implicit hate speech dataset with the goal of building a robust model capable of detecting hate speech. While the final models are found to fall behind state–of–the–art model architectures for sequence modelling in the validation metrics, it was found that the models are still able to perform competitively in terms of their recall. It was also found that the data quality of the dataset used could be improved, with instances of mislabelled data found in the dataset.

Model	Preprocessing	Precision	Recall	F1	Accuracy
Linear SVM Baseline	Removal of stopwords, TF-IDF	45.5	62.8	52.8	71.6
Custom LSTM	Common Crawl GLoVe Embedding	53.1	**74.5**	62.0	68.2
AWD-LSTM (ULMFiT)	Spacy Tokenization	**66.2**	52.8	**62.3**	**74.2**

TABLE 3: Validation Metrics

Model	Preprocessing	Precision	Recall	F1	Accuracy
Linear SVM Baseline (ours)	Removal of stopwords, TF-IDF	65.4	46.7	54.5	73.0
Custom LSTM	Common Crawl GLoVe Embedding	56.2	**76.6**	**64.8**	71.1
AWD-LSTM (ULMFiT)	Spacy Tokenization	**66.2**	57.0	61.2	**74.8**
Linear SVM	TF-IDF	59.5	68.8	63.9	71.6
BERT + Data Augmentation	BERT Tokenization	67.8	73.2	70.4	77.5

TABLE 4: Test Set Metrics

REFERENCES

[1] *1 in 1,000 Views on Facebook Is of Hate Speech*. en. URL: https://www.vice.com/en/article/3and8b/1-in-1000-views-on-facebook-is-of-hate-speech (visited on 11/24/2021).

[2] Pinkesh Badjatiya et al. "Deep Learning for Hate Speech Detection in Tweets". In: *Proceedings of the 26th International Conference on World Wide Web Companion - WWW '17 Companion* (2017). arXiv: 1706.00188, pp. 759–760. DOI: 10.1145/3041021.3054223. URL: http://arxiv.org/abs/1706.00188 (visited on 11/24/2021).

[3] Tommaso Caselli et al. "HateBERT: Retraining BERT for Abusive Language Detection in English". In: *arXiv:2010.12472 [cs]* (Feb. 2021). arXiv: 2010.12472. URL: http://arxiv.org/abs/2010.12472 (visited on 11/25/2021).

[4] *Common Crawl*. en-US. URL: https://commoncrawl.org/ (visited on 11/28/2021).

[5] *Community Standards Enforcement Report, Second Quarter 2021*. en-US. Aug. 2021. URL: https://about.fb.com/news/2021/08/community-standards-enforcement-report-q2-2021/ (visited on 11/24/2021).

[6] Thomas Davidson, Debasmita Bhattacharya, and Ingmar Weber. "Racial Bias in Hate Speech and Abusive Language Detection Datasets". en. In: (May 2019). URL: https://arxiv.org/abs/1905.12516v1 (visited on 11/25/2021).

[7] Thomas Davidson et al. "Automated hate speech detection and the problem of offensive language". In: *Proceedings of the 11th international AAAI conference on web and social media*. ICWSM '17. Place: Montreal, Canada. 2017, pp. 512–515.

[8] Quentin Dénigot and Heather Burnett. "Dogwhistles as Identity-based interpretative variation". In: *Proceedings of the Probability and Meaning Conference (PaM 2020)*. Gothenburg: Association for Computational Linguistics, June 2020, pp. 17–25. URL: https://aclanthology.org/2020.pam-1.3 (visited on 11/24/2021).

[9] Mai ElSherief et al. "Latent hatred: A benchmark for understanding implicit hate speech". In: *Proceedings of the 2021 conference on empirical methods in natural language processing*. Online and Punta Cana, Dominican Republic: Association for Computational Linguistics, Nov. 2021, pp. 345–363. URL: https://aclanthology.org/2021.emnlp-main.29.

[10] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT Press, 2016.

[11] *hate speech*. en-US. URL: https://dictionary.cambridge.org/us/dictionary/english/hate-speech (visited on 11/24/2021).

[12] *Hatebase*. en. URL: https://hatebase.org/ (visited on 11/25/2021).

[13] Jeremy Howard and Sebastian Ruder. "Universal Language Model Fine-tuning for Text Classification". In: *arXiv:1801.06146 [cs, stat]* (May 2018). arXiv: 1801.06146. URL: http://arxiv.org/abs/1801.06146 (visited on 11/21/2021).

