NLP@UIOWA at SemEval-2019 Task 6: Classifying the Crass using Multi-windowed CNNs

Jonathan Rusert and **Padmini Srinivasan**
Department of Computer Science
University of Iowa
Iowa City, IA, USA
{jonathan-rusert, padmini-srinivasan}@uiowa.edu

Abstract

This paper proposes a system for OffensEval (SemEval 2019 Task 6), which calls for a system to classify offensive language into several categories. Our system is a text based CNN, which learns only from the provided training data. Our system achieves 80 - 90% accuracy for the binary classification problems (offensive vs not offensive and targeted vs untargeted) and 63% accuracy for trinary classification (group vs individual vs other).

1 Introduction

Background. Social media (e.g. Twitter, Facebook) is widely used today. For example, 68% of all Americans report owning a Facebook account in 2018 (Smith and Anderson, 2018), while 71% of Americans (within the ages of 18-24) report using Twitter. Online gaming is a second popular use of the internet, reaching upwards of 80 - 100 million monthly users depending on game (Goslin, 2018). These uses demonstrate the internet as a way for humans to connect with others. However, connecting with others can carry a downside. More than 1 in 3 young people have been cyberbullied online (cyb, 2018), this extends to around half of teens. Besides cyberbullying, offensive language, on a public forum, can cause users to stay away from certain platforms. Because of this, companies have increased their efforts to remove offensive language from their platforms (Terdiman, 2018). With the large amount of traffic these platforms see, a purely manual approach to detecting/removing offensive language is impossible, which means an automated approach is needed to help. OffensEval (Zampieri et al., 2019b) provides a community driven opportunity to build such systems. We approach this problem of classifying offensive tweets with a CNN architecture trained on the provided training dataset.

2 Proposed Approach

Our system is a variation of a Convolutional Neural Network (CNN), which was chosen since it has seen success previously with classification tasks

CNN Infrastructure. We experimented with two different Convolutional Neural Networks. The first whose architecture is based on the CNN originally proposed in (Kim, 2014) (CNN 1), and the second a combination of multiple CNNs (CNN 2). We further discuss each of these CNNs and their comparison on the training data. The visual structure for CNN 1 and CNN 2 can be found in figure 1 and figure 2 respectively.

Preprocessing. Both CNN 1 and CNN 2 begin by preprocessing the text of the tweet. As noted in section 3, URLs and user mentions are already denoted as URL and USER. Basic cleaning of the text is applied, includes removal of punctuation, converting text to lowercase, and filtering of stopwords via NLTK's stopword list[1]. Finally, all separated words are tokenized via nltk's tokenize() function[2].

Embedding Layer. CNN 1 and CNN 2 encode the text of a tweet as a word embedding with dimension j. We experimented with several j arriving at j = 100. Word embeddings for Non-Out of Vocabulary (OOV) words are obtained from Glove (Pennington et al., 2014) which has been trained on Twitter data[3]. Experiments were also conducted with Glove common crawl data, but no visible improvement was found. OOV words are randomly initialized as a word embedding. The embedding layer takes in i word embeddings of length j, where the i word embeddings are combined in the same sequential order as they appear in the tweet. We choose i as the length of the

[1]nltk.org/api/nltk.corpus.html
[2]nltk.org/api/nltk.tokenize.html
[3]nlp.stanford.edu/projects/glove/

