BERT and fastText Embeddings for Automatic Detection of Toxic Speech

Ashwin Geet d'Sa, Irina Illina, Dominique Fohr

BERT and fastText Embeddings for Automatic Detection of Toxic Speech

Ashwin Geet D'Sa, Irina Illina, Dominique Fohr
Université de Lorraine, CNRS, Inria, LORIA, F-54000 Nancy, France

Abstract - With the expansion of Internet usage, catering to the dissemination of thoughts and expressions of an individual, there has been an immense increase in the spread of online *hate speech*. Social media, community forums, discussion platforms are few examples of common playground of online discussions where people are freely allowed to communicate. However, the freedom of speech may be misused by some people by arguing aggressively, offending others and spreading verbal violence. As there is no clear distinction between the terms *offensive, abusive, hate and toxic* speech, in this paper we consider the above mentioned terms as *toxic* speech In many countries, online toxic speech is punishable by the law. Thus, it is important to automatically detect and remove toxic speech from online medias. Through this work, we propose automatic classification of toxic speech using embedding representations of words and deep-learning techniques. We perform binary and multi-class classification using a Twitter corpus and study two approaches: (a) a method that consists extracting word embeddings and then using a DNN classifier. We observed that BERT fine-tuning performed much better.

Index Terms - Natural language processing, classification, deep neural network**,** hate speech.

1. Introduction

Hate speech expresses an antisocial behavior. The topics of the hate can be gender, race, religion, ethnicity, etc. [1].

There is no clear definition of the term *hate speech*. A Committee of Ministers from Council of European Union define hate speech as: "All forms of expression which spread, incite, promote or justify racial hatred, xenophobia, anti-semitism or other forms of hatred based on intolerance, including intolerance expressed by aggressive nationalism and ethnocentrism, discrimination and hostility towards minorities, migrants and people of immigrant origin."[1]. As there is no clear distinction between the terms *offensive, abusive, hate and toxic* speech, in the following of this paper, we will consider the above mentioned terms as *toxic* speech.

Table 1 gives some examples of toxic comments from social media.

She look like a tranny.
You Asian, they will deport you when they see your eyes.
I'm not going to believe any of the stupid rumors I hear about jews being friends of Christians.
We hate niggers, we hate faggots and we hate spics

Table 1: Examples of toxic comments from social media.

Toxic speech can be expressed in different forms. Explicit toxic speech contains offensive words such as *'fuck', 'asshole'*. Implicit toxic speech can be realized by a sarcasm and an irony[2][3]. While explicit toxic speech can be identified using the lexicons that forms the toxic speech, implicit toxic speech is often hard to identify and requires semantic analysis of the sentence. Examples of implicit and explicit toxic speech are shown in Table 2.

Toxic content on Internet platform can create fear, anxiety and threat to individuals. In the case of company or online platform, the company or platform may lose its reputation or the reputation of its product. Failure to moderate these contents may cost the company in multiple ways: loss of users, drop in stocks [2], penalty from legal authority[3], etc.

Explicit toxic speech
You are a real fag aren't you?
Go fuck yourself asswipe!
Haha you are a dumb shit.
Implicit toxic speech
Affirmative action means we get affirmatively second rate doctors and other professionals.
I will remove all your organs in alphabetical order.
She looks like a plastic monkey doll!

Table 2: Examples of explicit and implicit toxic speech.

Most of the online platforms such as social media or the forums, generally cannot be held responsible for the propagating of toxic speech. However, their inability to prevent its use is the reason for the spread of hate. Manual analysis of such content and its moderation are impossible because of the huge amount of data circulating on the internet. An effective solution to this problem would be the automatic detection of toxic comments.

In many countries, online hate speech is an offense and it is punishable by the law. In this case, the social medias are held responsible and accountable if they did not remove hate speech content promptly.

