Online abuse detection: the value of preprocessing and neural attention models

Dhruv Kumar
University of Waterloo
d35kumar@uwaterloo.ca

Robin Cohen
University of Waterloo
rcohen@uwaterloo.ca

Lukasz Golab
University of Waterloo
lgolab@uwaterloo.ca

Abstract

We propose an attention-based neural network approach to detect abusive speech in online social networks. Our approach enables more effective modeling of context and the semantic relationships between words. We also empirically evaluate the value of text pre-processing techniques in addressing the challenge of out-of-vocabulary words in toxic content. Finally, we conduct extensive experiments on the Wikipedia Talk page datasets, showing improved predictive power over the previous state-of-the-art.

1 Introduction

Over the past few years, there has been increasing attention devoted to the problems of abusive language and hate-based activity in online social networks, with big social media platforms feeling the pressure from governments to perform some moderation of their activities. The AI research community has begun to design automated methods to detect instances of hate speech in these networks, with a primary approach proposing the use of Natural Language Processing (NLP) to perform document classification (Schmidt and Wiegand, 2017).

A major challenge to performing this task is the intentional word and phrase obfuscation done by users to avoid detection (Nobata et al., 2016). Examples such as 'sh*t', '1d10t' and 'banmuslim' are human-readable but difficult to detect using algorithms that rely on keyword spotting. Obfuscation makes context modeling, a challenging problem in NLP, even harder. For example, in the sentences "You feminist cnt" and "I cnt understand this", 'cnt' is used as a shorthand. However, without considering the context, it is difficult to tell whether 'cnt' represents 'cannot' or a derogatory remark.

Early work in hate speech detection used classifiers such as Support Vector Machines and Logistic Regression, with features such as word n-gram counts and the number of insult words (Greevy and Smeaton, 2004; Kwok and Wang, 2013; Mehdad and Tetreault, 2016). With the recent success of deep learning models in solving a variety of classification problems, they have also become the state-of-the-art in detecting abusive speech.

In this paper, we make the following contributions towards detecting hate speech in social networks.

1. We propose the use of attention based deep learning models, the first being the usual word attention layer and the second being a self-targeted co-attention layer that considers the semantic relationships between word pairs.

2. We examine the value of text pre-processing techniques to reduce the number of out-of-vocabulary (OOV) words. We find that preprocessing not only helps to improve the accuracy of existing models, but also improves the proposed attention models.

Our solution addresses the main challenges in detecting abusive content: capturing context to identify important words when making classification decisions, which we achieve through the attention models, and out-of-vocabulary words, which we deal with through preprocessing. Altogether, we improve classification accuracy over the previous state of the art on the Wikipedia Toxicity, Personal Attack, and Aggression datasets (Wulczyn et al., 2017).

In the remainder of this paper, Section 2 discusses related work, Section 3 presents our pre-processing method, Section 4 discusses our deep learning models and the baseline, Section 5

Proceedings of the 10th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis, pages 16–24
Minneapolis, June 6, 2019. ©2019 Association for Computational Linguistics

presents experimental results, and Section 6 concludes the paper with directions for future work.

2 Related Work

Among the first to study the problem of online abuse detection were Yin et al. (2009) who focused on harassment on the Web. They used a linear Support Vector Machine (SVM) with character and word n-grams, sentiment, and contextual features of the document (cosine similarity of neighbouring text). One of the first to study hate speech were Djuric et al. (2015) who used comments from the Yahoo Finance website. They learned text embeddings using the neural language model from Le and Mikolov (2014) and used them to train a binary classifier. Nobata et al. (2016) trained a regression model on multiple features such as word and character n-grams, as well as linguistic (e.g., number of hate blacklist words), syntactic (part-of-speech tags) and distributional semantic features (e.g., embeddings). They showed that although best performance was achieved when all features were used together, character n-grams were the most important.