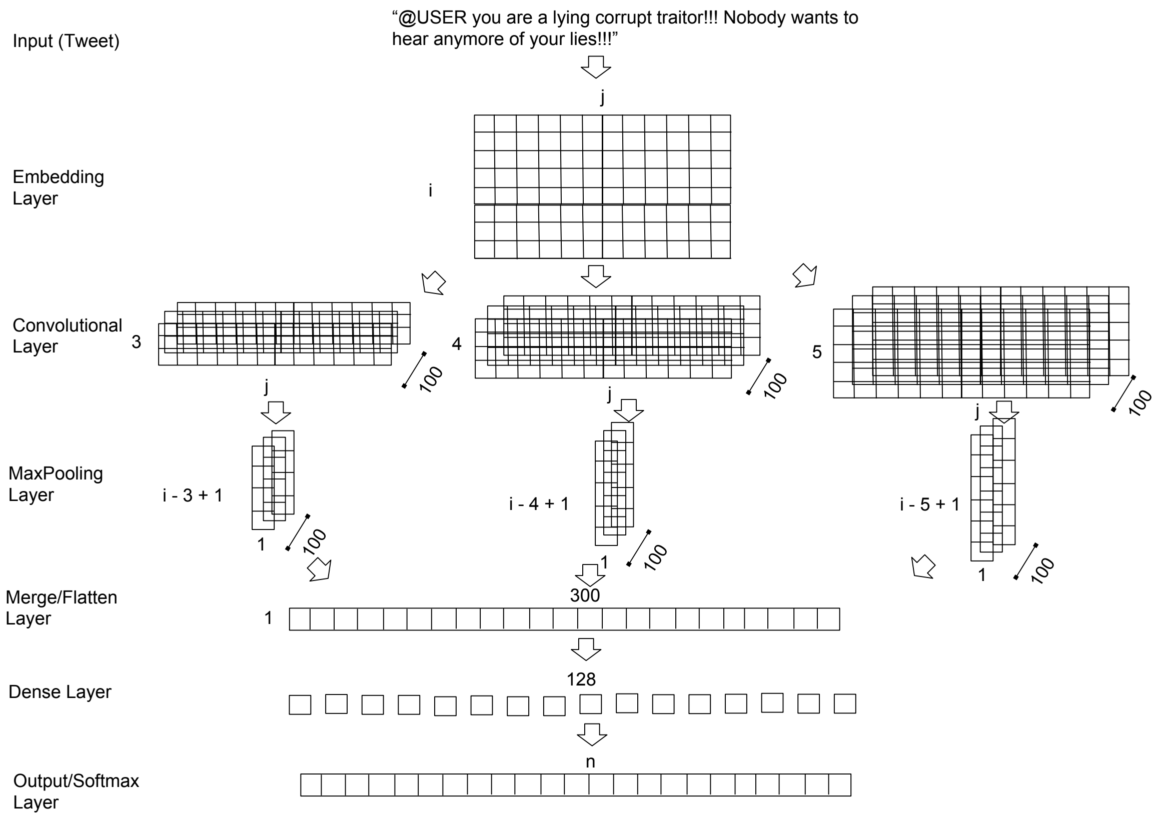

Figure 1: CNN 1 's Architecture

longest tweet (i.e. number of words after preprocessing). Any tweets less than *i* length are padded with zero embeddings at the end. CNN 1 and CNN 2 differ at this point and will be examined separately.

CNN 1 **Convolutional Layer.** CNN 1 applies three $k \times j$ convolutional windows to the embedding layer: a 3 x *j*, a 4 x *j*, and a 5 x *j* window. Applying each window to the embedding layer results in a $(i - k + 1) \times 1$ output, where k = {3,4,5} and corresponds to the length of the window. 100 filters of each window are applied to the embedding layer resulting in 100 $(i - k + 1) \times 1$ outputs for each *k*.

CNN 1 **Max pooling/Merge Layer** A max pooling of size $(i - k + 1) \times 1$ is applied to each separate filter output from the convolutional layer. The resulting outputs from all three max pooling streams are merged together then flattened to a 300 neuron layer.

CNN 1 **Dense Layer/Output Layer** The flattened layer is fed into a dense layer consisting of 128 neurons. ReLu is chosen as the activation function. Finally, the output of the dense layer is passed to the output layer of size *n* where *n* = number of classes. The output layer uses a softmax function as activation.

CNN 2 **Convolutional Layer.** CNN 2 applies three sets of three convolution windows to the embedding layer, each window in the format $k \times j$. The first set of convolution windows are $k = [2, 3, 4]$, the second are $k = [3, 4, 5]$, and third are $k = [4, 5, 6]$. Similar to CNN 1 , applying these filters results in a $(i - k + 1) \times 1$ output, and 100 filters exist for each *k* for each set of windows.