1 https://www.article19.org/data/files/medialibrary/3548/ARTICLE-19-policy-on-prohibition-to-incitement.pdf

2 https://www.telegraph.co.uk/technology/2018/07/27/twitter-stock-sinks-reporting-decline-active-users/

3 https://www.cnet.com/news/german-hate-speech-law-goes-into-effect-on-1-jan/

Automatic detection of toxic speech is a challenging problem in the field of *Natural Language Processing* (NLP). The approaches proposed for automatic toxic speech detection are based on the representation of the text in a numerical form and on the using of some classification models. In the state-of-the-art on this field, word and character n-grams [4], *Term Frequency-Inverse Document Frequency* (TF-IDF), *Bag of Words* (BoW), polar intensity, noun patterns [5] and word embedding are largely used as input features. The notion of word embedding is based on the idea that, semantically and syntactically similar words must be close to each other in an n-dimensional space [8]. *Global Vectors for word representation* (GloVe) [6] and random embeddings as input to DNN classifiers has been compared in [7]. Recently, sentence embeddings [9] and *Embeddings from Language Models* (ELMo) [10] were used as input to classifiers for toxic comment classification. Multi-features based approach combining various lexicons and semantic-based features is presented in [11].

Deep-learning techniques have shown to be very powerful in classifying toxic speech [7]. For example, Convolutional Neural Network (CNN) are able to capture the local patterns in text [12]. Long Short Term Memory (LSTM) model [13] or Gated Recurrent Unit (GRU) model [14] capture the long range dependencies. Such properties are important for modelling toxic speech [7], [15].

In this article, we propose a new methodology to automatically detect toxic speech. We perform toxic speech classification using two powerful word representations: fastText and BERT embeddings. These representations are used as inputs to DNN classifiers, namely CNN and Bi-LSTM classifiers. We study two cases: binary classification and multi-class classification. In the last case, we want to classify toxic speech more finely in *hate speech and abusive speech.* Moreover, we explore the capabilities of BERT fine-tuning on both binary and multi-class classification tasks. We evaluate the proposed approaches on the a Twitter corpus.

The contributions of our paper is as follow:

- We use fastText embeddings and BERT embeddings as input features to CNN and Bi-LSTM classifiers.
- We perform fine-tuning of the pre-trained BERT model.
- We study the classification of comments from two perspectives:
  (a) **binary classification**, where we consider two classes: *non toxic speech* versus *toxic speech (hate speech* and *offensive speech* together);
  (b) **multi-class classification,** where we use three classes *hate speech, offensive speech* and *neither.* This three class classification allows to perform fine-grained distinction between hate and offensive speech within toxic speech.

The rest of the paper is organized as follows. Section 2 describes the word embeddings. Section 3 presents the proposed methodology. Section 4 describes data and the preprossessing description. The results are discussed in section 5.

2. Word embeddings

The main idea of word embeddings is to project words in a continuous vector space. In this space, semantically or syntactically related words should be located in the same area. An important advantage of word embedding is that their training does not require a labeled corpus.

The embeddings are generally learned from a very huge unlabelled corpus. This training is time consuming and often requires high-level technical conditions (big GPU, large memory, etc). Pre-trained word embeddings are made available via Internet and can be used by researchers from around the world for different NLP tasks. For example, Facebook provided fastText model, Google provided several BERT models for different languages. In this paper, we propose to use these pre-trained embeddings. In the following of this section, we will describe the embeddings used in this study.

fastText embedding: It is an extension of Mikolov's embedding [8]. The fastText approach is based on the skip-gram model, where each word is represented as a bag of character n-grams [16], [17]. A vector representation is associated to each character n-gram; words being represented as the sum of these representations. The word representation is learned by considering a large window of left and right context words. Unlike Mikolov's embeddings, fastText is able to provide an embedding for misspelled word, rare words or words that were not present in the training corpus, because fastText uses character n-gram word tokenization.

BERT embedding: Currently BERT (*Bidirectional Encoder Representations from Transformers*) is one of the most powerful context and word representations [18]. BERT is based on the methodology of *transformers* and uses *attention* mechanism. Attention is a way to look at the relationship between the words in a given sentence [19]. Thanks to that, BERT takes into account a very large left and right context of a given word. It is important to note that the same word can have different embeddings according to the context. For example, the word *bank* can have one embedding when it occurs in the context *the bank account* and a different embedding when it occurs in the context *the bank of the river*. Moreover, BERT model uses word-piece tokenization. For instance, the word *singing* can be represented as two word-pieces: *sing* and *##ing.* The advantage is, that when the word is not in the BERT vocabulary, it is possible to split this word into word-pieces. Thus, it is possible to have embeddings for rare words, like in fastText.

