Ghmerti at SemEval-2019 Task 6: A Deep Word- and Character-based Approach to Offensive Language Identification

Ehsan Doostmohammadi[♣,♠], **Hossein Sameti**[♣], **Ali Saffar**[♠]

[♣]Speech Processing Lab, Department of Computer Engineering,
Sharif University of Technology, Tehran, Iran
[♠]NazarBin, Tehran, Iran
e.doostm72@student.sharif.edu, sameti@sharif.edu,
saffar@nazarbin.com

Abstract

This paper presents the models submitted by Ghmerti team for subtasks A and B of the OffensEval shared task at SemEval 2019. OffensEval addresses the problem of identifying and categorizing offensive language in social media in three subtasks; whether or not a content is offensive (subtask A), whether it is targeted (subtask B) towards an individual, a group, or other entities (subtask C). The proposed approach includes character-level Convolutional Neural Network, word-level Recurrent Neural Network, and some preprocessing. The performance achieved by the proposed model for subtask A is 77.93% macro-averaged F_1-score.

1 Introduction

The massive rise in user-generated web content, alongside with the freedom of speech in social media and anonymity of the users has brought about an increase in online offensive content and antisocial behavior. The consequences of such behavior on genuine users of the social media have become a serious concern for researchers in Natural Language Processing and related fields in recent years.

The shared task number 6 at SemEval 2019, OffensEval (Zampieri et al., 2019b), proposes to model the task of offensive language identification hierarchically, which means identifying the offensive content, whether it is targeted, and if so, the target of the offense. In OffensEval, offensive language is defined as "any form of non-acceptable language (profanity) or a targeted offense, which can be veiled or direct" which includes "insults, threats, and posts containing profane language or swear words" (Zampieri et al., 2019b).

We have participated in the first two subtasks (A and B) of OffensEval with the proposed approach of a deep model consisting of a Recurrent Neural Network (RNN) for word-level and Convolutional Neural Network (CNN) for character-level processing[1]. Character-level processing is beneficial, as offensive comments are likely to follow unorthodox writing styles, contain obfuscated words, or have irregular word separation which leads to tokenization issues (Mehdad and Tetreault, 2016; Nobata et al., 2016). We also experimented with two other methods, a Support Vector Machine (SVM) with TFIDF and count features and another SVM with BERT (Devlin et al., 2018) -encoded sentences as input, both with lower performances comparing with the deep model.

After overviewing the related work in section 2, we discuss the methodology and the data in details in section 3, and the results in section 4. In section 5, we analyze the results and conclude the paper in section 6.

2 Related Work

Offensive language identification has been of interest for researchers in recent years. Early work in the related fields include detection of online trolling (Cambria et al., 2010), racism (Greevy and Smeaton, 2004), and cyberbullying (Dinakar et al., 2012).

Papers published in recent years include (Davidson et al., 2017), which introduces the Hate Speech Detection dataset and experiments with different machine learning models, such as logistic regression, naïve Bayes, random forests, and linear SVMs to investigate hate speech and offensive language, (Malmasi and Zampieri, 2017) which experiments further on the same dataset using SVMs with n-grams and skip-grams features, and (Gambäck and Sikdar, 2017) and (Zhang et al., 2018), both exploring the performance of neural networks and comparing them with other machine

[1] You can find the code of the deep model on this project's repository on github: github.com/edoost/offenseval

Proceedings of the 13th International Workshop on Semantic Evaluation (SemEval-2019), pages 617–621
Minneapolis, Minnesota, USA, June 6–7, 2019.

learning approaches. Also, there has been published a couple of surveys covering various work addressing the identification of abusive, toxic, and offensive language, hate speech, etc., and their methodology including (Schmidt and Wiegand, 2017) and (Fortuna and Nunes, 2018).

Additionally, there were several workshops and shared tasks on offensive language identification and related problems, including TA-COS[2], Abusive Language Online[3], and TRAC[4](Kumar et al., 2018), and GermEval (Wiegand et al., 2018), which shows the significance of the problem.

3 Methodology and Data

The methodology used for both subtask A, offensive language identification, and subtask B, automatic categorization of offense types, consists of a preprocessing phase and a deep classification phase. We first introduce the preprocessing phase, then elaborate on the classification phase.

3.1 Preprocessing

The preprocessing phase consists of (1) replacing obfuscated offensive words with their correct form and (2) tweet tokenization using NLTK tweet tokenizer (Bird et al., 2009). In social media, some words are distorted in a way to escape the offense detection systems or to reduce the impertinence. For instance, 'asshole' may be written as 'a$$hole', 'a$sh0le', 'a**hole', etc. Having a list of English offensive words, we can create a list containing most of the possible permutations. Using such a list will ease the job for the classifier and searching in it is computationally cheap. Furthermore, replacing contractions, e.g. 'I'm' with 'I am', and replacing common social media abbreviations, e.g. 'w/' with 'with', were not helpful and were not used to train the final model.

