Neural Networks for Offensive Language detection

Talha Anwar
National University of Computer and Emerging Sciences,Pakistan
i181509@nu.edu.pk

Zeeshan Ali
National University of Computer and Emerging Sciences,Pakistan
i181411@nu.edu.pk

Abstract

Usage of offensive language need to be controlled on social media. There is a need of robust and accurate algorithm that can identify whether a sentence written on social media has an offensive word or not. We presented different approached based on convolutional neural net, recurrent neural net and transformers. Our based model uses count vectorizer to extract features and logistic regression to classify them. Our base model set an f1 score of 0.7. We tried different embeddings such as GloVe and FastText. Using glove embedding we got highest macro f1 score of 77% , and using fasttext f1 macro is 72%. Combining both embedding, and using bidirectional gated recurrent unit our f1 macro increase upto 80%. Using Temporal Convolution Network with attention mechanism , we got macro f1 score of 0.79. Using transformers we got f1 macro of 0.79.

Categories and Subject Descriptors CR-number [*subcategory*]: third-level

Keywords Transformers, LSTM, GRU, Attention Mechanism, Temporal Convolution Networks

1. Problem statement

Classification of offensive language on twitter

2. Introduction

With growing and easy access of social media without check and balance, ethical and moral values are some how put aside. One of the main issue is abusive behaviours, cyber bullying and offensive language. Natural language community is trying to detect such behaviour using machine learning and deep learning. One of the main issue in such task is to gather label data and classifying that data into offensive or not.

(Zampieri et al., 2019) [6] organized a competition and provided labeled data. This data has tweets annotated for offensive content using a new-grained three-layer annotation scheme. These three annotations are

- Level A - Offensive language identification;
- Level B - Automatic categorization of offense types;
- Level C - Offense target identification.

Level A is about whether a tweet is offensive or not. Level B tells about whether the offensive tweet is threat to individual or a particular group. Or they are generally offensive. Level 3 is about to whom this tweet is targeting. Is it targeting to famous personality or a group of particular ethnicity or religion. There are around 14100 tweets in this data set. They got F1 score of 80% using CNN, 75% using BiLSTM, 69% using SVM for level A. For our work, we chose Level A only. Liu [1] managed to score 0.82 macro f1 score using BERT-3 model. If we compare our self we stand on top 10 as we managed to score 0.8 macro f1 score. We tried Bidirectional GRU with averaging of fasttext and glove embeddings. We also concatenated the average pool layer and max pool layer. Both these technique help to increase f1 score. Same averaging technique is applied on temporal convolution network and transformers. Using temporal convolution network we got macro f1 score of 0.78 and with transformers we got macro f1 score of 0.79.

We averaged FASTTEXT and GloVe embeddings and feed them to bi directional Gated recurrent unit (GRU). Output of GRU is passed through global average pooling and global max pooling. Both these layers are concatenated and passed to fully connected layer followed by dense layer.

In the rest of this paper, we organize the content as follows: related work of Offensive Language detection is stated in section 3; section 4 introduces details of preprocessing, and the methodology of our models; experimental results are discussed in section 5.

Our implementation is available on
github[1]

3. Related work

A lot of data is required to handle this problem. (Sood et al., 2012)[2] collected 1.6 million comments from yahoo, among 6500 were labeled by 221 people. Several machine learning algorithms was trained on this data considering n-gram as feature. (Chen et al., 2012)proposed a model to examine individual abusive behavior based upon posting pattern. (Xu et al., 2012)[3] apply sentiment analysis to detect bullying roles in tweets using Latent Dirichlet allocation to identify bullying or not in the texts. (Nobata et al.,2016)[4] use n gram, linguistic feature (length of words, not capital words, no of token with non-alpha character, synthetic feature (parent of node, POS of parent), Distributional Semantics Features (word embedding such as word2vec). They got accuracy of 90%F-score of 78% using all above features on WWW2015 dataset. (Malmasi et al., 2018)[5] use n gram, skip grams,clustering based word representation. They tried single classifier to ensemble classifier and got an accuracy of 80%. The data set used contain 14509 English tweets labeled as offensive, non-offensive and OK.

