Hate Speech Detection with Convolutional Neural Networks

Yuyan Chen
McGill University
yuyan.chen2
@mail.mcgill.ca

Yingxuan Wang
McGill University
yingxuan.wang
@mail.mcgill.ca

Zijun Zhao
McGill University
zijun.zhao
@mail.mcgill.ca

Abstract

Hate speech detection has been brought to public attention in recent years with the development of social media platforms. Twitter, as one of the most renowned social media platforms, has been used for many natural language processing tasks including hate speech detection. Inspired by the success of the convolutional neural network (CNN) on text classification tasks, we explored hate speech detection with character level and word level CNNs on two Twitter datasets. We conducted a series of experiments with different data preprocessing and augmentation techniques and compared their effect on character level and word level CNN models. Our experiments showed that (1) CNN based models outperform the logistic regression baseline for hate speech detection. (2) WordCNNs are more suitable than CharCNNs for hate speech detection on small user-generated datasets. (3) Data augmentation and data preprocessing methods have the opposite effect on WordCNN and CharCNN. Our code and datasets can be found here: https://github.com/c-zzj/comp550-project

1 Introduction

As in Brown (2017)'s attempt to interpret the concept "hate speech", hate speech is defined as a heterogeneous collection of speech with the nature of hate. Emotions, feelings, or attributes of hate or hatred, yet subjective, are all part of the essential nature of hate speech, which makes it worse than mere stereotypes or biases. It is best charted as a family resemblances concept which consists of a wide range of intolerable expressive phenomena.

The Internet expands the influence hate speech has on individuals. Twitter, as one of the most renowned online microblogging platforms, while permitting the exchange of diverse expressions of opinions, also aggravates the circulation of unfriendly, even toxic messages, as well as contributing to undesirable cyber-bullying and cyber-aggression. Online hate speech alone could be detrimental to one's mental state, not to mention the intensifying tendency of it inciting people to commit actual crimes and violent actions. As a result, this vicious nature made it indispensable to construct an automatic system to detect hate speech and protect online users from it.

Unlike other text classification tasks, such as sentiment analysis, hate speech detection requires the models to learn more implicit features rather than pure word occurrence. Thus, deep learning models, such as CNNs, with the ability to extract higher-level features might outperform traditional statistical methods for hate speech detection.

As Tweets are user-generated text, they contain noises, including abbreviations, misspellings, slangs, and open-class words, which bring a challenge for word-level vectorization, as many words are Out-of-Vocabulary (OOV), resulting in a large sparse vector representation. Hence, hate detection based on character-level features might have a different performance from word-level models.

In this sense, we propose the following hypotheses: (1) a deep learning based method will have a better performance on the task; (2) a character-level discriminative model with adequate choices of preprocessing will have a better performance on the task; (3) preprocessing and data augmentation approaches shall have the same effects regardless of word-level or character-level models.

To verify our hypotheses, we applied word-level (WordCNN) and character-level convolutional neural networks (CharCNN) to hate speech detection. In particular, as there are few studies on suitable preprocessing techniques for character-level CNNs, we focused on exploring the effect of several text preprocessing and augmentation techniques on the performance of the models.

2 Related work

2.1 Convolutional Neural Network

Convolutional neural networks, designed upon the concept of convolution in signal processing, were first introduced (LeCun et al., 1989) for solving image classification problems. CNNs have been largely and successfully incorporated into many text classification tasks. Kim (2014) applied a simple CNN with one layer of convolution to sentence classification, resulting in a promising result. Zhang et al. (2016) proposed a fairly deep character-level CNN for text classification. The experiment results demonstrated that CharCNN might have better applicability to real-world scenarios. Their findings build the ground of our study.