3.2 Deep Classifier

Given a tweet, we want to know if its offensive or not (subtask A), and if the offense is targeted (subtask B). Regarding that both subtasks are problems of binary classification, we used one architecture to tackle both. To define the problem, if we have a tweet x, we want to predict the label y, `OFF` or `NOT` in subtask A, and `TIN` or `UNT` in subtask B. Two representations are therefore created for each input x:

1. x_c which is the indexed representation of the tweet based on its characters padded to the length of the longest word in the corpus. The indices include 256 of the most common characters, plus 0 for padding and 1 for unknown characters.

2. x_w which is the embeddings of the words in the input tweet based on FastText's 600B-token common crawl model (Mikolov et al., 2018).

Then, x_c is fed into an embedding layer with output size of 32 and a CNN layer after that. x_c is then concatenated with x_w and both are fed to a unidirectional RNN with LSTM cell of size 256, the output of which is the input to two consecutive fully-connected layers that map their input to an $\mathbb{R}^{128}$ and an $\mathbb{R}^2$ space, respectively. We also applied dropout of keeping rate 0.5 on CNN's output, x_w, RNN's output, and the first fully-connected layer's output.

The CNN layer consists of four consecutive sub-layers:

1. CNN consisting of 64 filters with kernel size of 2, stride of 1, same padding and RELU activation;

2. max-pooling layer with pool size and stride of 2;

3. another CNN, same as the first one, but with 128 filters;

4. the same max-pooling again.

Finally, we used an AdamOptimizer (Kingma and Ba, 2014) with learning rate of $1e{-}3$ and batch size of 32 to train the model.

3.3 Baseline Methods

We used two baseline methods for subtask A:

- an SVM with 1- to 3-gram word TFIDF and 1- to 5-gram character count feutrue vectors as input;

- an SVM with BERT representations of the tweets (using average pooling (Xiao, 2018)) as input using `BERT-Large, Uncased` model.

[2] http://ta-cos.org/

[3] https://sites.google.com/site/abusivelanguageworkshop2017/

[4] https://sites.google.com/view/trac1/home

The SVMs were trained for 15 epochs with stochastic gradient descent, hinge loss, alpha of 1e−6, `elasticnet` penalty, and `random_state` of 5. The SVMs were implemented using Scikit-learn (Pedregosa et al., 2011).

3.4 Data

The main dataset used to train the model is Offensive Language Identification Dataset (OLID) Zampieri et al. (2019a). The dataset is annotated hierarchically to identify offensive language (OFFensive or NOT), whether it is targeted (Targeted INsult or UNTargeted), and if so, its target (INDividual, GRouP, or OTHer). We divided the 13,240 samples in the training set into 12,000 samples for training and 1,240 samples for validation.

As neural networks require huge amount of training data, we tried adding more data from the dataset of the First Workshop on Trolling, Aggression, and Cyberbullying (TRAC-1) (Kumar et al., 2018) which was not helpful. However, adding the training data from Toxic Comment Classification Challenge on Kaggle (Conversation AI, 2017) increased the macro-averaged F_1-score on the validation set by $\sim 2\%$. This data comprises tweets with positive and negative tags in six categories: `toxic`, `severe_toxic`, `obscene`, `threat`, `insult`, `identity_hate`. We only used `toxic` and `severe_toxic` positive samples as `OFF` and the ones with no positive label in any category as `NOT`. None of the data from other categories, either positive or negative, were included in the additional training data. After that, we were left with 109,236 samples, most of which were labeled as `NOT`. To balance `OFF` and `NOT` samples, 84,626 of `NOT` samples were randomly removed. In the end, 12,305 `OFF` and 12,305 `NOT` samples were added to the training data.

4 Results

Finally, we trained the baseline models in 3.3 and the model described in 3.2 using the combination of the OLID training data and the data from Toxic Comment Classification Challenge (which is described in 3.4).

You can see the macro-averaged F_1-score and accuracy on the test set for the baseline scores provided by task organizers, baseline methods we used (on both training and validation data), and the deep classifier model (DeepModel) in table 1. DeepModel is trained on the training data (not including the validation data) and DeepModel+val on the combination of the training and validation data. The best performance is in bold.

System	Macro F_1	Accuracy
All NOT baseline	0.4189	0.7209
All OFF baseline	0.2182	0.2790
SVM	0.7452	0.8011
BERT-SVM	0.7507	0.8011
DeepModel	0.7788	0.8326
DeepModel+val	**0.7793**	**0.8337**

Table 1: Results for subtask A

The best performance belongs to DeepModel+val by a margin of more than 2.8 percent, with the best baseline performance, BERT-SVM. However, it should be mentioned that the results in the first two rows belong to a model trained only on OLID. You can see the confusion matrix for the best performance in figure 1.