BERT model can be used in two ways:

- for generating the embeddings of the words of a given sentence. These embeddings are further used as input for DNN classifiers;
- for fine-tuning a pre-trained BERT model using a task-specific corpus and further to perform the classification.

3. Proposed methodology

Figure 1 shows the proposed methodology. The general idea is as follow: we use pre-trained embeddings to represent each comment in the continuous space. After this, we use these embeddings as input features for a DNN classifier.

We propose to use the word representations in two ways, ***feature-based*** and ***fine-tuning*** approaches:

- in feature-based approach, two steps are performed. First, each comment is represented as a sequence of words or word-pieces and for each word or word-piece, an embedding is computed using fastText or BERT. Secondly, this sequence of embeddings will form the input to the DNN classifiers, that takes the final decision. We use CNN and Bi-LSTM models as classifiers.
- in fine-tuning approach, everything is done in a single step. Each comment is classified by a fine-tuned BERT model.

We classify each comment as *non toxic* or *toxic speech* for binary classification and *offensive*, *hate speech* or *neither* for multi-class classification.

3.1 Feature-based approaches

For feature-based approaches, we used pre-trained fastText and BERT models to obtain the sequence of embeddings for a given comment. This sequence of embeddings is used as input features to DNN classifiers. The sequence should have a fixed size. For this, we extend the short tweets by zero padding.

fastText model: We use pre-trained fastText embedding model and apply this model to generate one embedding for each word of a given comment. Thanks to the bag of character n-grams model of fastText, every word in a given comment will have an embedding, even out-of-vocabulary and rare words.

BERT model: Word-piece tokenization is performed on the comment and then used as input to a pre-trained BERT model. BERT model provides contextual embedding for the word-pieces.

The obtained embeddings from either fastText or BERT models are then used as input to a DNN classifier.

3.2 Deep Neural Networks classifiers

For the purpose of toxic tweet classification, we use CNN and Bi-LSTM deep neural network classifiers:

- CNN were traditionally used in the application of image processing, and are good at capturing the patterns. Kim [12] demonstrated the efficient use of CNN for Natural language processing on various benchmark tasks.
- Bidirectional LSTM (Bi-LSTM) is a class of RNN models, which overcomes the problem of vanishing gradient problem. Bi-LSTMs are used for sequential processing of the data and are efficient at capturing long-range dependencies.

3.3 BERT fine-tuning

The BERT pre-trained model can be fine-tuned to a specific task. This consists in the adapting of the pre-trained BERT model parameters to a specific task using a small corpus of task specific data. Since BERT is contextual model and pre-trained BERT model is trained on a very huge corpora containing few *toxic speech* or *twitter data*, it will be interesting to fine-tune this model with the toxic and twitter specific data set. For the purpose of classification task, a neural network layer is used on top of fine-tuned BERT model. So, the weights of this layer and the weights of the other layers of the Bert model are trained and fine-tuned correspondingly using task specific data in order to perform the classification task.

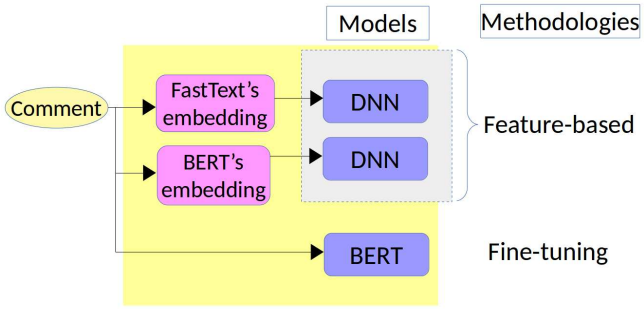

Figure 1: Proposed methodology

4. Experimental setup

4.1 Data set

For the purpose of toxic classification, we used the Twitter corpus [20]. The tweets are collected based on keywords from *hatebase.org* lexicon. The data set contains 24883 tweets and annotations performed by CrowdFlower. Each tweet is annotated by at least 3 annotators. The annotator agreement is 92%. The labels correspond to three classes: *hate speech*, *offensive language* and *neither*, representing 5.7%, 77.1% and 16.7% respectively. Thus, this data set is an unbalanced data set. Table 3 gives the statistics of the data set after pre-processing.