[14] Brendan Kennedy et al. *The Gab Hate Corpus: A collection of 27k posts annotated for hate speech*. Feb. 2020. DOI: 10.31234/osf.io/hqjxn.

[15] *Linguistic Features · spaCy Usage Documentation*. en. URL: https://spacy.io/usage/linguistic-features#how-tokenizer-works (visited on 11/28/2021).

[16] Stephen Merity, Nitish Shirish Keskar, and Richard Socher. "Regularizing and Optimizing LSTM Language Models". In: *arXiv:1708.02182 [cs]* (Aug. 2017). arXiv: 1708.02182 version: 1. URL: http://arxiv.org/abs/1708.02182 (visited on 11/21/2021).

[17] *N-Grams*. May 2019. URL: https://deepai.org/machine-learning-glossary-and-terms/n-gram (visited on 11/21/2021).

[18] *NeurIPS Data-Centric AI Workshop*. URL: https://datacentricai.org/ (visited on 11/28/2021).

[19] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. "GloVe: Global vectors for word representation". In: *Empirical methods in natural language processing (EMNLP)*. 2014, pp. 1532–1543. URL: http://www.aclweb.org/anthology/D14-1162.

[20] *Perspective API*. URL: https://www.perspectiveapi.com/ (visited on 11/24/2021).

[21] Matthew E. Peters et al. "Deep contextualized word representations". In: *arXiv:1802.05365 [cs]* (Mar. 2018). arXiv: 1802.05365. URL: http://arxiv.org/abs/1802.05365 (visited on 11/22/2021).

[22] Alina Selyukh. "Feeling Sidelined By Mainstream Social Media, Far-Right Users Jump To Gab". en. In: *NPR* (May 2017). URL: https://www.npr.org/sections/alltechconsidered/2017/05/21/529005840/feeling-

sidelined-by-mainstream-social-media-far-right-users-jump-to-gab (visited on 11/25/2021).

[23] Leslie N. Smith. "A disciplined approach to neural network hyper-parameters: Part 1 – learning rate, batch size, momentum, and weight decay". In: *arXiv:1803.09820 [cs, stat]* (Apr. 2018). arXiv: 1803.09820. URL: http://arxiv.org/abs/1803.09820 (visited on 11/25/2021).

[24] Leslie N. Smith and Nicholay Topin. "Super-Convergence: Very Fast Training of Neural Networks Using Large Learning Rates". In: *arXiv:1708.07120 [cs, stat]* (May 2018). arXiv: 1708.07120. URL: http://arxiv.org/abs/1708.07120 (visited on 11/25/2021).

[25] *Southern Poverty Law Center*. en. URL: https://www.splcenter.org/ (visited on 11/21/2021).

[26] Emma Strubell, Ananya Ganesh, and Andrew McCallum. "Energy and Policy Considerations for Deep Learning in NLP". In: *arXiv:1906.02243 [cs]* (June 2019). arXiv: 1906.02243. URL: http://arxiv.org/abs/1906.02243 (visited on 11/25/2021).

[27] "UN: Facebook has turned into a beast in Myanmar". en-GB. In: *BBC News* (Mar. 2018). URL: https://www.bbc.com/news/technology-43385677 (visited on 11/24/2021).

[28] *Uncovered: Online Hate Speech in the Covid Era*. en-GB. URL: https://www.ditchthelabel.org/research-papers/hate-speech-report-2021/ (visited on 11/24/2021).

[29] William Warner and Julia Hirschberg. "Detecting Hate Speech on the World Wide Web". In: *Proceedings of the Second Workshop on Language in Social Media*. Montréal, Canada: Association for Computational Linguistics, June 2012, pp. 19–26. URL: https://aclanthology.org/W12-2103 (visited on 11/24/2021).

[30] Zeerak Waseem and Dirk Hovy. "Hateful Symbols or Hateful People? Predictive Features for Hate Speech Detection on Twitter". en. In: *Proceedings of the NAACL Student Research Workshop*. San Diego, California: Association for Computational Linguistics, 2016, pp. 88–93. DOI: 10.18653/v1/N16-2013. URL: http://aclweb.org/anthology/N16-2013 (visited on 11/24/2021).

[31] Dawei Yin et al. "Detection of harassment on Web 2.0". In: (Jan. 2009).