[1] `https://github.com/talhaanwarch/Offensive-Language-Detection/`

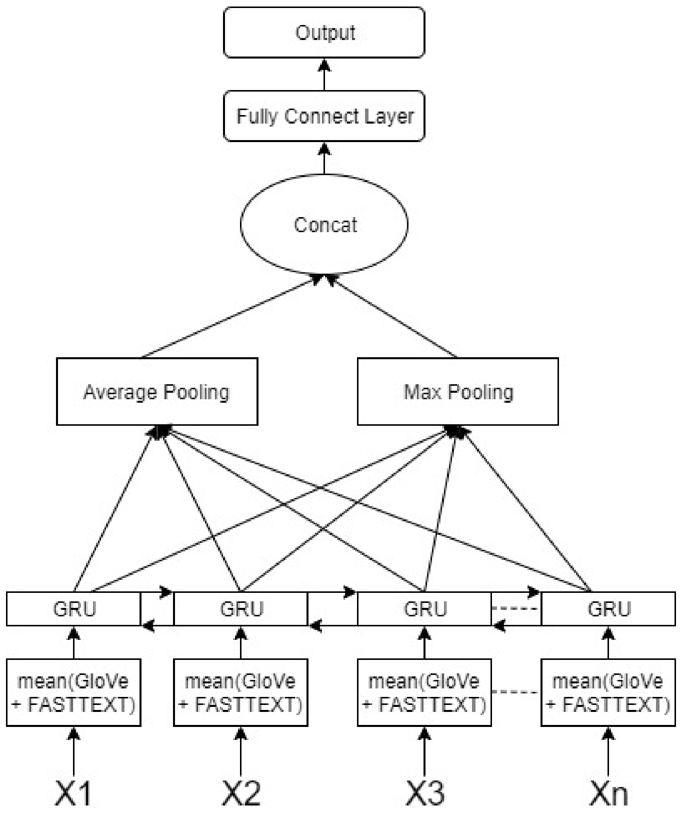

Figure 1. BiGRU with mean embeddings

(Mitrovic et al., 2019)[11] use the same OLID dataset and TF IDF as feature use machine learning classifier. They limit their feature space to 10,000 most frequently use bigram, trigram and unigram. They created word2vec embedding and feed them a bidirectional GRU model. They got 70.20 F1 score on SVM, and 76.2 on biGRU. They just considered level A for classification purpose.

(Doostmohammadi et al., 2019)[7] proposed approach includes deep model consisting of Convolutional Neural Network for character-level processing and Recurrent Neural Network (RNN) for word-level processing, and some preprocessing to address the problem of identifying, categorization offensive language in social media in three sub tasks that includes whether the content is offensive or not, whether it is targeted or not, if it is targeted then check whether its towards individual or group. As the offensive comments are likely to have unorthodox writing styles, which involve irregular issue that leads to tokenization issue So using character-level is beneficial in this case. This paper also performed other experiments included a Support Vector Machine (SVM) with TF-IDF, similarly count features and another SVM with BERT-encoded sentences as input, both have lower performances while comparing with the deep model. For effective results, they performed preprocessing which involves replacing obfuscated offensive words with their correct form and also tokenization of tweet. The CNN used here is consist of four consecutive sub layers including Convolution layer consisting of 64 filters with kernel size of 2 , max-pooling layer with pool size and stride of 2, third and fourth are similar but involve 128 filters instead of 64 while using similar pooling layer. There got macro F1 score of 77.9%.