2.2 Hate Speech Detection Based on CNN

During the past few years, as corporations like Twitter and Meta strived to combat hate speech, more and more scholars simultaneously tried to tackle this issue through training models capable of identifying abusive Twitter posts. Zhang and Luo (2018) proposed to use Deep Neural Network structures as feature extractors, seeking a better solution in targeting the types of hate. Their designed hybrid models: CNN + Gated Recurrent Unit (GRU) and CNN + skipped CNN gained better results than a simple CNN baseline, providing a reliable way for hate speech text feature extraction. Zampieri et al. (2019) composed a new tweets dataset which used a fine-grained three-layer annotation scheme, which was called Offensive Language Identification Dataset (OLID). CNN baseline had achieved the best performance among all baseline models on such a hierarchically annotated dataset. Elouali et al. (2020) adopted CharCNN to detect hate / non-hate tweets in seven languages. The study showed encouraging results for use of CharCNNs in hate speech detection.

3 Method

3.1 Dataset Analysis

We adopted two different datasets to test our hypotheses. Firstly, We adopted the English part of the dataset provided by Ousidhoum et al. (2019) (HOS). The dataset contained five different attributes: directness, hostility, target, group, annotator. We focused on the binary classification task for the "directness" attribute. Table 1 shows the detailed label distribution.

Attribute	Label	Number	Percentage
Directness	Indirect	5078	89.9%
	Direct	569	10.1%
Hostility	Offensive	4020	71.2%
	Normal	1359	24.1%
	Hateful	1278	22.6%
	Disrespectful	782	13.8%
	Abusive	671	11.9%
	Fearful	562	9.95%
Total number	5647		100%

Table 1: Label distribution of each attribute for HOS dataset

Attribute	Label	Number	Percentage
Class	NOT	4456	63.6%
	HOF	2549	36.3%
Hostility	NONE	4456	63.6%
	HATE	1267	18.1%
	OFFN	522	7.4%
	PRFN	760	10.8%
Target	NONE	4456	63.6%
	INT	2286	32.6%
	UNT	263	3.8%
Total number	7005		100%

Table 2: Label distribution of each attribute for HASOC dataset

Secondly, we adopted the dataset provided by HASOC 2019, which was introduced by Mandl et al. (2019). This dataset offered a hierarchical annotation for tweets containing hate speech. As shown in Table 2, stage-1 annotation of "class" was a coarse-grained binary classification which contains *NOT* (not hate speech) and *HOF* (hate or offensive speech). Then the *HOF* class got fine-grained to 3 more detailed categories: *HATE* (contains hateful comment toward groups because of race, political opinion, sexual orientation, gender, social status, health condition, or similar), *OFFN* (texts that are degrading, dehumanizing, insulting, or threatening an individual with violent acts), and *PRFN* (posts with swearword and explicit cursing.) The stage-3 annotation checked if the hate speech was targeting any individual, group, or other: *INT* (someone was targeted in the post), and *UNT* (no one was targeted but the language was unaccepted.) In this study, only the stage-1 annotation was of interest.

3.2 Experiment Details

3.2.1 Preprocessing

Word level For logistic regression and WordCNN, we cleaned the raw data by removing "@user" and "@URL", and removed all non-letter characters. We chose 8,000 words to fit for TF-IDF. For Word2Vec, we only used the first 50 words in each sentence based on Figure 1 and Figure 2 and transformed the words into word vectors pretrained on Google News negative 300.

Character level For CharCNN, we converted uppercase into lowercase and defined the following alphabet containing 68 ASCII characters.

```
abcdefghijklmnopqrstuvwxyz
0123456789
,;.!?:'"\|_@#$%ˆ&*˜‘+-=<>()[]{}
```

The input text is one-hot encoded based on the alphabet, and any character not on the alphabet is encoded as a zero vector.

In addition, we applied stemming (Stem.), lemmatization (Lem.), and stopwords removal (Stop.) separately to the cleaned dataset to explore the effect of data preprocessing on the performance of WordCNN and CharCNN.