We followed the 5-fold cross validation procedure as in [20]. We used 70% of data as training, 20% as test set and 10% as development set. The development set is used to choose the hyper-parameters. The test set is used to evaluate the proposed approaches.

In our experiments, during the **binary classification** we merged *hate-speech* and *offensive speech* together in the single class (*toxic speech)*. Thus, we have *toxic speech class* and *non toxic speech* class. For **multi-class classification** we use the three classes and the labels provided with the data set: *hate speech*, *offensive speech* and *neither*.

4.2 Text pre-processing

Most of the text classifiers results depend on the way the input text is pre-processed. For both, fastText and BERT embeddings, we decided to remove the numbers and all the special characters except *'!', '?', ',', '.'* and apostrophe.

We also performed tweet specific pre-processing. We removed user names (words beginning with symbol '@'). and the word 'RT', indicating *re-tweet*. We split hast-tags in multiple words. For example, *#KillThemAll* is split into *Kill Them All*.

	Hate speech	*Offensive speech*	*Neither*	*Total*
Nbr of tweets	1430	19190	4163	24783
Corpus size (word count)	19.6k	259.5k	62.1k	341.2k
Nbr. of unique words	3.7k	16.2k	9.9k	21.2k
Average nbr. of words per tweet	13.7	13.5	14.9	13.8

Table 3: Statistics of Twitter data set after pre-processing. k denotes thousand.

4.3 Embedding models

- **fastText embedding:** the model is provided by Facebook[4] and pre-trained on *Wikipedia 2017*, *UMBC webbase* and *statmt.org news* data sets with total 16B tokens. The embedding dimension is 300, the vocabulary is 1M words.
- **BERT model:** In our work, we used BERT-base-uncased word-piece model (for English), provided by Google[5] and pre-trained on *BookCorpus* and *Wikipedia* corpora. The model has 12 stacked transformer encoder layers, with 24 attention heads. The embedding dimension is 768, the number of word-pieces is 30k.

4.4 DNN configurations

We perform the classification experiments with different hyper-parameters and choose the final configuration based on the best performance obtained on the development set. The best model configurations are detailed below.

For Bi-LSTM, we used one or two bidirectional LSTM layers with varying LSTM units (between 50 and 128) followed by one or two dense layers with 64 and 256 dense units in the first dense layer and 16 and 64 dense units in the second layer. For CNN we have used either one or two layers (filter size between 3 and 5), and used between 16 and 64 units, followed by two dense layers having 64 and 256 dense units in the first dense layer and 16 and 64 dense units in the second layer. The dense units use *Rectified Linear Unit* activation (ReLU), while the final output neuron uses *sigmoid* activation. We use a varying dropout upto 0.2. We use l2 regularization. The models are trained using Adam optimizer with learning rate of 0.001. For BERT fine-tuning we used maximum sequence length 256, batch size 16, learning rate $2 \cdot 10^{-5}$ and 3 epochs.

4 https://fasttext.cc/docs/en/english-vectors.html

5 https://github.com/google-research/bert

We evaluate the performance of our approaches in terms of macro-average F1-measure. **F1-measure** is a statistical measure to analyze classification performance. This value ranges between 0 and 1, where 1 indicates the best performance. F1-measure is calculated as follow:

$$F1 = \frac{2 * (precision * recall)}{(precision + recall)}$$

where ***precision*** is the ratio between number of samples correctly predicted as class A and total number of samples predicted as class A by the classifier; ***recall*** is the ratio between number of samples correctly predicted as class A and total number of samples that should be predicted as class A.

Macro-average F1-measure provides the arithmetic mean of F1-measures of all classes:

$$macro\,F1 = \frac{1}{C} \sum_{i=1}^{C} F1_i$$

where, C is the total number of classes.