CNN 2 **Max pooling/Merge Layer.** Max pooling, in this instance, behaves similarly to CNN 1 . However, instead of merging all the pooled layers, only those in the same set are merged. This results in three separate flattened 300 neuron layers.

CNN 2 **Separate Dense layers** The three separate merged layers are fed through two dense layers consisting of 128 neurons each. ReLu activation function is used for all dense layers. Each merged layer is fed to their own respective dense layers. The second dense layers are finally merged together.

CNN 2 **Final Dense/Output Layers** The merged layer is fed through two more 128 dense layers with ReLu activation. Finally, the result is

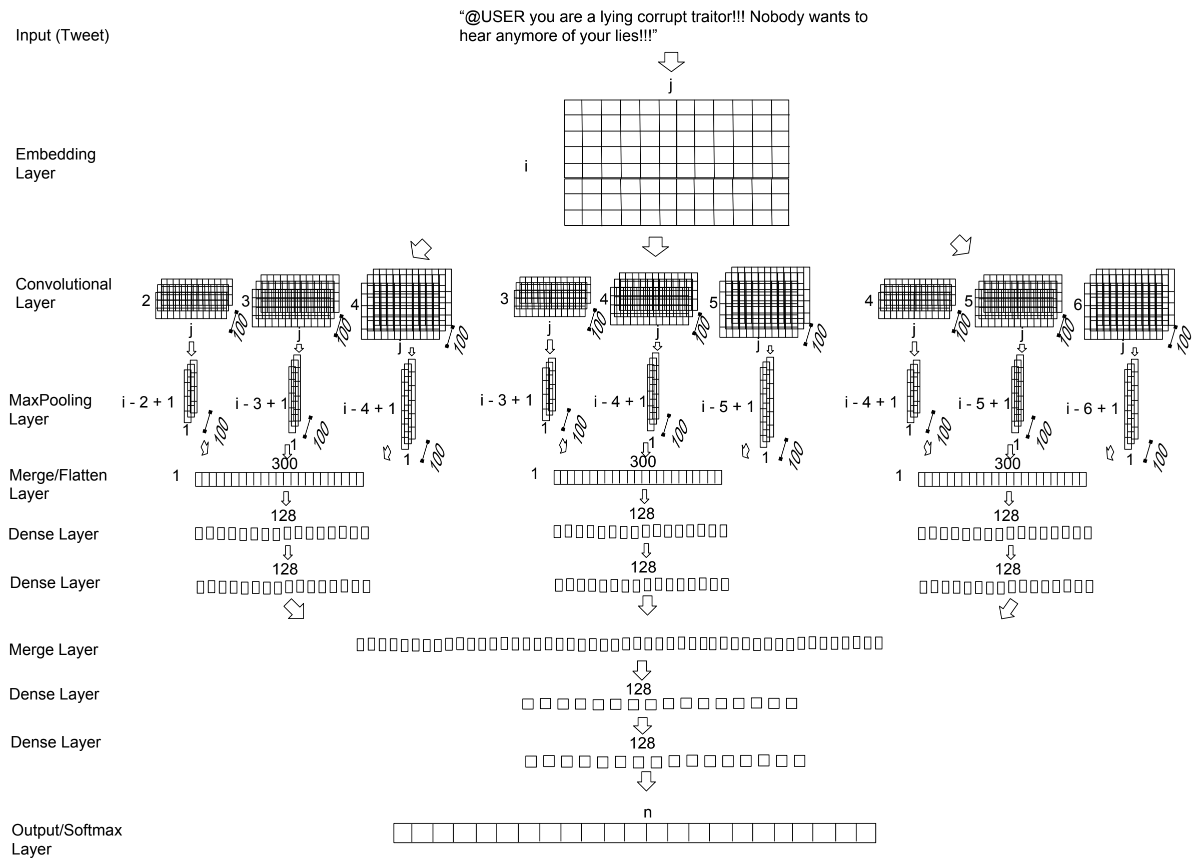

Figure 2: CNN 2 's Architecture

fed to the output layer of size n with softmax activation.