3.2.2 Data Augmentation

Because the size of our dataset is relatively small, we tried four common augmentation methods for textual data to enlarge our training set. We adopted the techniques used in Wei and Zou (2019) to realize the augmentation. There are four ways: *synonym replacement* (SR), *random swap* (RS), *random deletion* (RD), and *random insertion* (RI).

SR randomly replaced words in the sentence with its synonym provided from NLTK's WordNet Bird et al. (2009) at a given ratio. RS randomly swapped words within the sentence at a given ratio. RD randomly deleted words in the sentence at a given ratio. RI inserted synonyms of randomly chosen words in the sentence at a given ratio.

3.2.3 Model

CharCNN We adopted Zhang et al. (2016)'s designed architecture for our CharCNN model and changed the activation function from ReLU to ACON-C (Ma et al., 2021), a differentiable, generalized version of the maxout activation family with learnable parameters. Using ACON-C mitigated the dying neuron problem and increased the convergence speed of the model.

WordCNN For word-level CNN, we adopted the CNN static model proposed by Kim (2014). In this design, sentences were represented as vectors of words using Word2Vec, giving a 2D representation for each sentence. Convolutions of different window sizes (3, 4 and 5) are performed on the representations directly and then max pooled. A fully-connected layer was followed by a dropout layer. Predictions were made directly on the output layer.

For both models, we randomly initialized the weights with $\mathcal{N}(0, 0.5)$ to avoid the problem of converging to a local optimum. We set the learning rate to be 5e-4 and used Adam as the optimizer for all models.

All experiments were implemented using PyTorch Paszke et al. (2019). We used an 80-10-10 train-dev-test split and oversampled the test set to deal with imbalanced classes. Each model was set to run for 50 epochs. We used the model with the highest macro-f1 score to predict the labels for the test set and calculate the test scores.

4 Result

We report on both micro and macro-F1 scores on the two binary classification tasks we experimented in Table 3.

For the HOS dataset, WordCNN itself and WordCNN with random deletion achieved the highest macro-F1 score amongst all models up to 0.5905. However, only these two models outperformed our LR baseline. While for micro-F1 scores, some CharCNN and WordCNN models showed competitive results compared to our LR baseline. The highest micro-F1 score, 0.8902, occurred when stopwords removal was applied to our CharCNN model.

For the HASOC dataset, our WordCNN model with Word2Vec embedding performed the best across all designed models. It achieved the highest macro-F1 score up to 0.6688. For WordCNN, none of the preprocessing or data augmentation methods had a positive effect on its macro-F1 score. For CharCNN, removing stopwords and stemming improved the performance. Instead, lemmatization brought down the macro-F1 score substantially.

Even the same model behaved differently for the two datasets. In general, both traditional method and deep learning based methods achieved higher micro-F1 scores on HOS. Simultaneously, most of the methods achieved higher macro-F1 scores

Model	HOS			HASOC		
	Attribute	Micro-F1	Macro-F1	Attribute	Micro-F1	Macro-F1
LR + TF-IDF	Directeness	0.7805	0.5827	Class	0.6348	0.6228
CharCNN		0.8124	0.5130		0.622	0.524
CharCNN + Stop.		**0.8902**	0.5401		0.6063	0.5535
CharCNN + Lem.		0.8655	0.5317		0.5948	0.4956
CharCNN + Stem.		0.8672	0.5606		0.6234	0.568
CharCNN + SR		0.7185	0.4949		0.5235	0.504
CharCNN + RI		0.7186	0.4723		0.5606	0.5398
CharCNN + RS		0.6850	0.4906		0.5778	0.5273
CharCNN + RD		0.7186	0.4861		0.5635	0.5456
WordCNN		0.8584	**0.5905**		**0.7147**	**0.6688**
WordCNN + Stop.		0.8424	0.5820		0.6222	0.635
WordCNN + Lem.		0.8973	0.5197		0.6222	0.6363
WordCNN + Stem.		0.8672	0.5606		0.679	0.6388
WordCNN + SR		0.8424	0.5328		0.5806	0.5454
WordCNN + RI		0.8637	0.5814		0.6562	0.6014
WordCNN + RS		0.7416	0.4994		0.6719	0.6271
WordCNN + RD		0.8655	**0.5905**		0.6533	0.6289

Table 3: Full evaluation scores of the binary classification tasks

on HASOC, with the last three WordCNN models being exceptions.

