Received: 28 August 2021 | Revised: 20 October 2021 | Accepted: 29 October 2021 | Published online: 17 December 2021

RESEARCH ARTICLE

Journal of Computational and Cognitive Engineering
2023, Vol. 2(4) 278–286
DOI: 10.47852/bonviewJCCE2022010102

BON VIEW PUBLISHING

Application of Transformer-Based Language Models to Detect Hate Speech in Social Media

Swapnanil Mukherjee[1] and Sujit Das[2,*]

[1]*Department of Computer Science, Ashoka University, India*

[2]*Department of Computer Science and Engineering, National Institute of Technology Warangal, India*

Abstract: Detecting and removing hateful speech in various online social media is a challenging task. Researchers tried to solve this problem by using both classical and deep learning methods, which are found to have limitations in terms of the requirement of extensive hand-crafted features, model architecture design, and pretrained embeddings, that are not very proficient in capturing semantic relations between words. Therefore, in this paper, we tackle the problem using Transformer-based pretrained language models which are specially designed to produce contextual embeddings of text sequences. We have evaluated two such models—RoBERTa and XLNet—using four publicly available datasets from different social media platforms and compared them to the existing baselines. Our investigation shows that the Transformer-based models either surpass or match all of the existing baseline scores by significant margins obtained by previously used models such as 1-dimensional convolutional neural network (1D-CNN) and long short-term memory (LSTM). The Transformer-based models proved to be more robust by achieving native performance when trained and tested on two different datasets. Our investigation also revealed that variations in the characteristics of the data produce significantly different results with the same model. From the experimental observations, we are able to establish that Transformer-based language models exhibit superior performance than their conventional counterparts at a fraction of the computation cost and minimal need for complex model engineering.

Keywords: transformer, hate speech, social media, RoBERTa, XLNet, fine-tuning, natural language processing

1. Introduction

With the exponential rise in Internet usage in the last decade, the popularity and adoption of social media platforms have also risen greatly. These platforms provide an open space for individuals and communities to voice their opinions and conduct business. But due to this substantial degree of freedom granted to users, they often misuse such open platforms to pejorate, demean, and intimidate other users through the usage of offensive and hateful language. Although there is no formal and universally accepted definition of hate speech, from general consensus, we can define hate speech as something along the lines of a direct attack on an individual or a group of individuals based on their personal characteristics such as race, ethnicity, gender, religious affiliation, disability, etc.; this may or may not always include swear, curse, or abusive terminology. Due to a heavy increase in such content, social media companies are often faced with the strenuous task of detecting and timely removing such content from their platforms. In the recent past, many approaches concerning the problem of hate speech detection have been explored, which include classical machine learning models as well as deep learning approaches. But to the best of our knowledge, the research works that leverage state-of-the-art Transformer models for the task of hate speech detection are limited.

The Transformer (Vaswani et al., 2017) architecture with the self-attention mechanism, since its inception, has formed the basis of many cutting-edge language models (LMs) such as BERT (Devlin et al., 2019) and GPT (Radford et al., 2018). These models, having millions of parameters, are trained on vast text corpora (sometimes including the entire Wikipedia) and have outperformed all existing deep learning approaches in all major natural language processing (NLP) tasks (Vaswani et al., 2017) (classification, generation, summarization, etc.) and established benchmarks due to their semantic representation of text sequences. Currently, Transformer-based models achieve state-of-the-art performance in all sequence classification tasks. Hence, there arises an obvious need to investigate the applicability of Transformer models in the framework of hate detection study. Some significant contributions in hate speech detection are narrated below. In the domain of classical machine learning models, Davidson et al. (2017) proposed a machine learning approach based on Logistic Regression (LR) and Support Vector Machine (SVM) classifier to classify a corpus of ≈25k tweets divided into three categories—Hate, Offensive, and Neither. They focused on the distinction between tweets that contain hate speech and ones that are offensive by arguing that the mere presence of offensive words or language does not constitute "hate speech" since the use of such language is

*Corresponding author: Sujit Das, Department of Computer Science and Engineering, National Institute of Technology Warangal, India. Email: sujit.das@nitw.ac.in

characteristic to many groups and is also quite prevalent in popular culture. They noted that due to the similarity in content and structure of hateful and offensive tweets, and the limited number of hate tweets (5%) in the corpus as compared to offensive tweets (70%), many tweets belonging to the hate class were misclassified as offensive. An important aspect of their study was that they used manually extracted features from the corpora, instead of generating vector representations of the text samples. Next, Sreelakshmi et al. (2020) used different machine learning models such as SVM and Random Forests for classification for detecting hate speech text in English–Hindi code-mixed data using embeddings obtained from pretrained models such as fastText and doc2vec. In the space of deep learning models, Gomez et al. (2020) and Undirwade and Das (2021) focused on hate speech detection in multimodal publications, which contain images along with text. They created a multimodal dataset consisting of 150k samples, each containing a text and an image. For the text classification task, a 150-layer long short-term memory (LSTM) was used in conjunction with pretrained GloVe (Pennington et al., 2014) embeddings. Zhang et al. (2018) created a custom dataset of tweets concerning particular communities and classified them into "hate" and "nonhate" categories with their proposed model, which was a combination of convolutional neural network (CNN) and gated recurrent unit (GRU) models using Google's word2vec (Mikolov et al., 2013) pretrained on the Google News Corpus. They evaluated the performance of their model on other existing datasets, and it outperformed all existing baselines by significant margins. Qian et al. (2019) introduced two hate speech datasets which—instead of Twitter like most other publicly available datasets—were sourced from Reddit and Gab containing ≈22k and ≈34k samples, respectively. In addition to detecting hate speech, another objective task was automatic generative intervention whenever hate speech was detected. This was aimed at actively discouraging and reprimanding online hate speech in addition to detecting and removing such content. For the binary detection task, they experimented with an ensemble of classical machine learning models (LR and SVM) and deep learning models (CNN and GRU). It is important to note in this regard that all classical models (SVM, Random Forests, etc.) use handcrafted features such as TF-IDF and N-Grams while deep learning methods use pretrained vectors generated from static representation models such as word2vec and GloVe, which do not produce contextual embeddings. Recent study with Transformer models is given in Alshalan and Al-Khalifa (2020), where Alshalan and Al-Khalifa, (2020) investigated hate speech detection in Arabic tweets (in the Saudi context) by evaluating four deep learning models, one of which was the Transformer-based BERT. Despite BERT being the state-of-the-art model among those that were tested, it performed worse than other deep learning models such as GRU and 1D-CNN. This was attributed to the fact that BERT is pretrained on the English Wikipedia database and the problem concerned tweets in Arabic. Roy et al. (2020), in their study, used a customized RoBERTa model (Liu et al., 2019) to achieve the best results on a multilingual Twitter dataset sourced by the organizers of FIRE-HASOC 2020. The authors, focusing exclusively on the Transformer-based RoBERTa, demonstrated that large pretrained Transformer LMs are highly potent in classification tasks even when the dataset size is very small as compared to most other deep learning applications. Mutanga et al. (2020) also evaluated an ensemble of Transformer models, focusing mainly on the DistilBERT model, which is a scaled-down version of the original BERT with increased speed while maintaining considerable consistency in performance.