(Torres et al., 2019)[8] used Convolutional Neural Network for offensive tweets identification and Fast-Text pre-trained word embedding as learning representation. This paper only focuses on first task that is offensive language detection The aim is to perform effective classification as unstructured and noisy nature of user generated content. To examine the effectiveness of proposed solution, CNN is compared with other baseline models, such as linear regression and several other neural models includes sequential models such as LSTM and BiLSTM. The CNN model is evaluated with different word embeddings such as Random unified initialized, Word2Vec and FastText. Experiments showed that the FastText embeddings provide the best results with F1 score of 71.86. While during training they fine-tuned the embedding layer for each type of embeddings. The model outperformed sequential model and other baseline with accuracy of 0.8105 while the actual reason due to which CNN model perform better than sequential models is noisy and unstructured form of the tweets. By using different variations of CNN such as multi-channel and multi-view architectures and also better learning representations based on Deep contextualized embedding like BERT can enhance the performance as well. (Garain et.al., 2019)[12] used similar dataset OLID (Offensive Language Identification Dataset) for detecting offensive language for tweets, where two specific target audience are immigrants and women. For identifying tweet is offensive or not, whether it is directed towards individual, group or entity used Bi-directional LSTM based neural network for better prediction of class exist in given dataset. The given dataset includes original tweets along with corresponding labels. The Subtask A includes two type of labels such as OFF (offensive) and NOT (Not Offensive) and similarly Subtask 2 includes two labels whether tweet is targeted towards individual/group/other or its just profanity but not targeted (UNT). Firstly, they have converted the tweets into a sequence of words and then execute neural network-based algorithm on processed tweets, processing involves removing of mentions, removing of punctuation's, removing URLs and Extracting words from hash-tags. Therefore, the tweets are considered as sequence of words and for capturing the information from both past and future context they used Bi-Directional LSTM. To learn word embedding the input tweet is passed through an embedding layer which transforms the tweet into a 128-length vector and then used two bidirectional LSTM layers containing 64 units each followed by the final output layer of neurons with SoftMax activation. Here each neuron is used to predict the label as mentioned in the dataset. They concluded that the given dataset is very skewed therefore the validation set is included to avoid overfitting and class weights were assigned to the different classes present in the data. Results shows that Bi-directional LSTM performance is macro F1-score 0.4650 while accuracy is 0.5651 for task A. Similarly, performance of model against task B, is BiLSTM(thresh=0.40) macro F1-score 0.5796 and Accuracy is 0.8833. The main drawbacks of proposed approach are that the model is based on simple architecture there must include some features and secondly, model only uses data provided by dataset and does not using external data which may leads to poor results.

Pelicon et. al [10]presents techniques used by Embeddio team to efficiently handle the more unbalanced dataset of OLID, they trained different models for each subtask to get better score. For subtask A, they used pretrained Bert model for distinguishing between Offensive and non-offensive post while model is fine-tuned using 24 layer of size 1024 16 self-attention head are used on given dataset. Their proposed Bert model with fine-tuned on given dataset achieved 0.808 macro-averaged F1-score and place and fourth position in official ranking of Sem-Eval 2019. We check the online implementation of Pelicon and found that weighted averaged F1 score was calculated instead of macro f1 score.

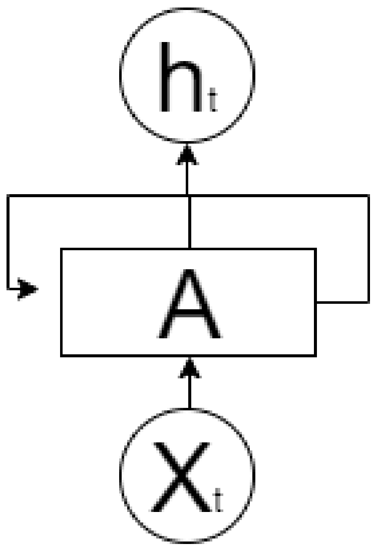

Figure 2. RNN folded

4. Methodology

4.1 Preprocessing

We removed punctuation, URLs, and @USER words from all tweets. All words converted to lower words. We removed stop words after tokenizing of sentences, and apply lematization. We noticed that without removing stopwords and doing lematization better result can be achieved. This is becuase removing stop words and doing lematization may remove the negation words.

4.2 Base Model

Simple logistic regression is used to classify offensive vs non offensive tweets. CountVectorizer is used to convert sentences to vectors. It counts the occurrence of words in a tweet.

4.3 Recurrent Neural Network

Recurrent Neural Network In traditional neural network it is assumed that all inputs and outputs are independent of each other. One input does not consider other input during feed forward or backward propagation. If we want to predict the next word in a sentence, we should know all previous word, so we can precisely tell what next word can be. Traditional Neural network like convolutional neural network are bad in this approach. That why we need recurrent neural network (RNN). RNN consider the previous input and stored its value in a hidden state as shown in figure 2 .Use the hidden state and compute the next output. So it knows every thing that happened before the event occurs.