Waseem and Hovy (2016) released a dataset containing 16,000 tweets that were manually labeled as either racist, sexist or clean. They used a Logistic Regression classifier and showed that character n-grams were important features. Working with the same dataset, Badjatiya et al. (2017) were one of the first to apply deep learning. They used a Gradient-Boosted Decision Tree (GDBT) on word embeddings learned using a Recurrent Neural Network (RNN). Also, Gambäck and Sikdar (2017) used Convolutional Neural Networks (CNN) on the same dataset. Furthermore, Park and Fung (2017) used the following two-step process. They first detected whether a tweet was abusive or not, and then, using another classifier, further classified the tweet as racist or sexist. They used a HybridCNN model, which is a variant of CNN that uses both words and characters to make classification decisions.

Wulczyn et al. (2017) created three datasets from the English Wikipedia Talk Page: one annotated for personal attacks, one for toxicity, and one for aggression. Their best model was a multilayer perceptron trained on character n-gram features. Pavlopoulos et al. (2017) then improved the accuracy on the toxicity and personal attack datasets using RNNs. In addition, they released another dataset, with 1.6 million manually annotated user comments from the Greek Sports Portal (Gazzetta), and embeddings trained on this dataset. Mishra et al. (2018) generated embeddings for OOV words and used them with RNNs and character n-gram features on the Twitter and the Wikipedia datasets. Lee et al. (2018) analyzed another dataset released by Founta et al. (2018), which also consists of tweets manually annotated into various categories of abusive speech.

Recently, attention models have been shown to be effective in various areas of NLP such as machine translation (Luong et al., 2015), question answering (Seo et al., 2016), entailment classification (Rocktäschel et al., 2015), and document classification (Yang et al., 2016). The idea is that different words in a sentence can have different relative importance. Attention models help identify this by assigning importance scores to words. However, there has been limited effort on exploring the utility of these models for detecting online abusive speech. One study on moderating user comments (Pavlopoulos et al., 2017) experimented with adding an attention module, and showed benefits for the Greek Sports Portal dataset, but found little improvements for the Wikipedia dataset. Another effort focused on Twitter (Lee et al., 2018) was also unable to see improvements, but since attention works better on longer sentences, this result is not surprising.

Co-attention is a specific kind of attention mechanism that was introduced for the task of Question Answering (QA) to measure the relationship between all pairs of context and query words (Seo et al., 2016; Xiong et al., 2016). Since hate speech detection takes single sentences as input, self targeted co-attention may be more appropriate, whose aim is to model a sentence against itself, and thus extract the relative importance of every word pair. We also take inspiration from a recent work by Tay et al. (2018) who applied a co-attention model for sarcasm detection. The modest effort to date with attention models for abuse detection and the limited success of these efforts provides an important opportunity for us to present a novel approach, with more effective results.

3 Preprocessing Methods

Social media content is noisy: it may contain shorthand, typos, emojis, etc. Furthermore, abusive content may be intentionally obfuscated to avoid detection. However, we found previous work to be inconsistent with the use of text preprocessing techniques and with quantifying their effects. Some approaches, such as Mishra et al. (2018); Pitsilis et al. (2018), applied minimal preprocessing, similar to our baseline defined below. Others, such as Zhang et al. (2018), used additional methods including Twitter tokenizers and normalizing Twitter hashtags. In our view, text preprocessing can be an important factor in improving hate speech detection capabilities and therefore we take on the task of measuring its value. Below, we detail the baseline and the preprocessing technique we use in this work.

std-approach serves as our baseline. It comprises of lower casing the text, light text cleaning such as handling elongated text (e.g., coverting 'yaaaay' to 'yaay'), and removing whitespaces and stop words. For tokenization, we use the standard nltk text tokenizer[1].

adv-approach consists of the following steps:

- **AT**: We replaced the nltk tokenizer with an advanced tokenizer[2] (Baziotis et al., 2017), designed for noisy data from social networks. It handles common emoticons, URLs, dates, and hashtags. It also labels common censored words such as sh*t but does not modify their form, e.g., it converts 'sh*t' to 'sh*t (censored)'.

- **SW**: We remove punctuation and words appearing only once. We also limit words to 50 characters (trimming longer words down to 50 characters). However, in contrast to the std-approach, we do not remove stop words since we observed that pronouns play an important role in hate speech detection (details in Section 5).

- **SC**: We employ a state-of-the-art spelling correction tool (Ekphrasis) to remove typos and obfuscation. However, we only use this tool on words whose suggested corrections are present in our pre-trained word embedding vocabulary (details in Section 5).