Through this study, we aim to leverage the pre-encoded knowledge in such models to detect hateful content on social media based on a fine-tuning approach. We observed that much of the existing research focused only on datasets sourced from Twitter with slight variations in them and used classical machine learning models alongside extensive hand-crafted features or deep learning methods with pretrained language embeddings, which are not often semantically accurate. Therefore, extending the existing work in this domain, our contributions in this paper are twofold. First, we evaluate two Transformer-based LMs—XLNet (Yang et al., 2019) and RoBERTa (Liu et al., 2019) and compare their performance to the existing state-of-the-art models to determine if they yield better results without requiring extensive model architecture or manual feature engineering. Second, we have evaluated the models over four datasets, two of which are from Reddit and Gab and the other two from Twitter, to broaden the scope of the study and analyze the effect the data from different social media platforms have on model performance, due to difference in their characteristics. We perform a binary classification task on three datasets (Hate or Non-Hate) and a multiclass classification task on one dataset (Hate, Offensive, or Neither).

The remainder of the paper has been structured in the following way: Section 2 contains background information mainly regarding the datasets. Method and experiments are presented in Section 3 followed by results and discussion in Section 4. Key conclusions are given in Section 5.

2. Backgrounds

This section briefly discusses the publicly available datasets that are used in this study, which includes an overview of each dataset, the techniques used for preparing the data in a usable format, and statistics of samples in each dataset after data cleaning and preparation.

2.1. Twitter dataset

Davidson et al. (2017) introduced a tweet dataset that was crawled from Twitter, based on terms from a hate speech lexicon obtained from Hatebase. This dataset contains 24,802 tweets distributed into 3 classes: Hate, Offensive, and Neither. The distribution of tweets in the three classes is shown in Figure 1.

The dataset was manually labeled by workers from CrowdFlower with at least three workers annotating every tweet, so as to avoid any personal bias. A characteristic of this dataset is that the data are highly skewed with the majority class containing 75% of the samples, and such severe data imbalance often makes

Figure 1
Distribution of tweets in the Davidson Twitter Dataset: Hate (5%), Offensive (76%), and Neither (16.6%)

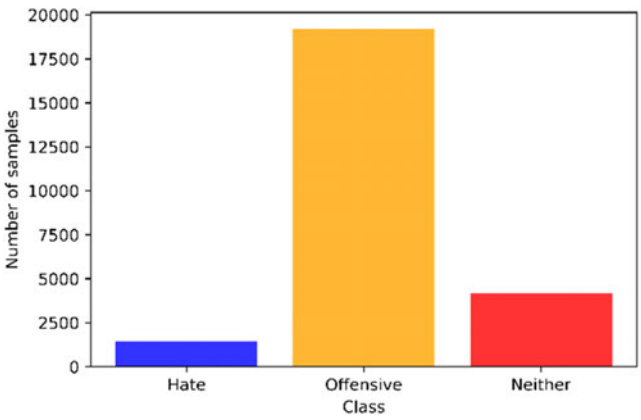


Figure 2
Class distribution of Reddit comments: Hate (30.80%) and Non-Hate (69.20%)

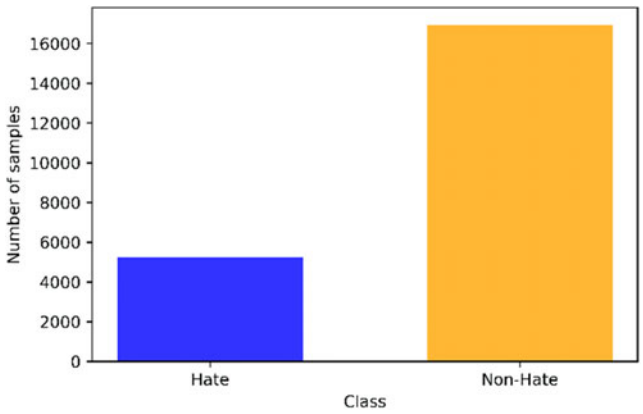
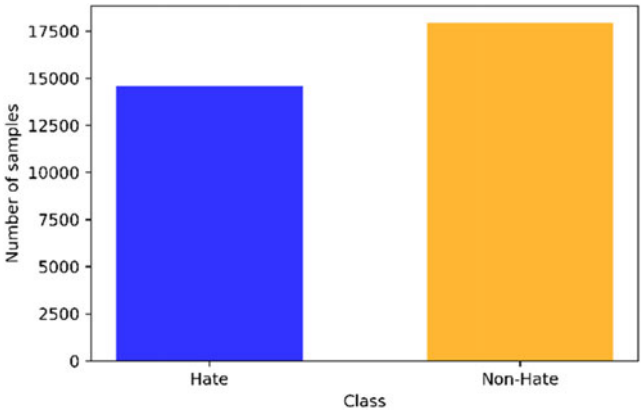


Figure 3
Class distribution of the Gab comments: Hate (43.26%) and Non-Hate (56.73%)



it very difficult for a neural network to learn the representations appropriately. Nevertheless, no data augmentation techniques were applied to normalize the class imbalance since it was also not implemented in the original text.