Hyperparameters/Training We experimented with different epochs and batch sizes and ended up finding epochs=30 and batch size = 50 worked best for our models. The only data trained on was the training data provided. More on this data in section 3. The system was implemented with Keras[4] and Tensorflow as the backend.

3 Dataset

Training Set. The data collection methods used to compile the dataset provided in OffensEval is described in Zampieri et al. (2019a). The training data set provided consists of 13,240 tweets. Each tweet, consists of up to three classifications, which correspond to subtasks further described in section 4. The classifications are as follows:

i. OFF - This post contains offensive language or a targeted (veiled or direct) offense

ii. NOT - This post does not contain offense or profanity.

iii. TIN - A post containing an insult or threat to an individual, a group, or others

iv. UNT - A post containing non-targeted profanity and swearing.

v. IND - The target of the offensive post is an individual: a famous person, a named individual or an unnamed person interacting in the conversation.

vi. GRP - The target of the offensive post is a group of people considered as a unity due to the same ethnicity, gender or sexual orientation, political affiliation, religious belief, or something else.

vii. OTH - The target of the offensive post does not belong to any of the previous two categories (e.g., an organization, a situation, an event, or an issue)

A tweet which is classified as OFF, can be further classified into TIN or UNT. If classified as TIN, the tweet can be further classified into IND, GRP, or OTH. A breakdown of the frequency of each class label can be found in table 1.

[4]https://keras.io/

OFF 4400				NOT 8840
TIN 3876			UNT 524	
IND 2407	GRP 1074	OTH 395		

Table 1: A breakdown of frequency of labels of tweets, the classes underneath are further classifications of classes above (e.g. a tweet labeled IND is also labeled TIN and OFF)

Test Set. The test set provided follows the same classification rules as training and consists of 860 tweets. The 860 tweets can be classified into OFF or NOT, then 240 OFF tweets can be classified as TIN or UNT, and finally 213 TIN tweets can be classified as IND, GRP or OTH.

4 Subtasks

OffensEval divided the overall task of identifying/classifying offensive language into three subtasks, subtask A, B, and C.

Subtask A. Subtask A requires a system to classify tweets as either offensive (OFF) or not offensive (NOT). An example of a tweet marked as OFF (in provided training):

@USER you are a lying corrupt traitor!!! Nobody wants to hear anymore of your lies!!!.

An example of a tweet marked as NOT:

@USER Buy more icecream!!!.

A more expanded look at the training data can be found in section 3.

Subtask A Results. As our system only trained on the provided gold standard, this data set was used to gauge the effectiveness of our two systems. Five fold cross validation was used for predicting training data. The results for subtask A on training data can be found in table 2. CNN 1 and CNN 2 achieve similar results, an accuracy of 0.7468 and 0.7555, and a macro F1 score of 0.7130 and 0.7114, respectively. The results for our systems' performance on OffensEval test data subtask A can be found in table 3. As with the training data, CNN 1 and CNN 2 perform similarly on this task, with CNN 1 achieving 0.8 accuracy and a macro F1 score of 0.73.

Subtask B. Subtask B requires further classification of OFF tagged into two categories, targeted (TIN) and untargeted (UNT). An example of untargeted tweet is:

System	Acc.	Pr.	Re.	F1
CNN 1	0.7469	0.7145	0.7117	0.7130
CNN 2	0.7555	0.7256	0.7036	0.7114

Table 2: Subtask A Training Data Results, Pr.=Macro Precision, Re.=Macro Recall, F1=Macro F1 Score

System	Acc.	Pr.	Re.	F1
CNN 1	0.7988	0.7552	0.7175	0.7314
CNN 2	0.7767	0.7250	0.7379	0.7306
All NOT	0.7209			0.4189
All OFF	0.2790			0.2182

Table 3: Subtask A Test Results, Pr.=Macro Precision, Re.=Macro Recall, F1=Macro F1 Score. All OFF, NOT are baselines where that specific label was assigned to all tweets.

