Comparative Analysis for Toxic Comment Classification

Ahmed Jameel Ismael[1], Herlina Abdul Rahim[1*], Udaya Mouni Boppana[2], Arpit Yadav[3] and Afia Zafar[4]

[1*]School of Electrical Engineering, Faculty of Engineering, Universiti Teknologi Malaysia, 81310 Skudai, Johor, Malaysia.
[2] Faculty of Computer Science and Information Technology, Universiti Tun Hussein Onn Malaysia, 86400 Parit Raja, Batu Pahat, Johor, Malaysia.
[3]College of Shri Vaishnav Vidyapeeth Vishwavidyalaya, Indore Madhya Pradesh. India.
[4]Department of Computer Science, NUTECH, Islamabad, Pakistan.

Corresponding author* email: herlina@utm.my
Accepted 3 March 2021, available online 31 March 2021

ABSTRACT

Due to the development in e-commerce, social media channels like twitter and Facebook flood of Information is poured into the Internet, while this is going to affect the quality of the people since these texts may include many levels of toxicity which results in online harassment, bullying, etc., There is paramount importance to monitor this data. Research and Industrial communities are trying to build an effective model to classify these toxic comments. Still, these attempts are not reaching sufficient levels. Luckily In recent days, there are advancements in hardware and data management techniques like Big Data that encourages computational approaches like Deep learning approaches which can improve the text classification. This paper focuses primarily on, various approaches for short toxic comment classification and the comparative experiments are set to evaluate their accuracy based on Area Under Curve (AUC) metric from "Wikipedia Talk Page Comments annotated with toxicity reasons" Dataset. Based on our experimental analysis, bidirectional GRU performed better as compared to other existing classification algorithms. These findings will be used as the benchmarking results for improvement of Toxic Comment Classification.

Keywords: Text classification; Artificial neural networks, Gated Recurrent Unit (GRU)

1. Introduction

Due to the evolution of the internet and online communication, we receive a tremendous amount of short comments. In order to process this large volume of data sophisticated text mining algorithms are required. This processing involves classification of comments into different levels based on the requirement, for example, topic categorization sentiment analysis [1]. etc., and this classification can be achieved using Natural Language processing. Text classification can be broadly dealing with assigning a set of documents with an appropriate label from a set of classes. Already numerable number of machine learning algorithms can classify these text documents with acceptable results. But mostly these algorithms are designed to deal with documents of large data which are facing hard to classify short comments with minimal shared context and while representing these documents in vector form leads to sparse matrices. Because of which, defining good similarity measures is not straight forward [2].

These comments might hide some hazards like sexual harassment, fake news, toxicity [3]. Few comments not only in results in verbal toxicity but also many personal attacks on reputed organizations and celebrities, bullying and targeting the self-respect of people. It may affect the individual's feelings and sometimes those individuals may commit suicides due these comments. These online comments may affect one's job security, organization revenues, etc. According to Wikimedia 54% of individuals who experienced this sort of harassment, bullying behavior expressed reduced participation in the particular project which occurred [4]. Due to which defining efficient similarity measures becomes difficult, for example, most famous word-frequency based algorithms results in degrading performance.

Basically, convolution neural networks are used when local spatial coherence needs to be considered often used in images. Recently, these convolution neural networks are applied to this classification of short messages discrete embedding [5] of words without using syntactic and semantic knowledge. These convnets can be applied at character level and word level, the work of Zhang et al [6] proves that character level embedding for convnets is efficient method. Recurrent ConvNets are also applied to this text classification without human crafted features [7] which outperforms the CNN. This model exploits the contextual information as the RNNs can preserve or memorize the previous inputs information.

Automatic suspicious comment identification in real-time is important. To control the adverse effects for net users. Jigsaw and Google started a project known as The Perspective [8] which uses Artificial Intelligence techniques to find on-line insults, abusive and harassment. Google provides an Application Programming Interface (API) called Perspective which allows developers to use the detector service running on Google's servers, to identify the toxic and abusive comments on social media. This API uses machine learning models to predict the impact of a comment on the individual and the level of toxicity involved. Admins and Instructors can use this score to give real-time feedback to commenter.

The main setback of those models is that they are not reliable, because sometimes a percentage of toxicity is not unpredicted or unidentified. The solution presented here is based upon the Kaggle competition. Kaggle is a platform where mathematicians, statisticians especially data scientists, data miners participate in competitions to compete by analysing the data or information models provided by trusted companies and users to find out the models of highest accuracy ratings.

In this paper we are going to use the dataset provided in the kaggle competition. Dataset provided by kaggle competition is from Wikipedia comments and these comments are annotated by human annotators for toxic behavior.

Different types of toxicity are:

a) threat
b) identity hate
c) obscene
d) toxic
e) insult
f) severe toxic

2. Data Preprocessing

Text can't be processed in its raw form which need to be converted into a form in which data can be processed. Representing textual sequences such as words and sentences is a fundamental component of natural language understanding systems.