2.2. Reddit dataset

This dataset along with the Gab dataset was first introduced in Qian et al. (2019) and contains 22,304 comments sourced from some of the "most toxic subReddits" such as *r/DankMemes, r/Imgoingtohellforthis, r/KotakuInAction,* and *r/MensRights*. The data were then crowd-sourced to Amazon Mechanical Turk workers, with each comment being labeled by three workers. The class of each comment was assigned by majority voting of the labels assigned by the workers. A detailed account of the data collection process is provided in Qian et al. (2019). The class distribution of comments in the Reddit dataset is depicted in Figure 2.

This dataset, similar to the Davidson tweets dataset, has a skewed class distribution but the skewness is not as severe. Characteristically, the comments in this dataset are much longer than the ones in the other three datasets used in this task as depicted in Table 1.

2.3. Gab dataset

The Gab dataset was introduced in Qian et al. (2019), but it is larger than the other datasets containing around 33,776 comments and the class distribution of the data can be considered to be almost balanced for deep learning applications. The procedure of data collection was the same as outlined in Section 2.2. An interesting characteristic of the Gab dataset is the near-balanced class distribution itself. The fact that this specific dataset contains almost 3 times more hate comments than the other datasets evidently points to the abundant availability of such content on the Gab platform. This can be easily attributed to Gab's policies about free speech and content moderation on their platform and the user base it fosters (Brandt & Dean, 2021). The class distribution for this dataset is shown in Figure 3.

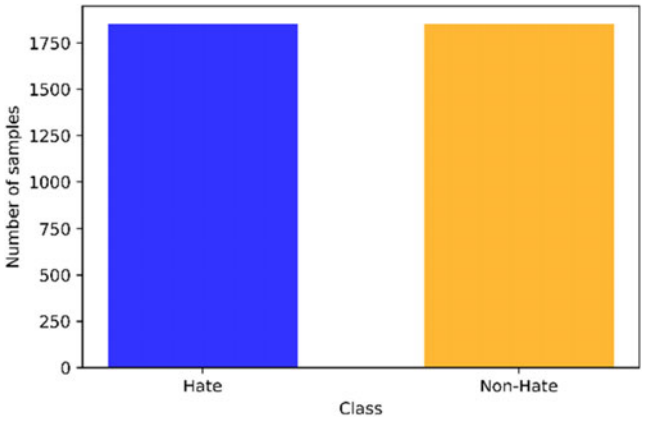
2.4. HASOC 2020 dataset

Hate Speech and Offensive Content (HASOC) identification is a competition aimed at promoting research into hateful content detection in online platforms. The dataset released by the organizers for the 2020 iteration of the competition is a multilingual one, consisting of tweets in three languages: English, German, and Hindi. The data are first divided into two classes: HOF (Hate and Offensive) and NOT (Non-Hate-Offensive). The HOF data are further subdivided into three classes: HATE (Hate), OFFN (Offensive), and PRFN (Profane), giving rise to two subtasks. Subtask A is about the coarse-grained classification of tweets into Hate and Non-Hate categories and Subtask B is related to the fine-grained classification of tweets containing Hate Speech into the three subcategories as mentioned earlier.

Table 1
Statistical analysis of all datasets

Dataset	Tokenizer	Avg sequence length	Avg sentence length	(p)Std. of sequence lengths	No. of samples having seq length > avg + 2*pstd
Davidson Tweets	Bert	18.04	12.71	8.18	670
	RoBERTa	16.98		7.41	746
	XLNet	18.64		8.42	729
Reddit	Bert	52.18	44.58	81.03	744
	RoBERTa	49.36		76.44	753
	XLNet	52.28		80.21	756
Gab	Bert	36.79	27.10	43.50	575
	RoBERTa	34.34		43.31	565
	XLNet	37.62		54.19	563
HASOC (train)	Bert	19.47	14.17	9.27	81
	RoBERTa	18.12		8.06	51
	XLNet	19.84		9.11	75

Figure 4
Class distribution of the train split of the HASOC 2020 English dataset: Hate (50%) and Non-Hate (50%)



For the sake of maintaining consistency across the datasets, in this study, we focus on the English tweets dataset and Subtask A. The organizers provide prebuilt train and test splits of the English dataset, and the train set contains a total number of 3,708 tweets out of which 1,856 tweets are categorized as HOF and the rest NOT. Figure 4 shows the class distribution for the English HASOC 2020 dataset. The test split contains a total of 815 tweets out of which 423 are HATE and 391 are NOT.

2.5. Data preparation

Since we are using Transformer models for our classification task, the raw data need to be preprocessed in a manner similar to the way the pretraining data were prepared during the training of that particular language model. If that is not done, the representations generated by the LM might be different than what we expect due to characteristic differences in the input data, which might lead to suboptimal performance on downstream tasks. For preparing the data in all four datasets, we broadly perform the following operations along with some platform-specific alterations due to the obvious differences between the nature of the raw data from different platforms.