- **WS**: We then deal with concatenated words such as 'stupidperson' or 'stupid_person'. The first case can be handled by replacing dashes with spaces and then applying a spell checker on the segmented words to identify typos. For the second case, we use a word segmenter library (Ekphrasis). Again, we only consider the result of the segmenter if each separated word is part of our embedding vocabulary. As a result, adv-approach cannot identify phrases composed of incorrectly spelled words such as 'bnamuslmis'.

4 Deep Learning Methods

In this section, we describe the deep learning methods for hate speech detection, including baselines and attention models.

4.1 BiRNN

Our first baseline is the Hidden State (HS) method adopted from Mishra et al. (2018). We refer to our modified version as BiRNN. Instead of using two layers of RNNs, we use a single-layer Bidirectional RNN (BiRNN) since it gave better results. A BiRNN consists of two RNNs, one operating on the sequence of words in the forward direction like a standard RNN, and the other going backwards. Each cell in a BiRNN is a GRU (Gated Recurrent Unit) (Chung et al., 2014). The model accepts a sentence as input. First, the embedding layer converts each word into a low dimensional embedding vector, producing a sequence of word embeddings $W \in R^{(n \times d)}$, where n and d denote the number of words in the sentence and the embedding dimension size, respectively. Thus, the sentence can be denoted by $(w_1, w_2, ..., w_n)$ where w_i represents the i_{th} word through its embedding vector. This is given as input to the BiRNN, which creates two sets of hidden states, $\overrightarrow{h}$ and $\overleftarrow{h}$. We concatenate these two hidden states to obtain the final hidden state vector $h \in R^{(n \times 2m)}$ represented as $(h_1, h_2, ..., h_n)$, where m is the number of hidden dimensions of each GRU cell. Finally, we perform a max-pooling over time operation (Collobert and Weston, 2008) over the hidden states to obtain the final representation vector.

4.2 Attn

Our second model is a variant of the attention mechanism originally proposed by Yang et al. (2016) and used by Pavlopoulos et al. (2017) on the same Wikipedia Talk datasets that we use in our experimental evaluation[3]. The intuition be-

[1] https://www.nltk.org/

[2] https://github.com/cbaziotis/ekphrasis

[3] However, they did not see any improvements in their results. We suspect this was because their attention model was

hind this attention model is that since not all words contribute equally to a sentence, the model should learn to focus on the important words. This mechanism is applied over the hidden states $(h_1, h_2, ..., h_n)$ of the BiRNN as shown below.

$$u_i = (ReLU(W_w h_i + b_w))$$

$$a = \mathit{Softmax}(u_i^T u_w)$$

$$v = \sum_{i=1}^{n} h_i a_i$$

Here, $W_w \in R^{(2m \times p)}$, $b_w \in R^{(p)}$ and $u_w \in R^{(p \times n)}$ is a context length vector, where m is the number of hidden dimensions of each GRU cell, p is a hyperparameter, and $ReLU$ is a rectified linear unit describing the activation function. All of these weights are learned during the training process. Thus, we obtain the attended hidden state vector v, which is given to the dense layer.

4.3 Co-Attn

Finally, we consider a co-attention model inspired by recent work on sarcasm detection (Tay et al., 2018). However, we propose several modifications. As shown in Figure 1, the model is composed of a co-attention module and a BiRNN. The idea behind co-attention is to learn the semantic relationship between each word pair in the sentence whereas the BiRNN learns the long-range dependencies in the sentence.