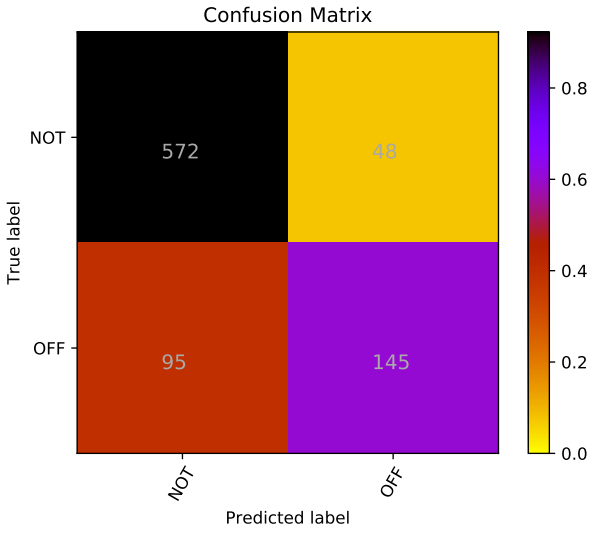

Figure 1: The confusion matrix for DeepModel+val in subtask A

From the confusion matrix we can see that the performance of DeepModel+val on `NOT` is quite good, but not on `OFF`. You can see the detailed results of DeepModel+val in table 2.

	Precision	Recall	F_1-score
`NOT`	0.8576	0.9226	0.8889
`OFF`	0.7513	0.6042	0.6697

Table 2: Detailed DeepModel+val results in subtask A

In subtask B, DeepModel+val outperformed the

baseline results by a large margin, like subtask A. The results for subtask B are presented in table 3.

System	Macro F_1	Accuracy
All TIN baseline	0.4702	0.8875
All UNT baseline	0.1011	0.1125
DeepModel	0.6065	0.8583
DeepModel+val	**0.6400**	**0.8875**

Table 3: Results for subtask B

This time, adding the validation data made a considerable difference, as the training data for subtask B is fewer. You can see the confusion matrix for DeepModel+val in figure 2.

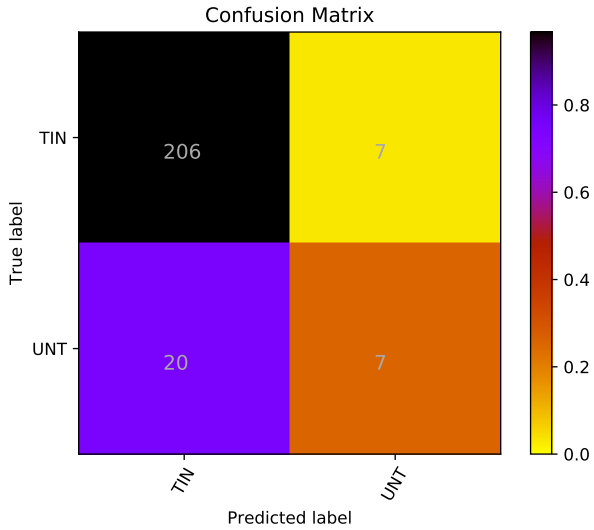

Figure 2: The confusion matrix for the DeepModel+val in subtask B

The confusion matrix shows that the performance of the model is good for TIN, but poor for UNT. Table 4 shows the detailed results for DeepModel+val in subtask B, which indicates that the imbalance is worse than subtask A and the poor performance on UNT is mainly due to low recall.

	Precision	Recall	F_1-score
TIN	0.9115	0.9671	0.9385
UNT	0.5000	0.2593	0.3415

Table 4: Detailed DeepModel+val results in subtask B

5 Analysis

In subtask A, DeepModel+val outperformed the second best method, BERT-SVM, by 2.86% Macro F_1-score. BERT-SVM results, however, were not much better than the SVM with TFIDF and count features, probably due the fact that the BERT model requires fine-tuning for more task-specific representations.

The majority of DeepModel+val's errors are in OFF class and can be categorized into (1) sarcasm: the model is unable to detect sarcastic language which is even difficult for humans to detect; (2) emotion: discerning emotions, such as anger, seems to be a challenge for the model; (3) ethnic and racial slurs, etc. Solving these problems require a more comprehensive knowledge of the context and the language, which was examined in works such as (Poria et al., 2016) and improved the results. However, experimenting with emotion embeddings in the current work was not helpful and did not appear in the final results. Being aware of the emotion of the text, personality of the author, and sentiment of the sentences is helpful to detect offensive language, as many offensive contents have an angry tone (ElSherief et al., 2018) or do not contain profane language (Malmasi and Zampieri, 2018). One can also make use of the benefits of BERT's context and sentence sequence awareness by fine-tuning it on the training data, which is computationally expensive and was not feasible for the authors of this paper.