- All rows with missing values were dropped.
- All usernames (e.g., @mike334) were removed.
- All platform-specific artifacts such as "RT" in Twitter and "\t", "r/" in Reddit and Gab were removed.
- Hashtag symbols (#) were removed but the hashtag contents were preserved (e.g., ###BlackLivesMatter was changed to BlackLivesMatter).
- All punctuations, links, URLs, and special characters were removed.
- The data were not tokenized during preprocessing. The corresponding tokenizer of each LM was used to tokenize the data at runtime.

For the Reddit and Gab dataset, comments from each conversation were separated out as individual samples. A Python package named *Redditcleaner* (Leitner, 2021) was used to clean the comments. All the datasets contain emojis in the input sentences, with the number being comparatively higher in the Twitter datasets than the Reddit and Gab datasets. We conducted experiments with emojis removed as well as preserved. In the cases where emojis were preserved, we changed the emojis into their text transcript (e.g., 😂 is converted to *face_with_tears_of_joy)* using the *emoji* (Kim, 2021) Python package.

2.6. Data statistics

In this subsection, we give an account of the statistical analysis of the data in all four datasets, which are important while setting the various hyperparameters of the language models for generating accurate representations. We analyze the data with three tokenizers: BertTokenizer (WordPiece), RoBERTaTokenizer (WordPiece), and XLNetTokenizer (SentencePiece). Although we do not compare the BERT model for the classification task in this work, the BertTokenizer has been included to provide a baseline context.

2.7. Models

We have used two language models in this study—XLNet (Yang et al., 2019) and RoBERTa (Liu et al., 2019). Although both of these models are based on the original BERT, some changes have been incorporated in their training schema to alleviate some of the drawbacks of BERT. XLNet employs Transformer-XL (Dai et al., 2019) as its backbone, an upgraded version of the original Transformer where the "XL" stands for extra large. Transformer XL is capable of effectively handling sequences much longer than 512 tokens (the limit for the original Transformer) and introduces two techniques (Recurrence Mechanism and Relative Positional Encoding) which enable it to do so. It is also 1800+ times faster than the vanilla Transformer and beats it on all benchmarks due to these improved training paradigms. Consequently, XLNet proved to be a much better model than its predecessors.

RoBERTa was introduced by Facebook AI in 2019. One of its key differences from BERT is that it removed the Next Sentence Prediction pretraining objective from its training scheme. Some of the model hyperparameters were also changed; the learning rate and mini batch size were increased, and it was trained for much longer on almost 10x more data than BERT. Finally, it yielded state-of-the-art results and matched the leading score (XLNet's) on the GLUE benchmark.

3. Method and Experiments

In this paper, we try to determine the efficacy of pretrained Transformer-based language models through fine-tuning. To determine how good these models perform in their bare form, without any custom architectural addition or algorithmic enhancement, we use the vanilla Transformer models along with a simple classification head.

3.1. Setup

All experiments were conducted on Google Colab and Kaggle Kernels. We used the Transformers (Wolf et al., 2020) library, which contains HuggingFace's implementation of Transformer-based language models. The code was implemented in PyTorch (Paszke et al., 2019) and SkLearn (Pedregosa et al., 2011). All experiments on the Davidson dataset were conducted on the multiclass distribution.

3.2. Fine-tuning

We evaluated two Transformer language models in this study: XLNet and RoBERTa. Both of them are based on BERT, where some of the shortcomings of BERT are mitigated through approaches explained in Khan (2021). Since our problem focuses on "classifying" text into two or more categories, we used the *XLNetForSequence Classification* and *RobertaForSequenceClassification* models provided by the HuggingFace API, which adds a classification head to the base models. We did not use a base model and then add a

custom classification head to it as our aim was to evaluate to what degree the sole Transformer models could perform without any special enhancements. But it is often done when more control on the model output and architecture of the classification head is desired (Roy et al., 2020).

Since the LMs are already pretrained on vast amounts of data, we are only required to fine-tune them for the task at hand. There are standard strategies for fine-tuning Transformer models for classification, the most common of which are outlined as follows:

1. Freezing all the layers of the Transformer and only fine-tuning the weights of the classifier layer
2. Freezing some of the initial layers of the Transformer model and fine-tuning the remaining layers along with the classifier layer
3. Fine-tuning all the layers of the Transformer.

There is no consensus regarding which of these strategies work best, and it usually depends on the data used for the given problem. Thus, we tested on all of them to determine which of them yielded the best results for our task. For this, we trained all models on randomly shuffled 85% of the data in each dataset (in separate experiments) and tested model performance on the remaining 15% data. The results of these experiments are presented in Tables 3 and 4.

A common practice in classification problems where the class distribution is skewed is to use *class weights,* where weights are assigned to each class based on the number of samples belonging to a particular class, which is factored into the loss function so that the model is penalized more when it makes an incorrect prediction on the minority class(es), which reduces the bias of the model toward the majority class. Since the dataset given in Davidson et al. (2017) was skewed much in comparison to the other datasets, we compared the effect on model performance with and without adding class weights to the loss function. We obtained lower F1 scores when using class weights. A discussion of the same has been undertaken in Section 4.

3.3. Hyperparameter selection

One of the most crucial aspects of training deep neural network models is the selection of optimal hyperparameters. Since we were dealing with Transformer models, which, without being supplied the correct hyperparameters, can easily perform worse than the simplest models, hyperparameter's selection was especially important. After determining the best approach for fine-tuning the models from the experiments in Section 3.2 with a fixed set of hyperparameters, we tested different hyperparameters that affected model performance the most while the others were left to their default values. Adam optimizer, provided by the Transformers library, was used in all experiments.

The learning rate is, by far, the most important hyperparameter, which can drastically affect results if not selected appropriately. We experimented with different learning rates, and the LR of 3e-5 worked optimally for us yielding the best results on all datasets. The learning rate of 2e-5 also performed quite well, producing slightly lower results than the former, but in some cases, it failed to converge, which led us to use the rate of 3e-5 for all other experiments. After determining the best model configuration for the fine-tuning strategy, we used a linear schedule for the learning rate during cross-validation and when compared against cross-validation with a constant LR, it outperformed the constant rate in most cases by small margins.