We used Long Short term memory (LSTM) and Gated Recurrent Unit (GRU) variants of recurrent neural network. We have tried both unidirectional and bidirectional LSTM and GRU.

Let suppose if we want to know whether an offensive sentence *That because you are an old man* is directed to some one or not. We should know all the words before the abusive words. If there is some subjective word, our model will predict it as direct offensive sentence otherwise in-directed offensive word. Figure 3 shows it graphically. Saying an old man is not an offensive word, but saying it after 'That Because' make it offensive. So RNN know what words come before an offensive word in order to make sentence target to some one or not.

In some case the object of offensive word can be after offensive word, for example. *I love her but I hate you*, so if we dont know the word *you*, our RNN will consider the word *hate* for the word *her* for on this we need to feed data to recurrent neural network in backward way. We can use bidirectional neural network which is nothing but combination of two neural net. In one neural network feed forward is in regular way. In second neural network, input start from end of sentence and move to beginning of sentence as shown in figure 4

Unfortunately, if sentence is too long, RNN is not capable to handle this because of vanishing gradient problem. So we need variants of RNN called as Long Short Term Memory Network(LSTM). Gated Recurrent Unit is another modified form of LSTM. LSTMs are capable of learning from long dependencies without much struggle.

4.3.1 Main RNN Architectures

Five types of layers were used

- Embedding
- LSTM/GRU
- Dropout
- Global Maximum Layer
- Dense layer

In all of RNN variants architecture we initialized with embedding layer, followed by one or two LSTM/GRU layer. After that there is global maximum pooling layer. Two dense layer with sigmoid or RelU activation function is used. A dropout layer followed by output dense layer remain same in all cases. RMSprop/Adam optimizer is used for all architectures. As we have unbalance classes, so class weights are added to over come this issue.

4.3.2 FASTTEXT GLOVE averaged embeddings

Textual data in converted to Fasttext and Glove embeddings. Both embeddings are averaged and feed to bidirectional gated recurrent unit with 150 units. biGRU is followed by a concatenation of average pooling and max pooling. Prior to output layer, there are two dense layer with 64 and 32 neurons. No lemmatization was performed and none of stop words were removed. The averaging of fasttext and glove embeddings provide us the best results. We tried adding attention layer, but that did not provide much better result.

4.3.3 Bidirectional LSTM and GRU

biLSTM Embedding layer input dimension is 10,000, output dimension is 50, input length is 200. After that two bidirectional LSTM layer with 64 and 32 neurons were used. Global Maximum pooing followed by two dense layer consist of 100 and 20 neurons with ReLu as activation function is used in both layers.

biGRU Embedding layer input dimension is 10,000, output dimension is 50, input length is 200. After that one bidirectional GRU layer with 64. Global Maximum pooing followed by two dense layer consist of 50 and 20 neurons with ReLu as activation function is used in both layers.

FASTTEXT biGRU Number of features are 1000, max length of input is 200, two bidirectional gated recurrent unit with 100 and 50 units were used followed by concatenation of max and average pooling. Two fully connect layers with 50 and 30 layers are used before output layer

4.3.4 Unidirectional LSTM and GRU

GLove based GRU After spatial-dropout layer, GRU layer with 100 unit is used. Dense layer has 50 and 30 neurons having sigmoid and RELU activation functions shows GloVe embedding based Uni Directional Gated Recurrent Unit layers and parameters **GLove based LSTM** After spatialdropout layer, GRU layer with 100 unit