@USER @USER My favourite part of this is watching all the conservatives lose their minds as usual. Once again the Democrats a re being mean to us boo-hoo. LOL.

An example of targeted:

@USER You are a complete knob! It's ppl like you who are messing up this country. More details on data in section 3.

Subtask B Results. The results for cross validation on subtask B's training data are found in table 4. CNN 2 achieves a greater accuracy over CNN 1 on this subtask, 0.8723 compared to 0.8222, but still achieves a smaller macro F1 score of 0.5673 compared to 0.5827. Subtask B test results for our systems are found in table 5. Similar to training, CNN 2 outperforms CNN 1 in accuracy, 0.8958 to 0.8750, but achieves a similar trend in macro F1 scores, 0.6511 and 0.6528.

Subtask C. Subtask C requires further classification of those tweets tagged are targeted (TIN), into three classes, Individual (IND), Group (GRP), and Other (OTH). Examples:

IND tweet -*@USER You are a complete knob! It's ppl like you who are messing up this country*

GRP tweet - *@USER Assuming liberals are unarmed would be a grave mistake by the deplorables.*

System	Acc.	Pr.	Re.	F1
CNN 1	0.8222	0.5815	0.5839	0.5827
CNN 2	0.8723	0.6442	0.5545	0.5673

Table 4: Subtask B Training Data Results, Pr.=Macro Precision, Re.=Macro Recall, F1=Macro F1 Score

System	Acc.	Pr.	Re.	F1
CNN 1	0.8750	0.6732	0.6385	0.6528
CNN 2	0.8958	0.7478	0.6179	0.6511
All TIN	0.8875			0.4702
All UNT	0.1125			0.1011

Table 5: Subtask B Test Results, Pr.=Macro Precision, Re.=Macro Recall, F1=Macro F1 Score. All TIN, UNT are baselines where that specific label was assigned to all tweets.

System	Acc.	Pr.	Re.	F1
CNN 1	0.6925	0.5372	0.5224	0.5282
CNN 2	0.6772	0.5147	0.5136	0.5140

Table 6: Subtask C Training Data Results, Pr.=Macro Precision, Re.=Macro Recall, F1=Macro F1 Score

OTH tweet - *@USER Shooting in USA is so common no one is talking about gun control any more.*

More details on subtask C data in section 3.

Subtask C Results. The results for subtask C's cross validation on training data can be found in table 6. CNN 1 slightly outperforms CNN 2 in this task, achieving an accuracy of 0.6925 over 0.6772 and a macro F1 score of 0.5282 over 0.5140. Subtask C's test data results can be found in table 7. As subtask C is a three class problem (compared to the two class problem of A, B), the accuracy and macro F1 scores are lower overall. As with training, CNN 1 slightly outperforms CNN 2 in both accuracy, 0.6291 to 0.6197, and macro F1 score, 0.5061 to 0.4939.

5 Discussion

Systems outperform single labels. On test data, for all three subtasks, our system outperforms the baseline, provided by organizers, for assigning a single label for all tweets. Outperforming the best baseline (labeling tweets with highest frequent label) in terms of F1 by 0.32 in subtask A, 0.18 in subtask B, and 0.29 in subtask C. Outperformances in accuracy are seen in all three subtasks as well.

System	Acc.	Pr.	Re.	F1
CNN 1	0.6291	0.5513	0.5148	0.5061
CNN 2	0.6197	0.5079	0.5014	0.4939
All GRP	0.3662			0.1787
All IND	0.4695			0.2130
All OTH	0.1643			0.0941

Table 7: Subtask C Test Results, Pr.=Macro Precision, Re.=Macro Recall, F1=Macro F1 Score. All GRP, IND, OTH are baselines where that specific label was assigned to all tweets.