Batch size is another factor that influences the cost of calculation, and subsequently the optimization process of neural networks. We tested batch sizes of 32, 64, and 128 but they made no noticeable difference. Hence, the default batch size of 32 was used to prevent running out of memory especially while training on the Reddit and Gab datasets, which have significantly long sequences.

Elaborating on sequence lengths, Transformer models can only take inputs of a fixed specified length, the default and maximum in most cases being 512. The tokenizer for every model served by HuggingFace has an attribute "max_length," which specifies the maximum length of the input the model can take. Shorter sequences are padded and longer sequences are truncated to that length. This hyperparameter is of utmost significance since it alters the data being inputted to the model, thus affecting model performance. We determined the max length parameters for each dataset based on the distribution of sequence lengths in that particular dataset. Apart from the effect on the model performance, it is interesting to note that the computational time increases in proportional to the sequence length of the data. The Reddit and Davidson tweets datasets approximately contain the same number of samples but the max_length for the Reddit dataset was ≈6.5 times the max_length for the Davidson tweets dataset, and this is reflected in the training time. It took us around 4 mins to train 1 epoch on the tweets dataset, whereas it took 30 mins for the same on the Reddit dataset (≈7× the time for the tweets dataset).

For the parameter weight decay, we experimented with two weight decay rates of 0.1 and 0.01 and discovered that they both produced similar results, the latter being slightly more stable throughout training. Both these values worked almost at par for our datasets and the difference in results was marginal.

We trained all models for all experiments to the point where their performance on the test and validation sets started to drop, so as to gain a definite idea about the number of epochs to train the models where their performances peak. In most of the experiments, we noticed that the performance was highest after training for three epochs, and in some cases, two epochs. On further training, the model started to overfit and the validation and test accuracy dropped steadily. We present a summary of the best hyperparameters for both models on all four datasets in Table 2.

After determining the best configuration of each model and for every dataset, we performed 5-fold cross-validation with the selected configuration(s) to determine the ability of the models to generalize to unseen data and to validate whether the results achieved in the previous experiments were coincidental and reproducible only for

Table 2
Selected hyperparameters for each dataset for the classification task

Learning rate (LR)	Batch size	Weight decay	Number of epochs	
3e-5 (Linear schedule)	32	0.1	3	
	Davidson tweets	Reddit	Gab	HASOC
Sequence length	40	260	128	50

the particular data split. In addition to this, we trained a model on the entire Gab dataset and made predictions on the Reddit dataset and *vice versa*, and the results were in line with all the past experiments showing that the Transformer-based models are extremely robust, even to the point of achieving performance at par with models natively trained and tested on data belonging to only one source distribution.

4. Results and Discussion

In this section, we showcase the results of the experiments outlined in the preceding section to show that the Transformer models perform significantly better than existing approaches on all the datasets that we tested. Since the datasets in consideration are unbalanced, we settled upon the F1 score as the primary measure of model performance as accuracy can sometimes be very misleading in the case of unbalanced data. Hence, we only report the F1 scores (expressed as a percentage) accounting for class imbalance and make all comparisons with previous works in respect to the same.

4.1. Emojis

As introduced in Section 2.5, we compared the results before and after removing emojis from the Twitter datasets, and the latter yielded better performance. This is surprising because emojis provide useful context in a piece of text, and previous works have shown that model performance (Roy et al., 2020) was benefited by incorporating emojis. A possible explanation for this could be the way the emojis are passed on to the model. Roy et al. (2020) used emo2vec (Wang et al., 2020) to convert the emojis directly into vector representations and then concatenated them to the (cleaned) text representation obtained from the Transformer model, which was then inputted to the classifier. On the other hand, our preprocessing approach replaced the emojis with their respective textual transcript, which creates an abrupt disruption in the otherwise natural flow of a sentence. Since the Transformer looks at the neighboring context of every token in the sequence, the sudden interruption by the emoji transcription within the sentence may have created confusion for the model.

4.2. Class weights

As introduced in Section 3.2, surprisingly, the usage of class weights negatively impacted model performance. A possible explanation for this occurrence is that class weights penalize the loss function more for making incorrect predictions on the minority class compared to making incorrect predictions on the majority class. The number of samples in the "hate" and "neither" classes is very less as compared to the "offensive" class, and as reported in Davidson et al. (2017), there is a lot of linguistic and semantic overlap between the hate and offensive class, which makes the data ambiguous in respect to the pretrained representations of the Transformer, and consequently, the classification task also becomes harder. Hence, the model while not being able to improve performance on the minority classes also performs worse on the majority class due to increased penalty. Thus, the overall performance is going down.

4.3. Fine-tuning strategy

The best results achieved on a particular test split for each fine-tuning strategy tested are presented in Tables 3 and 4.

Table 3
Performance of different fine-tuning strategies for XLNet

Strategy	Davidson Tweets	Reddit	Gab	HASOC
Classifier layer only	88.41	89.16	88.67	84.52
Classifier layer with last 2 encoder layers	90.70	92.00	91.31	86.60
Classifier layer with last 5 encoder layers	90.50	91.87	91.34	86.85
All layers	90.87	92.34	92.06	88.94

Table 4
Performance of different fine-tuning strategies for RoBERTa

Strategy	Davidson tweets	Reddit	Gab	HASOC
Classifier layer only	87.34	89.42	89.45	85.21
Classifier layer with last 2 encoder layers	91.07	91.78	91.60	88.20
Classifier layer with last 5 encoder layer	90.38	92.15	91.75	88.43
All layers	91.66	92.40	92.75	89.80

From Tables 3 and 4, a general trend can be noticed that model performance increases with the number of encoder layers being fine-tuned, and the best performance was achieved when all layers of the model were fine-tuned. It is notable in this regard that the difference of performance between freezing the first seven layers and the first five layers is not very pronounced. This can be attributed to the fact that each encoder layer of the Transformer encodes different information about the input, but fine-tuning three more layers, which are directly next to the layers fine-tuned in the previous experiment, will not provide a big performance improvement because they are likely to encode a similar kind of information.