5 Discussion and conclusion

Since both our datasets are highly imbalanced, we use macro-f1 as the metric to evaluate models' performance. In general, all models have a relatively low macro-f1 score which can be explained by the nature of the task. Since our training set consists entirely of user-generated tweets, each sample can be very different in terms of content and writing style. Besides, unlike other classification tasks, such as sentiment analysis, which highly depends on the occurrence of words, classifying a sentence into hate speech also depends on the context of words. Also, hate speech can have many different targets, for example, race, gender, or religion, making each sample less similar to each other in terms of actual content. Hence, there are fewer lexical features that models can extract during training, decreasing the generalization of the models.

5.1 Model comparison

On both datasets, deep learning models outperformed logistic regression. Specifically, deep learning models increased macro-f1 by 4.60% on HASOC and 0.78%. However, the improvement was not very significant potentially due to the fact that both datasets are relatively small. In Zhang et al. (2016), the authors also mentioned the impact of data size on CharCNN. In their experiments, word-based models keep outperforming CharCNN until the dataset reaches the scale of several million. Hence, even though CharCNN can solve the problem of OOV caused by misspelling, slang, abbreviations, and censored words, the size of datasets limits the performance of CharCNN on hate speech detection based on tweets.

5.2 Data preprocessing

In general, preprocessing had a negative impact on WordCNN but a positive impact on CharCNN. For our first dataset, we found that adding preprocessing strategies to our CharCNN models slightly improved the overall macro-F1 scores by at least 3%. For our second dataset, lemmatization significantly had a negative impact on our CharCNN model. Other than lemmatization, other preprocessing strategies improved our CharCNN model. For both datasets, all preprocessing approaches had a fairly negative impact on the WordCNN model, decreasing macro-F1 scores by an average of 3%. Personal pronouns are key indicators of the directness and hateness of hate speech. As stopwords mainly contain numerous pronouns, our data lost a significant amount of information when stopwords were removed, leading to overall downward performance of word-level models.

5.3 Data augmentation

As we expected, data augmentation has the same effect on CharCNN and WordCNN. In the HOS dataset, all usernames and urls were masked as "user" and "URL", and we cleaned these words during preprocessing. However, in HASOC dataset, the original usernames and urls were kept in the training text. These words cannot imply if a sentence is hate speech and also most of them would be OOV, which might be a reason why data augmentation has a surprisingly negative impact on the HASOC dataset in terms of macro-f1. Moreover, since the augmentation strategies are fairly naive, augmented texts might introduce additional noises to the dataset.

5.4 Conclusion

In conclusion, our experiment results partially aligned with our hypotheses. Indeed, deep learning based methods outperformed the baseline model, and data augmentation had the same effect on character-level and word-level models. The results are inconsistent with our prediction in that (1) the character-level model did not outperform word-level models in all cases; (2) preprocessing had opposite effects on two deep learning models.

Hate speech itself was much more diverse and complicated than other types of text. For example, movie or restaurant reviews in principle have fixed topics, whereas hate speech does not. This makes it difficult for our models to learn useful in-group information. Also, since we had limited access to computational resources, we did not get a chance to explore larger datasets.

For future works, it would be interesting to test our models on larger-scaled datasets. Also, many tweets include non-ASCII characters like emoticons and emojis, which can sometimes serve as a meaningful expression to indicate whether a sentence is hate speech or not. In this sense, it would be interesting to include non-ASCII characters into the alphabet for CharCNN to incorporate emojis and emoticons into the training set. Preprocessing techniques on the character level and other types of text augmentation methods are also worth exploring in future studies.