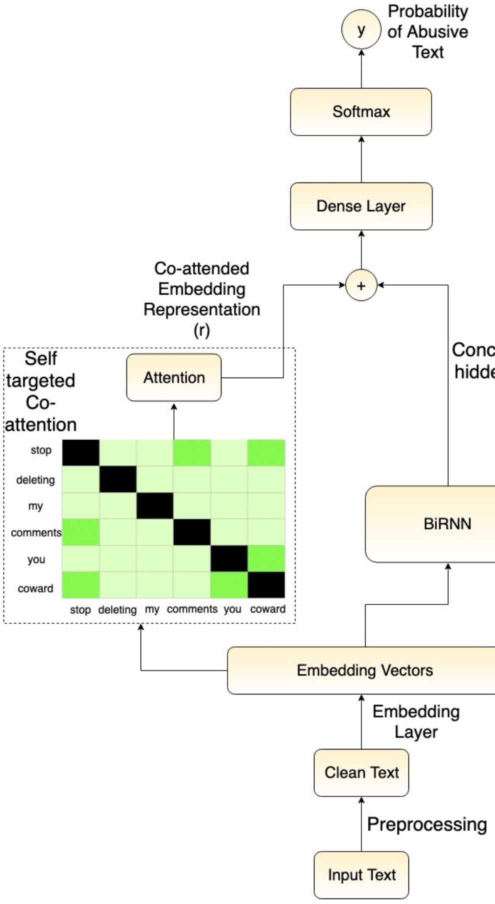

Figure 1: Structure of the Co-Attn model. (Best viewed in color)

We apply the co-attention layer directly on the embedding vectors (we also tested it over the outputs of the BiRNN but obtained worse accuracy). We generate a similarity matrix $S \in R^{(n \times n)}$ to learn the relationships between words, where s_{ij} denotes the score between words e_i and e_j. Our similarity matrix is as follows:

$$s_{ij} = WEW^T$$

where $E \in R^{(d \times d)}$ is a learnable weight matrix, and, as mentioned earlier, $W \in R^{(n \times d)}$ is the word embedding matrix, where n and d denote the length of sentence and embedding dimension size, respectively. We also mask the values in S where $i == j$, so the similarity of a word with respect to itself is not considered. Next, we apply a row-wise average pooling operation to S (as compared to max pooling that was originally proposed), which is followed by a Softmax to learn the attention vector a:

$$a = \mathit{Softmax}(avg_{row}(S))$$

where $a \in R^{(n)}$ represents the learned attention weights. Then the attention vector is used to learn the weighted representation $r \in R^{(d)}$ of W, given by the equation below.

$$r = \sum_{i=1}^{n} w_i a_i$$

Now, instead of learning only from the output of the final hidden state of the BiRNN, the classification layer learns from the joint representation of the co-attended embedding representation (r) and the BiRNN last hidden state vector (h_n), as shown below:

$$f = (ReLU(W_f([r; h_n]) + b_f))$$

deeper than the one we propose and may have led to overfitting.

where $W_f \in R^{((d+2m)\times m)}$ and $b_f \in R^m$. The embedding representation captures relationships between words while the BiRNN captures the sequential information within the sentence.

5 Experiments

For consistency with previous work, our experiments are based on the recently released Wikipedia datasets: Toxicity (W-Tox), Personal Attack (W-At) and Aggression (W-Ag) (Wulczyn et al., 2017). W-Tox contains 159,686 records, while W-At and W-Ag both contain 115,864 records each. These datasets were created by having annotators from the Crowdflower platform label Wikipedia Talk Page comments as toxic or not, personal attack or not, and aggressive or not, respectively. Each comment was judged by multiple annotators, and, in this work, we take the majority vote as the class label. This gives us a binary classification problem. Roughly 10 percent of the comments in each dataset are labelled as toxic, personal attacks or aggressive. For a fair comparison to Mishra et al. (2018), we use a 60:40 training-testing split.

Following Mishra et al. (2018), we use 300-dimensional Glove (Pennington et al., 2014) embedding vectors and we further tune them during training via back-propagation. We create embedding vectors for OOV words with random values in the range ± 0.25. We use 175 as the length of the sequence and we use cross-entropy loss with the Adam optimizer (Kinga and Adam, 2015), with an initial learning rate of 0.001 and L2 regularization of 10^{-6}. Each GRU cell has a hidden dimension size of 150. We experimented with batch sizes of 128, 200 and 256. We implemented all the models in Pytorch (Paszke et al., 2017) and we use the sigmoid output layer in all the models. Our source code is available at `https://github.com/ddhruvkr/Online_Abuse_Detection`

We first evaluate the two methods of preprocessing from Section 3, std-approach and adv-approach. We then evaluate the models from Section 4. To measure the accuracy of the models, we report macro (i.e., average) F1 scores over both classes (labelled “Overall” below) as well as the (micro) F1 scores for just the toxic classes (defined in the standard way, as a harmonic mean of precision and recall). In some experiments, we also report precision (P) and recall (R) individually. For each method, we repeat the experiments five times and report the average.