4.4. Cross-validation

After determining the optimal fine-tuning strategy and values for the hyperparameters from the initial experiments, we performed 5-fold cross-validation on three datasets with the data distributed proportionally in each fold. We also used a linear schedule on the learning rate, which proved to yield slightly better results than the constant rate. We left the HASOC dataset out of this experiment since it already contained a predefined test split. Therefore, cross-validating the model to observe generalization ability was not necessary as we were only required to maximize performance on the given test split. From Table 5, it can be observed that the cross-validation F1 scores are lower than the scores on the individual test split, which shows that a model might perform exceptionally well on a data split due to the distribution of that particular split but it may not necessarily

Table 5
F1 scores for 5-fold cross-validation on each dataset

Model	Davidson tweets	Reddit	Gab
XLNet	90.69	91.48	91.76
RoBERTa	90.75	91.81	92.21

perform so well on unseen data. In our case, the cross-validation scores, although lower, are quite close to the best scores on the test split so it can be inferred that these models are capable of generalizing quite well to new data.

4.5. Cross testing

We studied a step beyond cross-validating models on different folds of the training data and trained models on a particular dataset only to assess its performance on an entirely different dataset. This was done to test if these huge Transformer-based LMs, which already encode a lot of general textual information owing to their pretraining, are truly robust enough to perform decently on new data if fine-tuned appropriately for that task, independent of their pretraining data. For this, we trained two RoBERTa models each on the Gab and Reddit datasets, respectively, and tested them on the other dataset. The Reddit and Gab datasets were chosen due to the similarity in their platforms and the characteristics of the data. The model trained on Reddit data achieved an F1 score of 90.35 on the entire Gab dataset and the model trained on Gab data achieved an F1 score of 91.31 on the Reddit dataset. The lower score for the model trained on the Reddit data is supported by the fact that the class distribution of the Reddit dataset is quite imbalanced in comparison to that of the Gab data. But overall, it can be noticed that the F1 scores are quite close to those achieved in the previous experiments where models were trained and validated on different splits of the same dataset. It can be thus ascertained that if Transformer-based LMs are fine-tuned on sufficient amounts of high quality for a particular task, they are very likely to perform quite good on new data, which might be from an entirely different source given the new data are characteristically and linguistically not very different from the data used for fine-tuning.

4.6. Comparison and analysis

The section compares the performance of the Transformer LMs with the best performance attained in Davidson et al. (2017), Qian et al. (2019), and Roy et al. (2020) on the respective datasets. The results presented in Table 6 are the best scores selected from multiple runs of the 5-fold cross-validation.

It is clear from the results that both the Transformer LMs, XLNet and RoBERTa, have outperformed previous baselines, which are based on classical and deep learning methods. We now undertake an analysis of the results and compare them to the previous baselines.

It might seem at first glance that the improvement in the Davidson tweets dataset is insignificant for Transformer models and as compared to other datasets, but this can be explained through evidence provided in the original dataset. The biggest drawback of this dataset is the extreme skewness of the classes. Only 5% of the samples are labeled as hate speech when that is the class that the model is trying to detect and the phenomenon of this severe class imbalance is a huge problem for deep neural network models (Zhang & Luo, 2019). Moreover, the final scores reported by Davidson et al. (2017) showed the result by training the model on the entire dataset, whereas we report the best scores after cross-validation, which keeps aside 20% of the data. Hence, it can be believed that the performance will increase if trained on the entire dataset, which will give it exposure to more training data. Nevertheless, the F1 score was reported to be 0.51 for the "hate" class in Davidson et al. (2017), whereas the highest we were able to achieve was 0.53 on the RoBERTa model—an improvement of two percentage points. Another important factor to consider is the "inter-coder" agreement score provided by CrowdFlower, the service which was used to annotate the dataset. In Davidson et al. (2017), it was mentioned that the intercoder agreement was 92%, which means that only 92% of the time the annotators agreed upon a common label for a particular data sample. Since no model of present times can be expected to have performance better than that, it can thus be considered as the performance upper limit.

On the other hand, there has been quite an improvement in the Reddit dataset over the baseline score and that can be attributed to the experimental setup and adopted approaches mentioned in Qian et al. (2019). Recurrent Neural Networks (RNNs), while being able to process sequential data efficiently, fail to provide optimal performance when the length of the sequences becomes excessively large. They were used in conjunction with pretrained word embeddings obtained from word2vec, which do not take context into account when generating word embeddings. Transformers are well equipped to overcome these problems since they can (1) handle comparatively longer sequences better than RNNs and (2) the multiheaded self-attention is the key strength of the Transformer architecture, which takes into context neighboring tokens for generating contextualized embeddings. This gives the Transformer models a strong edge over the RNN, especially in the Reddit dataset, where the sequences are very long.

Transformer models also made a good improvement on the Gab dataset, though not as pronounced as on the Reddit dataset. This can be attributed to the fact that the CNN model performed quite well in (Qian et al., 2019) because the sequences in the Gab dataset are half the length of those in the Reddit dataset, which reduces the chances of poisoning of the model by artifacts and unwanted noise. CNN model also serves as a better "feature extractor" in comparison to RNN, which paired with a nearly balanced dataset provided quite good results. Hence, making dramatic improvements on the Gab dataset were not a very likely event.