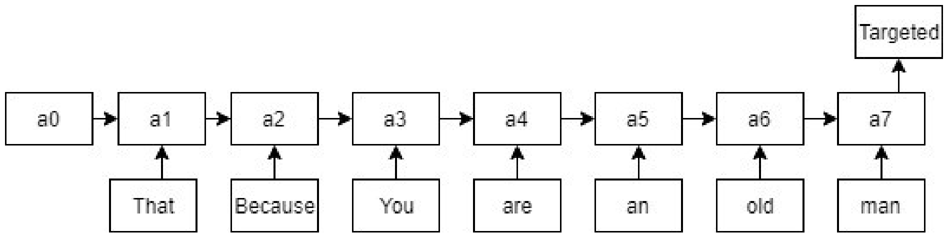

Figure 3. RNN many to one

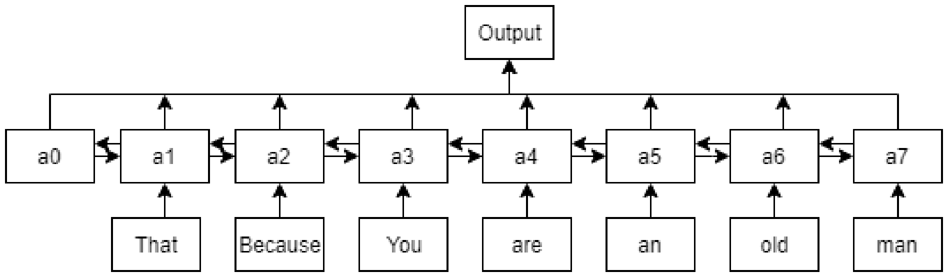

Figure 4. biRNN many to one

is used. Dense layer has 80 and 50 neurons having sigmoid and reLu activation functions

4.4 Convolution Neural Network

4.4.1 Simple CNN

We tried simple CNN with 3 convolution layer, each followed by by max pooling layer. Number of filter in each convolution layer is 128. Filter size is 5 in all layers. There is one fully connected layer with 128 neurons before output layer. Glove embedding with 100 dimension is used. Max number of features are 1000 and max length is 200.

4.4.2 Temporal convolution network

We tried Temporal convolution network [13] to classify our offensive language task because of varying input length and causal in nature.

- The architecture can take a sequence of any length and map it to an output sequence of the same length, just as with an RNNs.
- The convolutions in the architecture are causal, meaning that there is no information "leakage" from future to past.

TCN uses a 1D fullyconvolutional network (FCN) architecture, where each hidden layer is the same length as the input layer to feed input of varying length. Zero padding of length (kernel size 1) is added to keep subsequent layers the same length as previous ones. TCN uses causal convolutions, convolutions where an output at time t is convolved only with elements from time t and earlier in the previous layer. As compared to RNN, TCN are parallel and more flexible receptive fields. In TCN instead of feeding textual data or single embeddings, we averaged glove, word2vec and fasttext embeddings, so we can cover embeddings of all words occuring in our dataset.Two TCN layers with 128 and 64 units having dilation5 of 1,2,4 is used. Average Pooling and Max pooling are concatenated to give more flexible learning to our network. Then two fully connected layers with 64 and 32 neurons are used. Bath size is 128 and epochs are 10 for this TCN network.

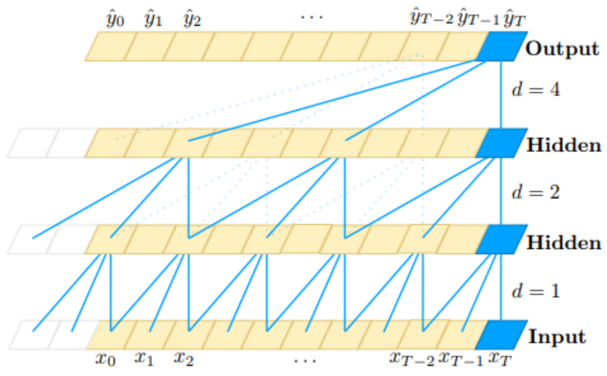
$\hat{y}_0$ $\hat{y}_1$ $\hat{y}_2$ ... $\hat{y}_{T-2}$ $\hat{y}_{T-1}$ $\hat{y}_T$
Output
$d = 4$
Hidden
$d = 2$
Hidden
$d = 1$
Input
x_0 x_1 x_2 ... x_{T-2} x_{T-1} x_T

Figure 5. TCN dilation

4.4.3 Attention based temporal Convolution Network

In order pay more attention to meaning ful embeddings, we use attention mechansim in hybrid with TCN.

4.5 Transformers

Transformers are considered as an alternative to RNNs. RNNS take input one by one and travel sequentially. Transformers take all input at once using encoder and give output from decoders. Transformers are more machine friendly and take less time as compared to RNNs. Transformers even know which word to give more attention and which to less.