Method	W-Tox	W-At / W-Ag
std-approach	13617	10703
adv-approach	3418	2755

Table 1: OOV counts after applying standard and advanced pre-processing techniques.

Method	Overall	Toxic
W-Tox		
std-approach	88.76	79.58
std-approach + AT	89.05	80.19
std-approach + SW	88.95	80.04
std-approach + WS + SC	88.93	79.94
adv-approach	89.47	81.02
W-At		
std-approach	87.08	77.09
std-approach + AT	87.53	77.89
std-approach + SW	87.71	78.27
std-approach + WS + SC	87.41	77.71
adv-approach	88.03	78.89
W-Ag		
std-approach	86.45	76.15
std-approach + AT	86.71	76.63
std-approach + SW	86.86	77.01
std-approach + WS + SC	86.64	76.48
adv-approach	87.22	77.59

Table 2: Overall and toxic F1 score after applying various preprocessing techniques using the BiRNN baseline model.

5.1 Impact of Pre-Processing

We first compare the OOV word count in the data after the simple preprocessing method (std-approach) to after applying additional preprocessing (adv-approach). Table 1 compares the OOV word count after applying the two preprocessing approaches on the three tested datasets. Our advanced preprocessing method reduces the number of OOV words by a factor of 4.

To assess the impact of the different preprocessing steps from Section 3 on classification accuracy, Table 2 shows the Overall average F1 scores and the toxic class F1 scores for the BiRNN model (baseline model). We test the standard approach, the standard approach plus the advanced tokenizer (AT), the standard approach plus punctuation and rare word removal, and stopwords added back (SW), the standard approach plus spellchecking (SC) and segmenting concatenated words (WS),

Method	**W-Tox**	**W-At**	**W-Ag**
Context HS+CNG*	89.35	87.44	-
BiRNN	89.47 ± 0.18	88.03 ± 0.20	87.22 ± 0.23
Attn	89.65 ± 0.15	88.18 ± 0.11	**87.49** ± **0.22**
Co-Attn	**89.76** ± **0.14**	**88.34** ± **0.08**	87.35 ± 0.16

Table 3: Overall Macro F1 scores in the three datasets. * denotes results taken directly from the original papers.

and the advanced approach, which includes all of AT, SW, SC, and WS. In general, the adv-approach outperforms the std-approach on all three tested datasets. In particular, the inclusion of stopwords (SW), specifically pronouns, contributes the most to improving the performance on the W-At and W-Ag datasets. On the other hand, the advanced tokernizer (AT) is the most important preprocessing step for the W-Tox dataset. Word Segmentation (WS) and spelling correction (SC) also improve the scores for all three datasets.

5.2 Impact of Attention Models

The remainder of our experiments examine the value of neural attention models, Attn and Co-Attn, compared to 1) the baseline BiRNN 2) and a variation of the baseline that also uses character n-gram features in addition to a RNN, abbreviated Context-HS+CNG. (Mishra et al., 2018). We include Context-HS+CNG because it is the previous state-of-the-art model on our datasets.

First, to compare overall performance, Table 3 shows the overall macro F1 scores of each tested method on the three datasets. We take the scores of Context HS+CNG directly from the original papers (they did not test it on W-Ag, so we omit this number). Overall, we observe that the baseline model BiRNN with text pre-processing already performs better than the previous state-of-the-art. Applying the attention mechanism (Attn) improves the scores, and the Co-Attn model is even better than Attn on W-Tox and W-At.

In addition to reporting the average macro F1 scores, Table 3 also includes the standard deviation over the five experimental runs. In addition to having the highest scores on W-Tox and W-At, Co-Attn also has the lowest standard deviation.

To obtain further insight into the performance on the minority (toxic, personal attack or aggression) class, we show the micro precision (P), recall (R) and F1 scores for the minority class in Table 4. The attention models outperform the baselines in terms of recall and F1, but not precision. The Co-attn model gives the best F1 score for the W-Tox dataset, improving it by close to one point over the previous state-of-the-art (Context-HS+CNG). For the W-At dataset, Co-Attn also has the highest F1 score, improving the baseline by 1.8 points. For the W-Ag dataset, the Attn model improves the BiRNN baseline by about 0.6 points. Using a paired t-test, we found that the differences between BiRNN and Co-Attn for the W-Tox and W-At datasets and between BiRNN and Attn for the W-Ag dataset are statistically significant using a p value of 0.05.