Lastly, the HASOC dataset is the only one where we could not improve our results over existing ones. This is easily explained by the fact that the authors in Roy et al. (2020) used an XLM-RoBERTa, though with a different architecture, and thus we were not expecting any major improvements on that dataset although both

Table 6
Comparison of results achieved by the Transformer models with the best results obtained by original authors. Best results are highlighted in bold

	RoBERTa	XLNet	Author's best
Davidson tweets	**90.84**	90.69	90.00 (SVM)
Reddit	**91.81**	91.48	77.5 (RNN + word2vec)
Gab	**92.21**	91.76	89.6 (CNN + word2vec)
HASOC	89.80	88.94	**90.29** (RoBERTa + custom classification head)

our works used the same model, and it is important to note the significant differences in model architecture and training methodology. The authors of Roy et al. (2020) used a base multilingual RoBERTa model to generate the embeddings for the input data and then fed it to a 12-layer feedforward network for classification. They also used a self-adjusting learning rate where the LR would be automatically decreased to lower values based on the macro F1 score for the current epoch instead of the conventional validation loss, and a model reinstatement mechanism would automatically revert the model to the last best state if the current training epoch performs worse than the last best. Their incorporation of emojis was also quite different from ours as described in Section 4.1. In contrast, we evaluated the performance of the standard Transformer classification models provided by the HuggingFace API, which attaches a two-layer dense network on top of the vanilla Transformer for classification. We also did not make any other custom changes to suit this particular dataset to maintain parity with the other datasets. Nevertheless, our performance was very close to the performance of Roy et al. (2020) on much simpler model training constraints and architecture, thus saving a lot of computational power and model engineering complexity.

Apart from comparing Transformer LMs to other deep learning and classical models, it would be only fair to pit the models against themselves to determine which is better suited for this particular task. From the results, it is evident that RoBERTa beats XLNet in almost all experiments, even if by small margins. This might come across as slightly surprising since RoBERTa inherited a lot from BERT including most of the core pretraining setup and XLNet had beaten BERT in all benchmarks due to its *Permutation Language Modelling* mechanism. But what made the crucial difference is the data on which the models were pretrained. As highlighted in Section II.G, RoBERTa used almost 10 times more training data than XLNet most of which consisted of Internet data such as reviews, comments, and web articles. Therefore, RoBERTa had already encoded a lot more information about the kind of data commonly available on social media platforms than XLNet during its pretraining phase, which made it easier to fine-tune for our task and consequently achieved better scores.

4.7. Limitations

Although Transformer-based LMs are proved to be very efficient in their results and brought several advantages over conventional methods, they have some inherent limitations. The Transformer is a very complex neural network, and like other neural network models, it works as a black box. That is, we can only supply it with the input and receive the corresponding output, but the input is transformed into the corresponding output inside the various layers of the network using some complex process. Although we may have the knowledge of how a certain network functions, it is very difficult to explain the predictions of such complex systems because of the lack of metadata on the input in the intermediary stages of the network. Consequently, we are limited in our power to achieve complete transparency in the systems that we develop, which might give rise to ethical and philosophical issues. Another disadvantage of these Transformer models is also one of its biggest advantages—the rigidity of model architecture. These models, unlike conventional neural network models, cannot be re-engineered at will to suit a specific problem or scenario. Their design and pretraining scheme is very specifically developed following highly complex methods, and if one were to try to modify the structure of the model beyond the permitted specifications, it may result in a catastrophic failure of the entire system. Thus, there is very little flexibility as to what can be done with the model architecture-wise.

5. Conclusion and Future Work

In this study, we have investigated the performance of two Transformer language models, RoBERTa and XLNet, on four different datasets three of which were from different social media platforms and compared them to the existing deep learning and classical architectures. Both the models surpassed the existing baseline scores on three of the four datasets and performed nearly as well on the fourth dataset, which used a RoBERTa model. From our experiments, it can be concluded that these state-of-the-art models, which have revolutionized the domain of natural language processing, prove to be equally effective and promising for the task of hate speech detection. This approach brings some clear advantages—(1) The computation cost for the whole process is largely minimized because these models are pretrained on vast amounts of data. (2) There is no need to expedite effort on developing new architectural designs of models and validate their efficiency for this particular task because the Transformer-based LMs use the same underlying architecture, which is very efficient for the purpose it was developed. (3) Since the effort on model engineering is minimized, it gives a greater scope to procure and experiment with the data, which is usually not the point of focus in such studies. Although we tested only two models in this work, research in this domain is moving at a fast pace and new and better model architectures are being developed. Hence, there may exist other Transformer architectures that perform better than the ones explored in this work, which will require further investigation and experimenting to ascertain. Furthermore, we only considered textual data consisting of tweets and comments, but the problem of multimodal hate speech analysis, where text data are often accompanied by data from other modalities such as images or video, is a tougher challenge. Therefore, Transformer-based models can be used in conjunction with image-processing models to take into account both modalities to make more accurate predictions. Recent application of the Transformer architecture to image processing resulted in Vision Transformer (Dosovitskiy et al., 2020), which has beaten the best CNN models to become the new state of the art. Thus, the performance of conventional CNN models can be compared to Vision Transformers. The multimodal CNN–Transformer combination can also be compared to other conventional multimodal models. Another very important observation we made through this study is that the data used to fine-tune the model is a more important contributor toward the results achieved than the model itself; smaller but quality datasets yielded results as good as large datasets of relatively poorer quality. Given that the Transformer is a very complex model with a carefully designed architecture, which due to its pretraining already encodes most of the necessary information, it might be a good opportunity for the research community to gravitate toward sourcing better data, both in scale and quality, for the fine-tuning phase, and shift their focus to data engineering rather than model engineering in support of the emerging trend of "Data-Centric AI."

Conflicts of Interest

The authors declare that they have no conflicts of interest to this work.