		NOT	OFF	
Key	NOT	559	61	620
	OFF	112	128	240
		System		860

Table 8: CNN 1 's Confusion matrix for subtask A

		TIN	UNT	
Key	TIN	201	12	213
	UNT	18	9	27
		System		240

Table 9: CNN 1 's Confusion matrix for subtask B

Results follow data distribution. The test confusion matrices for the higher scoring system (CNN 1), for subtask A, B, and C, can be found in table 8, table 9, and table 10 respectively. Training data for OFF and NOT make up 67% and 33% respectively. For test data, the percentages are 28% for OFF and 78% for NOT. As expected, our system identifies better identifies NOT (559/620 tweets) compared to OFF (128/240 tweets). Similar results occur for subtask B, TIN (201/213) compared to UNT(9/27), and subtask C, GRP (33/78) compared to (95/100) compared to (6/35). These results all follow distribution of training data, which might point to lack of training data for poorer results for smaller classes since deep learning systems depend on a good amount of training data.

Added complexity of CNN 2 **adds little to no improvement.** Although CNN 2 shows greater performance in accuracy on the training data for subtasks A and B, the performance does not follow through in the test data, as CNN 1 outperformed CNN 2 in subtask A and C for accuracy and all

		GRP	IND	OTH	
Key	GRP	33	34	11	78
	IND	3	95	2	100
	OTH	12	17	6	35
		System			213

Table 10: CNN 1 's Confusion matrix for subtask C

three subtasks for macro F1 score. Although CNN 2 performs slightly worse, this may not be due to the structure itself, CNN 2 is currently trained the traditional way (updates all weights as once) but it may be necessary for the branches to be trained separately. This requires further testing in the future.

6 Related Work

Offensive language detection and classification has become increasingly relevant in recent years with the rise of social media. Subsequently, researchers have also begun to look at aggression, cyberbullying, hate speech, and abusive language identification.

Cyberbullying. Cyberbullying detection has been approach by several teams. Dinakar et al. (2011) show that binary classifiers for individual labels outperforms multi-label classifiers. Xu et al. (2012) demonstrate that social media is a rich environment for studying cyberbulling with NLP. Dadvar et al. (2013) show the effectiveness of including context around a comment.

Abusive Language. Abusive language has also seen increase in study. Nobata et al. (2016) construct a machine learning algorithm and test with different lexical features, outperforming at the time state-of-the-art methods. Mubarak et al. (2017) expand abusive language identification to Arabic social media. Fišer et al. (2017) propose a legal framework, dataset and anotation schema for abusive online language in Slovene. Su et al. (2017) propose a system which can not only detect, but also rephrase abusive language in Chinese. Waseem et al. (2017) propose breaking abusive language identification into further subtasks. Founta et al. (2018) leverage crowd sourcing to produce a large (80,000) annotated data set of abusive Twitter language.

Hate speech. Hate speech identification has come to the forefront for research as it deals with current hot button issues (e.g. racism, sexism). Schmidt and Wiegand (2017) and Fortuna and Nunes (2018) compile a surveys of current hate speech detection.

Other teams have brought to question how we view and handle hate speech. Ross et al. (2016) show the difficulty of annotating hate speech and propose handling classification as non-binary. Malmasi and Zampieri (2017) establish lexical baselines for hate speech detection by applying supervised classification methods and Malmasi and Zampieri (2018) show the problems which can arise when distinguishing profanity from hate speech. ElSherief et al. (2018) further look to understand hate speech by looking into the target of hate speech (i.e. at a individual or more general group).

Machine learning classifiers are leveraged in this field as well. Kwok and Wang (2013) employ a machine learning classifier to identify racist tweets. Burnap and Williams (2015) test a machine learning system on different n-gram features to identify hate speech on Twitter. Tulkens et al. (2016) use hate speech dictionaries along with support vector machines to identify racism on Dutch social media. Schofield and Davidson (2017) demonstrate three standard methods for producing features for text classification, targeting specifically the problem of automatic hate speech identification.

Subsequently, deep learning is seen in hate speech detection as well. Djuric et al. (2015) train and leverage comment embeddings to help identify hate speech. Gambäck and Sikdar (2017) show the effectiveness of convolutional neural networks (CNN) when identifying hate speech. Zhang et al. (2018) identify hate speech using a convolution-GRU based deep neural network.