Method	**P**	**R**	**F1**
W-Tox			
Context HS+CNG*	**85.42**	76.17	80.53
BiRNN	83.49	78.69	81.02
Attn	83.57	79.04	81.24
Co-Attn	83.67	**79.42**	**81.49**
W-At			
Context HS+CNG*	81.39	74.28	77.67
BiRNN	**83.43**	74.81	78.89
Attn	82.28	76.40	79.23
Co-Attn	81.42	**77.62**	**79.47**
W-Ag			
Context HS+CNG*	-	-	-
BiRNN	**82.32**	73.37	77.59
Attn	81.57	**75.13**	**78.22**
Co-Attn	81.8	74.55	78.01

Table 4: Micro precision, recall and F1 scores for toxic/personal attack/aggression classes.

5.3 Interpretability

A useful feature of attention mechanisms is that they can help interpret the classification decisions made by the models. To do so, we analyze the representations formed by the attention layers. In Table 5, we consider five comments marked as personal attacks in the W-At dataset. We examine examples where both Attn and Co-Attn predicted the correct label and where their prediction

Model	Prediction	Confidence (in%)	Sentence
Attn	Attack	85.13	stop deleting my comments you coward
Co-Attn	Attack	92.42	stop deleting my comments you coward
Attn	Attack	59.04	you queer boy stop messing with my edits
Co-Attn	Attack	82.87	you queer boy stop messing with my edits
Attn	Non-Attack	71.41	hey queer boy stop messing with my edits
Co-Attn	Attack	64.39	hey queer boy stop messing with my edits
Attn	Attack	77.41	thanks for testing my resolution not to refer to anyone as douchebag
Co-Attn	Attack	74.87	thanks for testing my resolution not to refer to anyone as douchebag
Attn	Attack	65.11	thanks for testing my resolution not to refer to anyone as douche bag
Co-Attn	Non-Attack	50.25	thanks for testing my resolution not to refer to anyone as douche bag

Table 5: Visualization of attention maps, predicted class, and the confidence percentage of the two attention models on personal attack (W-At) comments.

was incorrect. We highlight words found to be important (darker shading means the word was more important), and we show the confidence percentage scores, which represent the probability of the class predicted by the models.

For the first sentence, both models give an accurate prediction. The Attn model captures the relationship between “you” and “coward” whereas the Co-Attn model focuses on the word “stop” in addition to “coward”. In general, we observed that the Attn model relied heavily on pronouns. We see an example of this in the next two sentences.

For the second sentence, both models correctly predicted the class. The Attn model relies on “you” and “queer”. In the third sentence, we replace the word “you” with “hey”, and we see that the Attn model incorrectly labels the sentence as not a personal attack. On the other hand, the Co-Attn model is still able to predict the label correctly.

The next two sentences demonstrate where the Co-Attn model breaks. In the fourth sentence, both models are correct in their predictions. However, the Attn model mainly attends to the word “douchebag” whereas Co-attn observes the interaction between the words “anyone” and “douchebag”. However, when we modified the sentence by splitting the word “douchebag” into two (last sentence), the Co-Attn model attends to both “anyone” and “bag” along with the word “douche”. This results in the model being indecisive and incorrectly predicting that the label is not a personal attack. The confidence score of 50.25% further confirms that the model is uncertain of its prediction. On the other hand, the Attn model still correctly predicts the class as it only focuses on the word “douche”. In general, we found that the Co-attn model was able to capture more interactions between words as compared to the Attn model.

6 Conclusions and Future Work

In this paper, we demonstrated the utility of attention models in detecting online abusive speech. We also showed the importance of reducing the number of out-of-vocabulary words through preprocessing techniques. Our experimental results showed that combining text processing with attention mechanisms, both of which aim to filter out as much noise as possible, is more effective than the previous state of the art, especially at predicting the minority (toxic) class.