References

Alshalan, R., & Al-Khalifa, H. (2020). A deep learning approach for automatic hate speech detection in the saudi twittersphere. *Applied Sciences, 10*(23), 8614. https://doi.org/10.3390/app10238614.

Brandt, L., & Dean, G. (2021). Gab, a social-networking site popular among the far right, seems to be capitalising on Twitter bans and Parler being forced offline. It says it's gaining 10,000 new users an hour. *Business Insider, 11*.

Dai, Z., Yang, Z., Yang, Y., Carbonell, J., Le, Q., & Salakhutdinov, R. (2019). Transformer-XL: Attentive language models beyond a fixed-length context. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2978–2988. https://doi.org/10.18653/v1/P19-1285.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., . . . , & Houlsby, N. (2020). An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint: 2010.11929.* https://doi.org/10.48550/arXiv.2010.11929

Davidson, T., Warmsley, D., Macy, M., & Weber, I. (2017). Automated hate speech detection and the problem of offensive language. In *Proceedings of the International AAAI Conference on Web and Social Media, 11*(1), 512–515. https://doi.org/10.1609/icwsm.v11i1.14955

Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2019). BERT: Pre-training of deep bidirectional Transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, 1,* 4171–4186. https://doi.org/10.18653/v1/N19-1423.

Gomez, R., Gibert, J., Gomez, L., & Karatzas, D. (2020). Exploring hate speech detection in multimodal publications. In *2020 IEEE Winter Conference on Applications of Computer Vision,* 1470–1478. https://doi.org/10.1109/WACV45572.2020.9093414.

Khan, S. (2021). *BERT, RoBERTa, DistilBERT, XLNet — Which one to use?* Retrieved from: https://towardsdatascience.com/bert-RoBERTa-distilbert-xlnet-which-one-to-use-3d5ab82ba5f8

Kim, T. (2021). *Emoji*. Retrieved from: https://github.com/carpedm20/emoji

Leitner, L. (2021). *Redditcleaner*. Retrieved from: https://github.com/LoLei/redditcleaner

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., . . . & Stoyanov, V. (2019). RoBERTa: A robustly optimized BERT pretraining approach. *arXiv preprint: 1907.11692.* https://doi.org/10.48550/arXiv.1907.11692

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., & Dean, J. (2013). Distributed representations of words and phrases and their compositionality. *Advances in Neural Information Processing Systems, 26.*

Mutanga, R. T., Naicker, N., & Olugbara, O. O. (2020). Hate speech detection in twitter using Transformer methods. *International Journal of Advanced Computer Science and Applications, 11*(9). https://doi.org/10.14569/IJACSA.2020.0110972.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., . . . , & Chintala, S. (2019). Pytorch: An imperative style, high-performance deep learning library. *Advances in Neural Information Processing Systems, 32.*

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., . . . , & Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *The Journal of Machine Learning Research, 12*, 2825–2830.

Pennington, J., Socher, R., & Manning, C. D. (2014). Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing,* 1532–1543. https://doi.org/10.3115/v1/D14-1162.

Qian, J., Bethke, A., Liu, Y., Belding, E., & Wang, W. Y. (2019). A benchmark dataset for learning to intervene in online hate speech. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing,* 4754–4763. https://doi.org/10.18653/v1/D19-1482.

Radford, A., Narasimhan, K., Salimans, T., & Sutskever, I. (2018). *Improving language understanding by generative pre-training*. Retrieved from: https://blog.openai.com/language-unsupervised/

Roy, S. G., Narayan, U., Raha, T., Abid, Z., & Varma, V. (2020). Leveraging multilingual transformers for hate speech detection. In *Proceedings of FIRE 2020 Working Notes, Forum for Information Retrieval Evaluation.*

Sreelakshmi, K., Premjith, B., & Soman, K. P. (2020). Detection of hate speech text in Hindi-English code-mixed data. *Procedia Computer Science*, *171*, 737–744. https://doi.org/10.1016/j.procs.2020.04.080.

Undirwade, A., & Das, S. (2021). Image anonymization using deep convolutional generative adversarial network. *Machine Learning Algorithms and Applications,* 305–330. https://doi.org/10.1002/9781119769262.ch17.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., . . . , & Polosukhin, I. (2017). Attention is all you need. *Advances in Neural Information Processing Systems, 30.*

Wang, S., Maoliniyazi, A., Wu, X., & Meng, X. (2020). Emo2Vec: Learning emotional embeddings via multi-emotion category. *ACM Transactions on Internet Technology*, *20*(2), 1–17. https://doi.org/10.1145/3372152.

Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., . . . , & Rush, A. M. (2020). Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations,* 38–45. https://doi.org/10.18653/v1/2020.emnlp-demos.6.

Yang, Z., Dai, Z., Yang, Y., Carbonell, J., Salakhutdinov, R. R., & Le, Q. V. (2019). Xlnet: Generalized autoregressive pretraining for language understanding. *Advances in Neural Information Processing Systems,* 32.

Zhang, Z., Robinson, D., & Tepper, J. (2018). Detecting hate speech on twitter using a convolution-gru based deep neural network. In A. Gangemi, R. Navigli, M.-E. Vidal, P. Hitzler, R. Troncy, L. Hollink, A. Tordai, & M. Alam (Eds.), *The semantic web,* (pp. 745–760). Springer International Publishing. https://doi.org/10.1007/978-3-319-93417-4_48.

Zhang, Z., & Luo, L. (2019). Hate speech detection: A solved problem? The challenging case of long tail on twitter. *Semantic Web, 10*(5), 925–945. https://doi.org/10.3233/SW-180338.

How to Cite: Mukherjee, S. & Das, S. (2023). Application of Transformer-Based Language Models to Detect Hate Speech in Social Media. *Journal of Computational and Cognitive Engineering*, 2(4), 278–286, https://doi.org/10.47852/bonviewJCCE2022010102