6 Statement of contribution

Yuyan Chen implemented WordCNN and wrote the report. Yingxuan Wang wrote the majority of the report and searched for data sets and data augmentation methods. Zijun Zhao provided the backbone structure of the whole experiment and implemented CharCNN and data augmentation.

References

Steven Bird, Ewan Klein, and Edward Loper. 2009. *Natural language processing with Python: analyzing text with the natural language toolkit*. " O'Reilly Media, Inc.".

Alexander Brown. 2017. What is hate speech? part 1: The myth of hate. *Law and Philosophy*, 36(4):419–468.

Aya Elouali, Zakaria Elberrichi, and Nadia Elouali. 2020. Hate speech detection on multilingual twitter using convolutional neural networks. *Revue d'Intelligence Artificielle*, 34(1):81–88.

Yoon Kim. 2014. Convolutional neural networks for sentence classification.

Yann LeCun, Bernhard Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne Hubbard, and Lawrence D Jackel. 1989. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4):541–551.

Ningning Ma, Xiangyu Zhang, Ming Liu, and Jian Sun. 2021. Activate or not: Learning customized activation.

Thomas Mandl, Sandip Modha, Prasenjit Majumder, Daksh Patel, Mohana Dave, Chintak Mandlia, and Aditya Patel. 2019. Overview of the hasoc track at fire 2019: Hate speech and offensive content identification in indo-european languages. In *Proceedings of the 11th Forum for Information Retrieval Evaluation*, FIRE '19, page 14–17, New York, NY, USA. Association for Computing Machinery.

Nedjma Ousidhoum, Zizheng Lin, Hongming Zhang, Yangqiu Song, and Dit-Yan Yeung. 2019. Multilingual and multi-aspect hate speech analysis. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4675–4684, Hong Kong, China. Association for Computational Linguistics.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. dAlché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc.

Jason Wei and Kai Zou. 2019. EDA: Easy data augmentation techniques for boosting performance on text classification tasks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6383–6389, Hong Kong, China. Association for Computational Linguistics.

Marcos Zampieri, Shervin Malmasi, Preslav Nakov, Sara Rosenthal, Noura Farra, and Ritesh Kumar. 2019. Predicting the type and target of offensive posts in social media. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1415–1420, Minneapolis, Minnesota. Association for Computational Linguistics.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2016. Character-level convolutional networks for text classification.

Ziqi Zhang and Lei Luo. 2018. Hate speech detection: A solved problem? the challenging case of long tail on twitter.

A Appendix

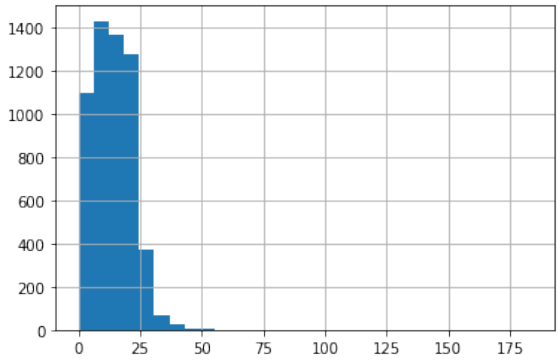

Figure 1: Sentence length of dataset 1: HOS

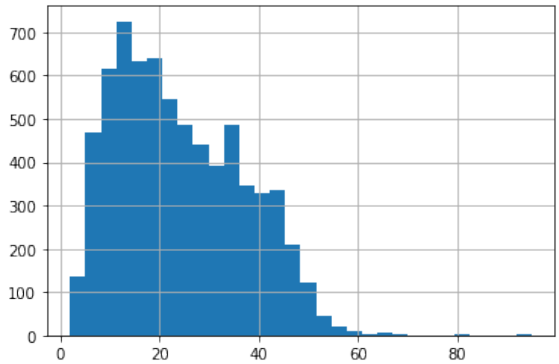

Figure 2: Sentence length of dataset 2: HASOC