In future work, we will investigate alternative spell checkers. In the context of hate speech detection, a problem with standard spell checkers is with their handling of profanity. For example, “sh*t” is corrected to “shot” and “b*tch” to “batch”. Recent work on context-sensitive spelling correction may be a good starting point for this extension (Gong et al., 2019), although it is not clear if intentional obfuscation should be corrected since it can be a strong indicator of hate speech.

We also plan to investigate the performance of our preprocessing and attention methods on other datasets such as Twitter and Facebook (Waseem and Hovy, 2016; Waseem, 2016; Kumar et al., 2018). As mentioned by Mishra et al. (2018), the Wikipedia datasets that we used in this paper have more standard language and less obfuscation than Twitter datasets. Thus, we expect preprocessing to be important for those datasets as well. We will also study the importance of different preprocessing steps when combined with contextualized character embeddings such as ELMo (Peters et al., 2018).

Another interesting direction for future work is to explore *adversarial training* in hate speech detection. This concept originated in the field of computer vision, and refers to the practice of adding noise to training data so as to make the model resistant to noise in test data (Goodfellow et al., 2015). For example, in computer vision, it was observed that when some calculated noise was added to the training data of an image classification model, the model made an incorrect classification decision even though there was no change to a human eye. It can be argued that intentional obfuscation of hate comments affects hate speech classifiers in a similar way. Recent work found that adversarial training does not completely mitigate these issues in hate speech detection and that character level features are more robust than word level features (Gröndahl et al., 2018). However, more work can be done to explore the potential of this idea.

Finally, Schmidt and Wiegand (2017) point out that little research has been done in the field of hate speech detection in languages other than English. They mention that hate speech could have strong cultural implications and therefore advancing the area of multi-lingual hate speech detection is important. They further state that it remains to be seen that how successful techniques in detecting hate speech in English perform when applied to different languages.

Acknowledgements

We thank Michael Cormier, Mount Allison University, for his input on this paper. We also thank the anonymous reviewers for their valuable comments and suggestions.

References

Pinkesh Badjatiya, Shashank Gupta, Manish Gupta, and Vasudeva Varma. 2017. Deep learning for hate speech detection in tweets. In *Proceedings of the 26th International Conference on World Wide Web Companion*, pages 759–760. International World Wide Web Conferences Steering Committee.

Christos Baziotis, Nikos Pelekis, and Christos Doulkeridis. 2017. Datastories at semeval-2017 task 4: Deep lstm with attention for message-level and topic-based sentiment analysis. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 747–754, Vancouver, Canada. Association for Computational Linguistics.

Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*.

Ronan Collobert and Jason Weston. 2008. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th international conference on Machine learning*, pages 160–167. ACM.

Nemanja Djuric, Jing Zhou, Robin Morris, Mihajlo Grbovic, Vladan Radosavljevic, and Narayan Bhamidipati. 2015. Hate speech detection with comment embeddings. In *Proceedings of the 24th international conference on world wide web*, pages 29–30. ACM.

Antigoni Maria Founta, Constantinos Djouvas, Despoina Chatzakou, Ilias Leontiadis, Jeremy Blackburn, Gianluca Stringhini, Athena Vakali, Michael Sirivianos, and Nicolas Kourtellis. 2018. Large scale crowdsourcing and characterization of twitter abusive behavior. In *Twelfth International AAAI Conference on Web and Social Media*.

Björn Gambäck and Utpal Kumar Sikdar. 2017. Using convolutional neural networks to classify hatespeech. In *Proceedings of the First Workshop on Abusive Language Online*, pages 85–90.

Hongyu Gong, Yuchen Li, Suma Bhat, and Pramod Viswanath. 2019. Context-sensitive malicious spelling error correction. *arXiv preprint arXiv:1901.07688*.

Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. 2015. Explaining and harnessing adversarial examples. corr (2015).

Edel Greevy and Alan F Smeaton. 2004. Classifying racist texts using a support vector machine. In *Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 468–469. ACM.

Tommi Gröndahl, Luca Pajola, Mika Juuti, Mauro Conti, and N Asokan. 2018. All you need is" love": Evading hate-speech detection. *arXiv preprint arXiv:1808.09115*.

D Kinga and J Ba Adam. 2015. A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, volume 5.