6 Conclusion

In this paper, we introduced Ghmerti team's approach to the problems of 'offensive language identification' and 'automatic categorization of offense type' in shared task 6 of SemEval 2019, OffensEval. In subtask A, the neural network-based model outperformed the other methods, including an SVM with word TFIDF and character count features and another SVM with BERT-encoded tweets as input. Furthermore, analysis of the results indicates that sarcastic language, inability to discern the emotions such as anger, and ethnic and racial slurs constitute a considerable portion of the errors. Such deficiencies demand larger training corpora and variety of other features, such as information on sarcasm, emotion, personality, etc.

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.".

Erik Cambria, Praphul Chandra, Avinash Sharma, and Amir Hussain. 2010. Do not feel the trolls. *ISWC, Shanghai*.

Conversation AI. 2017. Toxic comment classification challenge: Identify and classify toxic online comments.

Thomas Davidson, Dana Warmsley, Michael Macy, and Ingmar Weber. 2017. Automated Hate Speech Detection and the Problem of Offensive Language. In *Proceedings of ICWSM*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Karthik Dinakar, Birago Jones, Catherine Havasi, Henry Lieberman, and Rosalind Picard. 2012. Common sense reasoning for detection, prevention, and mitigation of cyberbullying. *ACM Transactions on Interactive Intelligent Systems (TiiS)*, 2(3):18.

Mai ElSherief, Vivek Kulkarni, Dana Nguyen, William Yang Wang, and Elizabeth Belding. 2018. Hate Lingo: A Target-based Linguistic Analysis of Hate Speech in Social Media. *arXiv preprint arXiv:1804.04257*.

Paula Fortuna and Sérgio Nunes. 2018. A Survey on Automatic Detection of Hate Speech in Text. *ACM Computing Surveys (CSUR)*, 51(4):85.

Björn Gambäck and Utpal Kumar Sikdar. 2017. Using Convolutional Neural Networks to Classify Hate-speech. In *Proceedings of the First Workshop on Abusive Language Online*, pages 85–90.

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.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Ritesh Kumar, Atul Kr. Ojha, Shervin Malmasi, and Marcos Zampieri. 2018. Benchmarking Aggression Identification in Social Media. In *Proceedings of the First Workshop on Trolling, Aggression and Cyberbulling (TRAC)*, Santa Fe, USA.

Shervin Malmasi and Marcos Zampieri. 2017. Detecting Hate Speech in Social Media. In *Proceedings of the International Conference Recent Advances in Natural Language Processing (RANLP)*, pages 467–472.

Shervin Malmasi and Marcos Zampieri. 2018. Challenges in Discriminating Profanity from Hate Speech. *Journal of Experimental & Theoretical Artificial Intelligence*, 30:1–16.

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.

Tomas Mikolov, Edouard Grave, Piotr Bojanowski, Christian Puhrsch, and Armand Joulin. 2018. Advances in pre-training distributed word representations. In *Proceedings of the International Conference on Language Resources and Evaluation (LREC 2018)*.

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.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Soujanya Poria, Erik Cambria, Devamanyu Hazarika, and Prateek Vij. 2016. A deeper look into sarcastic tweets using deep convolutional neural networks. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 1601–1612.

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. Association for Computational Linguistics*, pages 1–10, Valencia, Spain.

Michael Wiegand, Melanie Siegel, and Josef Ruppenhofer. 2018. Overview of the GermEval 2018 Shared Task on the Identification of Offensive Language. In *Proceedings of GermEval*.

Han Xiao. 2018. bert-as-service. https://github.com/hanxiao/bert-as-service.

Marcos Zampieri, Shervin Malmasi, Preslav Nakov, Sara Rosenthal, Noura Farra, and Ritesh Kumar. 2019a. Predicting the Type and Target of Offensive Posts in Social Media. In *Proceedings of NAACL*.

Marcos Zampieri, Shervin Malmasi, Preslav Nakov, Sara Rosenthal, Noura Farra, and Ritesh Kumar. 2019b. SemEval-2019 Task 6: Identifying and Categorizing Offensive Language in Social Media (OffensEval). In *Proceedings of The 13th International Workshop on Semantic Evaluation (SemEval)*.

Ziqi Zhang, David Robinson, and Jonathan Tepper. 2018. Detecting Hate Speech on Twitter Using a Convolution-GRU Based Deep Neural Network. In *Lecture Notes in Computer Science*. Springer Verlag.