For each experiment, we compute an average macro-average F1-measure obtained from the 5-folds test sets.

5. Results and discussion

Table 4 gives the macro average F1 results for binary classification task using Bi-LSTM and CNN classifiers with fastText and BERT embeddings as input features. Table 5 presents the macro-averaged F1-measure results for multi-class classification task.

From table 4 and table 5, it can be observed that both fastText and BERT embeddings provide nearly the same results. Among the classifiers, Bi-LSTM performs slightly better than CNN. The performance of binary classification, presented in table 4, is better than multi-class classification performance, given in table 5. This can be explained by the fact that it is difficult to distinguish between hate speech and offensive speech.

Finally, BERT fine-tuning performs better than feature-based approaches in both binary as well as in multi-class classification: compared to feature-based approaches, we obtained 63% and 42% of classification error reduction for binary and multi-class classification correspondingly. One reason may be that in the feature-based approach the

A. Feature-based approaches		
	CNN	Bi-LSTM
fastText embedding	91.5	91.9
BERT embedding	90.9	91.9
B. BERT fine-tuning		
BERT fine-tuning	**97.0**	

Table 4: Macro-average F1-measure for different classifiers and different embeddings. Binary classification.

A. Feature-based approaches		
	CNN	Bi-LSTM
fastText embedding	70.9	72.3
BERT embedding	71.9	72.4
B. BERT Fine-tuning		
BERT fine-tuning	**84.0**	

Table 5: Macro average F1-measures for different classifiers and different embeddings. Multi-class classification.

embeddings vectors have not been trained on hate speech or offensive data. On the contrary, the BERT fine-tuning approach is fine-tuned on twitter data to distinguish hate, offensive and non toxic speech and this allow to create more accurate model for toxic speech.

Table 6 and 7 present the confusion matrices between the 3 classes for multi-class classification: table 6 for feature-based Bi-LSTM with BERT embeddings and table 7 forBERT fine-tuning. We can notice that the main confusions occur between hate speech and offensive speech. This suggest that the model is biased towards classifying tweets as less hateful or offensive than the human annotators. This result is close to the results obtained in [20]. The feature-based approach is able to detect only 31% of the hate speech tweets, while Bert fine-tuning achieved 53%. Many fewer tweets are classified as more offensive or hateful than their true category.

True label	hate	31	60	9
	offensive	2	95	3
	neither	3	10	87
		hate	offensive	neither
		Predicted labels		

Table 6: Confusion matrix for feature-based Bi-LSTM with BERT embeddings (in %). Multi-class classification.

True label	hate	53	43	4
	offensive	1	98	1
	neither	1	4	95
		hate	offensive	neither
		Predicted labels		

Table 7: Confusion matrix for BERT fine-tuning (in %). Multi-class classification.

6. Conclusion

In this article, we proposed new approaches for automatic toxic speech detection in the social media. These approaches are based on deep learning classifiers and word embeddings. We have explored the classification from two perspectives-binary classification and multi-class classification. For binary classification we consider toxic speech (hate speech and offensive speech together) and non toxic speech. For multi-class, we considered *hate speech, offensive speech* and *neither*.

We proposed feature-based approaches and fine-tuning of pre-trained BERT model. In feature-based approaches, fastText and BERT embeddings are used as input features to CNN and Bi-LSTM classifiers. Further, we have compared these configurations with fine-tuning of pre-trained BERT model.

We observed that BERT fine-tuning performed much better than feature-based approaches on a Twitter corpus. The main confusions occur between offensive speech and hate speech. In the future work, we want to investigate this problem. Proposed methodology can be used for any other type of social media comments.

7. Acknowledgment

This work was funded by the M-PHASIS project supported by the French National Research Agency (ANR) and German National Research Agency (DFG) under contract ANR-18-FRAL-0005.

8. References

[1] R. Delgado and J. Stefancic, “Hate Speech in Cyberspace”, *Social Science Research Network,* 2014.

[2] Z. Waseem, T. Davidson, D. Warmsley, and I. Weber, “Understanding Abuse: A Typology of Abusive Language Detection Subtasks,” in *Proceedings of the First Workshop on Abusive Language Online*, 2017, pp. 78–84.