Ritesh Kumar, Aishwarya N Reganti, Akshit Bhatia, and Tushar Maheshwari. 2018. Aggression-annotated corpus of hindi-english code-mixed data. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC-2018)*.

Irene Kwok and Yuzhou Wang. 2013. Locate the hate: Detecting tweets against blacks. In *Twenty-seventh AAAI conference on artificial intelligence*.

Quoc Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents. In *International Conference on Machine Learning*, pages 1188–1196.

Younghun Lee, Seunghyun Yoon, and Kyomin Jung. 2018. Comparative studies of detecting abusive language on twitter. *arXiv preprint arXiv:1808.10245*.

Minh-Thang Luong, Hieu Pham, and Christopher D Manning. 2015. Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025*.

Yashar Mehdad and Joel Tetreault. 2016. Do characters abuse more than words? In *Proceedings of the 17th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 299–303.

Pushkar Mishra, Helen Yannakoudakis, and Ekaterina Shutova. 2018. Neural character-based composition models for abuse detection. In *Proceedings of the 2nd Workshop on Abusive Language Online (ALW2)*, pages 1–10.

Chikashi Nobata, Joel Tetreault, Achint Thomas, Yashar Mehdad, and Yi Chang. 2016. Abusive language detection in online user content. In *Proceedings of the 25th international conference on world wide web*, pages 145–153. International World Wide Web Conferences Steering Committee.

Ji Ho Park and Pascale Fung. 2017. One-step and two-step classification for abusive language detection on twitter. In *Proceedings of the First Workshop on Abusive Language Online*, pages 41–45.

Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. 2017. Automatic differentiation in pytorch. In *NIPS-W*.

John Pavlopoulos, Prodromos Malakasiotis, and Ion Androutsopoulos. 2017. Deep learning for user comment moderation. In *Proceedings of the First Workshop on Abusive Language Online*, pages 25–35.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. *arXiv preprint arXiv:1802.05365*.

Georgios K Pitsilis, Heri Ramampiaro, and Helge Langseth. 2018. Detecting offensive language in tweets using deep learning. *arXiv preprint arXiv:1801.04433*.

Tim Rocktäschel, Edward Grefenstette, Karl Moritz Hermann, Tomáš Kočiskỳ, and Phil Blunsom. 2015. Reasoning about entailment with neural attention. *arXiv preprint arXiv:1509.06664*.

Anna Schmidt and Michael Wiegand. 2017. A survey on hate speech detection using natural language processing. In *Proceedings of the Fifth International Workshop on Natural Language Processing for Social Media*, pages 1–10.

Minjoon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. 2016. Bidirectional attention flow for machine comprehension. *arXiv preprint arXiv:1611.01603*.

Yi Tay, Anh Tuan Luu, Siu Cheung Hui, and Jian Su. 2018. Reasoning with sarcasm by reading in-between. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1010–1020.

Zeerak Waseem. 2016. Are you a racist or am i seeing things? annotator influence on hate speech detection on twitter. In *Proceedings of the first workshop on NLP and computational social science*, pages 138–142.

Zeerak Waseem and Dirk Hovy. 2016. Hateful symbols or hateful people? predictive features for hate speech detection on twitter. In *Proceedings of the NAACL student research workshop*, pages 88–93.

Ellery Wulczyn, Nithum Thain, and Lucas Dixon. 2017. Ex machina: Personal attacks seen at scale. In *Proceedings of the 26th International Conference on World Wide Web*, pages 1391–1399. International World Wide Web Conferences Steering Committee.

Caiming Xiong, Victor Zhong, and Richard Socher. 2016. Dynamic coattention networks for question answering. *arXiv preprint arXiv:1611.01604*.

Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. 2016. Hierarchical attention networks for document classification. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1480–1489.

Dawei Yin, Zhenzhen Xue, Liangjie Hong, Brian D Davison, April Kontostathis, and Lynne Edwards. 2009. Detection of harassment on web 2.0. *Proceedings of the Content Analysis in the WEB*, 2:1–7.

Ziqi Zhang, David Robinson, and Jonathan Tepper. 2018. Detecting hate speech on twitter using a convolution-gru based deep neural network. In *European Semantic Web Conference*, pages 745–760. Springer.