5. Evaluation and Experiments

6. Result

6.1 RNN variants

Table 1 shows that using simplest logistic regression we got an accuracy of 78% and macro F1 as 70%, weighted accuracy is 77%

	Precision	Recall	F1 Score	Support
Offensive	0.67	0.45	0.54	240
Not Offensive	0.81	0.91	0.86	620
Macro Average	0.74	0.68	0.70	860
Weighted Average	0.77	0.78	0.77	860

Table 1. Base Model using simple logistic regression

Table 2 shows that using GloVe embedding weighted Gated Recurrent Unit shows macro F1 score of 77%. Accuracy is 80.4% .

	Precision	Recall	F1 Score	Support
Offensive	0.63	0.74	0.68	240
Not Offensive	0.89	0.83	0.86	620
Macro Average	0.76	0.78	0.77	860
Weighted Average	0.82	0.80	0.81	860

Table 2. GloVe Unidirectional Gated Recurrent Unit

Table 3 shows result of LSTM macro F1 Score of 77%, weighted F1 score is 81
% and accuracy is 81.2% which is a bit higher than GRU model. The weight matrix is based on GloVe Embeddings.

	Precision	Recall	F1 Score	Support
Offensive	0.66	0.68	0.67	240
Not Offensive	0.87	0.87	0.87	620
Macro Average	0.77	0.77	0.77	860
Weighted Average	0.81	0.81	0.81	860

Table 3. GloVe embedding based UniDirectional LSTM

Table 4 shows result of bi directional LSTM with Glove Embedding weights. Using biLSM we got F1 macro as 76% and weighted F1 as 81%. Accuracy is round of 80% in this case.

	Precision	Recall	F1 Score	Support
Offensive	0.66	0.65	0.65	240
Not Offensive	0.86	0.87	0.87	620
Macro Average	0.76	0.76	0.76	860
Weighted Average	0.81	0.81	0.81	860

Table 4. GloVe embedding based biDirectional LSTM

Table 5 shows result of bi directional GRU with gloVe embeddings. Using biGRU we got F1 macro as 76% and weighted F1 as 81%. Accuracy is round of 80% in this case.

Table 6 shows result of bi directional GRU. Using biGRU we got F1 macro as 71% and weighted F1 as 77%. Accuracy is 77% in this case.

Table 7 shows result of bi directional LSTM. Using biLSTM we got F1 macro as 70% and weighted F1 as 75%. Accuracy is 75% in this case.

Table 8 shows using fast text embedding we got an macro average f1 score of 0.72 and weightage average is 0.78. This is far less than glove embedding

6.2 CNN results

Table 9 Our simple CNN give macro f1 of 70% which is low because we did not used goof number of features.

	Precision	Recall	F1 Score	Support
Offensive	0.64	0.70	0.67	240
Not Offensive	0.88	0.85	0.86	620
Macro Average	0.76	0.77	0.76	860
Weighted Average	0.81	0.80	0.81	860

Table 5. GloVe embedding based biDirectional GRU

	Precision	Recall	F1 Score	Support
Offensive	0.58	0.61	0.59	240
Not Offensive	0.85	0.83	0.84	620
Macro Average	0.71	0.72	0.71	860
Weighted Average	0.77	0.77	0.77	860

Table 6. biDirectional GRU

	Precision	Recall	F1 Score	Support
Offensive	0.55	0.62	0.58	240
Not Offensive	0.85	0.80	0.82	620
Macro Average	0.70	0.71	0.70	860
Weighted Average	0.76	0.75	0.75	860

Table 7. biDirectional LSTM

	Precision	Recall	F1 Score	Support
Offensive	0.63	0.54	0.58	240
Not Offensive	0.83	0.88	0.85	620
Macro Average	0.73	0.71	0.72	860
Weighted Average	0.78	0.78	0.78	860

Table 8. FASTTEXT biGRU

Table 10 shows our macro f1 score of 0.78. Where TCN average macro f1 score is 0.82. TCN is bit promissing our simple CNN used.

6.3 Transformer results

Table 11 shows that our result with transformers are quite good. we got f1 macro of 0.79.

A. Appendix Title

This is the text of the appendix, if you need one.

Acknowledgments

Acknowledgments, if needed.