Offensive Language. Offensive language identification aims to broaden the scope of negative language identification. Wiegand et al. (2018) proposed and ran a GermEval task similar to OffensEval, which had participants classify offensive language as offensive or other, then further classify the offensive tagged language.

7 Conclusion/Future Work

We have proposed and tested two different CNN architectures for identifying offensive language. Future work would aim to improve the current CNN design by testing different word windows and training techniques. Furthermore, since deep learning performs better with large amounts of training data, increasing the training data, perhaps even with silver standards if not gold, should help further improve the system's predictions.

References

2018. Cyber Bullying Statistics.

Pete Burnap and Matthew L Williams. 2015. Cyber hate speech on twitter: An application of machine classification and statistical modeling for policy and decision making. *Policy & Internet*, 7(2):223–242.

Maral Dadvar, Dolf Trieschnigg, Roeland Ordelman, and Franciska de Jong. 2013. Improving cyberbullying detection with user context. In *Advances in Information Retrieval*, pages 693–696. Springer.

Karthik Dinakar, Roi Reichart, and Henry Lieberman. 2011. Modeling the detection of textual cyberbullying. In *The Social Mobile Web*, pages 11–17.

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 Companion*, pages 29–30. International World Wide Web Conferences Steering Committee.

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*.

Darja Fišer, Tomaž Erjavec, and Nikola Ljubešić. 2017. Legal Framework, Dataset and Annotation Schema for Socially Unacceptable On-line Discourse Practices in Slovene. In *Proceedings of the Workshop Workshop on Abusive Language Online (ALW)*, Vancouver, Canada.

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

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. *arXiv preprint arXiv:1802.00393*.

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.

A. Goslin. 2018. Fortnite has 78.3 million monthly players, according to Epic.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882*.

Irene Kwok and Yuzhou Wang. 2013. Locate the hate: Detecting Tweets Against Blacks. In *Twenty-Seventh AAAI Conference on Artificial Intelligence*.

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.

Hamdy Mubarak, Darwish Kareem, and Magdy Walid. 2017. Abusive Language Detection on Arabic Social Media. In *Proceedings of the Workshop on Abusive Language Online (ALW)*, Vancouver, Canada.

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.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.

Björn Ross, Michael Rist, Guillermo Carbonell, Benjamin Cabrera, Nils Kurowsky, and Michael Wojatzki. 2016. Measuring the Reliability of Hate Speech Annotations: The Case of the European Refugee Crisis. In *Proceedings of the Workshop on Natural Language Processing for Computer-Mediated Communication (NLP4CMC)*, Bochum, Germany.

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.

Alexandra Schofield and Thomas Davidson. 2017. Identifying Hate Speech in Social Media. *XRDS: Crossroads, The ACM Magazine for Students*, 24(2):56–59.

A. Smith and M. Anderson. 2018. Social Media Use in 2018.

Huei-Po Su, Chen-Jie Huang, Hao-Tsung Chang, and Chuan-Jie Lin. 2017. Rephrasing Profanity in Chinese Text. In *Proceedings of the Workshop Workshop on Abusive Language Online (ALW)*, Vancouver, Canada.

D. Terdiman. 2018. Heres How Facebook Uses AI To Detect Many Kinds Of Bad Content.

Stéphan Tulkens, Lisa Hilte, Elise Lodewyckx, Ben Verhoeven, and Walter Daelemans. 2016. A Dictionary-based Approach to Racism Detection in Dutch Social Media. In *Proceedings of the Workshop Text Analytics for Cybersecurity and Online Safety (TA-COS)*, Portoroz, Slovenia.

Zeerak Waseem, Thomas Davidson, Dana Warmsley, and Ingmar Weber. 2017. Understanding Abuse: A Typology of Abusive Language Detection Subtasks. In *Proceedings of the First Workshop on Abusive Langauge Online*.

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*.

Jun-Ming Xu, Kwang-Sung Jun, Xiaojin Zhu, and Amy Bellmore. 2012. Learning from bullying traces in social media. In *Proceedings of the 2012 conference of the North American chapter of the association for computational linguistics: Human language technologies*, pages 656–666. Association for Computational Linguistics.

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.