[3] L. Gao, A. Kuppersmith, and R. Huang, “Recognizing Explicit and Implicit Hate Speech Using a Weakly Supervised Two-path Bootstrapping Approach,” in *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, 2017, pp. 774–782.

[4] C. Nobata, J. Tetreault, A. Thomas, Y. Mehdad, and Y. Chang, “Abusive Language Detection in Online User Content”, in *Proceedings of the 25th International Conference on World Wide Web*, 2016, pp. 145–153.

[5] M. Wiegand, J. Ruppenhofer, A. Schmidt, and C. Greenberg, “Inducing a Lexicon of Abusive Words–a Feature-Based Approach,” in *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies,* Volume 1, 2018, pp. 1046–1056.

[6] J. Pennington, R. Socher, and C. Manning, “Glove: Global Vectors for Word Representation”, in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2014, pp. 1532–1543.

[7] P. Badjatiya, S. Gupta, M. Gupta, and V. Varma, “Deep Learning for Hate Speech Detection in Tweets”, in *Proc. 26th Int. Conf. World Wide Web Companion - WWW 17 Companion*, pp. 759–760, 2017.

[8] T. Mikolov, K. Chen, G. Corrado, and J. Dean, “Efficient Estimation of Word Representations in Vector Space”, *ArXiv13013781 Cs*, 2013.

[9] V. Indurthi, B. Syed, M. Shrivastava, N. Chakravartula, M. Gupta, and V. Varma, “FERMI at SemEval-2019 Task 5: Using Sentence embeddings to Identify Hate Speech Against Immigrants and Women in Twitter”, in *Proceedings of the 13th International Workshop on Semantic Evaluation*, 2019, pp. 70–74.

[10] M. Bojkovský and M. Pikuliak, “STUFIIT at SemEval-2019 Task 5: Multilingual Hate Speech Detection on Twitter with MUSE and ELMo Embeddings”, in *Proceedings of the 13th International Workshop on Semantic Evaluation*, 2019, pp. 464–468.

[11] S. Almatarneh, P. Gamallo, and F. J. R. Pena, “CiTIUS-COLE at SemEval-2019 Task 5: Combining Linguistic Features to Identify Hate Speech Against Immigrants and Women on Multilingual Tweets”, in *Proceedings of the 13th International Workshop on Semantic Evaluation*, 2019, pp. 387–390.

[12] Y. Kim, “Convolutional Neural Networks for Sentence Classification”, in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2014, pp. 1746–1751.

[13] A. Baruah, F. Barbhuiya, and K. Dey, “ABARUAH at SemEval-2019 Task 5 : Bi-directional LSTM for Hate Speech Detection”, in *Proceedings of the 13th International Workshop on Semantic Evaluation*, 2019, pp. 371–376.

[14] K. Cho, B. V. Merrienboer, C. Gulcehre , D. Bahdanau, F. Bougares , H. Schwenk, Y. Bengio. “Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation,” in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2014, pp. 1724–1734.

[15] S. Bodapati, S. Gella, K. Bhattacharjee, Y. Al-Onaizan “Neural Word Decomposition Models for Abusive Language Detection”. In *Proceedings of the Third Workshop on Abusive Language Online,* pp. 135-145, 2019.

[16] A. Joulin, E. Grave, P. Bojanowski, M. Douze, H. Jégou, and T. Mikolov, “FastText.zip: Compressing Text Classification Models”, *ArXiv Prepr. ArXiv161203651*, 2016.

[17] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, “Enriching Word Vectors with Subword Information,” *Transactions of the Association for Computational Linguistics*, vol. 5, no. 1, pp. 135–146, 2017.

[18] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “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,* Volume 1, 2019, pp. 4171–4186.

[19] A. Vaswani *et al.*, “Attention is All You Need,” in *Advances in Neural Information Processing Systems*, 2017, pp. 5998–6008.

[20] T. Davidson, D. Warmsley, M. Macy, and I. Weber, “Automated Hate Speech Detection and the Problem of Offensive Language,” in *Eleventh International Conference on Web and Social Media*, 2017.