References

[1] Liu, Ping, Wen Li, and Liang Zou. "Nuli at semeval-2019 task 6: Transfer learning for offensive language detection using bidirectional transformers." Proceedings of the 13th International Workshop on Semantic Evaluation. 2019.

[2] Sara Owsley Sood, Elizabeth F. Churchill, and Judd Antin. 2012. Automatic identification of personal insultson social news sites. Journal of the American Society for Information Science and Technology 63(2):270–285.

[3] Ying Chen, Yilu Zhou, Sencun Zhu, and Heng Xu. 2012. Detecting offensive language in social media toprotect adolescent online safety. In Proceedings of the 2012 ASE/IEEE International Conference on SocialComputing and 2012 ASE/IEEE International Conference on

	Precision	**Recall**	**F1 Score**	**Support**
Offensive	0.58	0.54	0.56	240
Not Offensive	0.83	0.85	0.84	620
Macro Average	0.70	0.69	0.70	860
Weighted Average	0.76	0.76	0.76	860

Table 9. simple CNN

	Precision	**Recall**	**F1 Score**	**Support**
Offensive	0.65	0.75	0.70	240
Not Offensive	0.90	0.84	0.88	620
Macro Average	0.77	0.80	0.79	860
Weighted Average	0.83	0.82	0.82	860

Table 10. TCNN

	Precision	**Recall**	**F1 Score**	**Support**
Offensive	0.68	0.72	0.70	240
Not Offensive	0.89	0.87	0.88	620
Macro Average	0.79	0.80	0.79	860
Weighted Average	0.83	0.83	0.83	860

Table 11. Transformers

Privacy, Security, Risk and Trust. IEEEComputer Society, Amsterdam, The Netherlands, pages 71–80

[4] Nobata, Chikashi, et al. ”Abusive language detection in online user content.” Proceedings of the 25thinternational conference on world wide web. International World Wide Web Conferences Steering Committee,2016.

[5] Malmasi, Shervin, and Marcos Zampieri. ”Challenges in discriminating profanity from hate speech.” Journalof Experimental & Theoretical Artificial Intelligence 30.2 (2018): 187-202

[6] Zampieri, M., Malmasi, S., Nakov, P., Rosenthal, S., Farra, N.and Kumar, R., 2019. Predicting the Type and Target of OffensivePosts in Social Media. Proceedings of NAACL-HLT 2019, pages1415–1420

[7] Doostmohammadi, E., Sameti, H. and Saffar, A., 2019, June. Gh-merti at SemEval-2019 Task 6: A Deep Word-and Character-basedApproach to Offensive Language Identification. In Proceedings ofthe 13th International Workshop on Semantic Evaluation (pp. 617-621). 2019 Association for Computational Linguistics.

[8] Torres, J. and Vaca, C., 2019, June. JTML at SemEval-2019 Task6: Offensive Tweets Identification using Convolutional Neural Net-works. In Proceedings of the 13th International Workshop on Seman-tic Evaluation (pp. 657-661)

[9] Garain, A. and Basu, A., 2019, June. The Titans at SemEval-2019Task 6: Offensive Language Identification, Categorization and TargetIdentification. In Proceedings of the 13th International Workshop onSemantic Evaluation (pp. 759-762)

[10] Pelicon, Andraž, Matej Martinc, and Petra Kralj Novak. ”Embeddia at SemEval-2019 Task 6: Detecting Hate with Neural Network and Transfer Learning Approaches.” Proceedings of the 13th International Workshop on Semantic Evaluation. 2019.

[11] Mitrovic, Jelena, Bastian Birkeneder, and Michael Granitzer. ”nlpUPat SemEval-2019 Task 6: A Deep Neural Language Model for OensiveLan-guage Detection.” Proceedings of the 13th International Workshop[1] on Semantic Evaluation. 2019

[12] Garain, A. and Basu, A., 2019, June. The Titans at SemEval-2019Task 6: Offensive Language Identification, Categorization and TargetIdentification. In Proceedings of the 13th International Workshop onSemantic Evaluation (pp. 759-762).

[13] Bai, Shaojie, J. Zico Kolter, and Vladlen Koltun. ”An empirical evaluation of generic convolutional and recurrent networks for sequence modeling.” arXiv preprint arXiv:1803.01271 (2018).