Embeddings: Text data can be represented in many forms by using bag-of-words and representing them using one hot encoded vectors and Embeddings etc., Embedding is a mapping between discrete or categorical objects like words to vectors of continuous real numbers. Basically, embeddings are low-dimensional, continuous vector representations of discrete variables. these learned by neural networks. These embeddings have significance because they can reduce the dimensionality and semantically represent categories in the transformed space [9].

These can be trained by using our own specific domain data or by publicly available pretrained embeddings like glove, Word2Vec and Fast Text etc., According to the experiment work embeddings learned on domain specific data performs better than pretrained embeddings. We preprocessed the data using embedding, which was learned on Twitter text since our data is semantically near to the twitter data. As stated earlier these embeddings can be performed in various types eg., word level or character n-grams etc. According to the work of John Wieting, Mohit Bansal, Kevin Gimpel, Karen Livescu [10] character n-gram embeddings perform better than normal embeddings due to the fact that these embeddings will support new words also.

In order to convert the text into embedding vectors, prior we need to tokenize the data and then we index the data which will be fed to LSTM layer to get embeddings. Figure 1 shows the example of tokenizing the data followed by indexing.

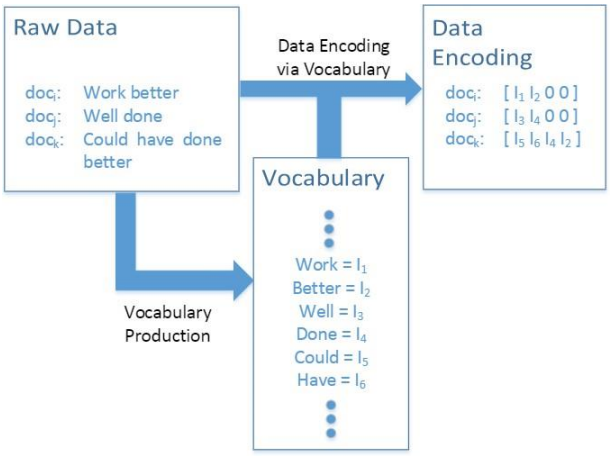

Figure. 1. Example of Tokenizing the data followed by indexing.

3. Learning Through GRU's

After converting the data into embedded vectors, these vectors will be fed to neural network. Bidirectional GRU was used and various other models like LSTMS CONVNETS, SVM and KNN etc.., Bidirectional GRU's was observed and it was performing better than other, as shown in Table 1.

Table 1. Performance of Bidorectional GRU's

Layer (type)	Output Shape	Param #
input_1 (InputLayer)	(None, 600)	0
embedding_1 (Embedding)	(None, 600, 300)	615600
conv1d_1 (Conv1D)	(None, 600, 120)	144120
max_pooling1d_1 (MaxPooling1	(None, 150, 120)	0
bidirectional_1 (Bidirection	(None, 150, 160)	96480
global_max_pooling1d_1 (Glob	(None, 160)	0
dense_1 (Dense)	(None, 70)	11270
dropout_1 (Dropout)	(None, 70)	0
dense_2 (Dense)	(None, 6)	426

Total params: 867,896
Trainable params: 867,896
Non-trainable params: 0

In the input layer each comment it is tokenized and indexed as shown in the Table 1 example later this index vector size is trimmed to length of 600 elements if any comment is having less than 600 elements then that vector is padded to satisfy the vector size of 600. This padding and trimming are required to keep the input size constant. Then these indexed vectors are converted into Embedding vectors of size 300. Then these vectors are sent to Convolution Layer followed by MaxPooling this step is to reduce the dimensions of the embedding vectors then the Resultant is fed to Bidirectional Gru Layer followed by MaxPooling. Finally, the resultant is fed to Neural Layer, which contains 70 neurons in it. Dropouts [11] was used in order to reduce the overfitting. Categorical Loss function with Adam optimizer [12] was used that combines the concepts of Momentum and RMSprop to accelerate the minimization of loss functions by gradient descent as shown in Figure 2 and 3.

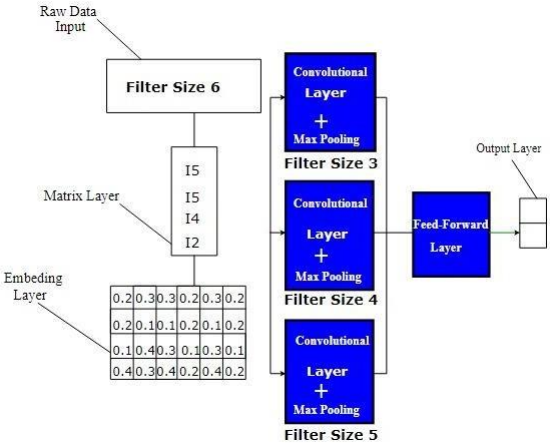

Figure.2. Example of Neural Network Architecture

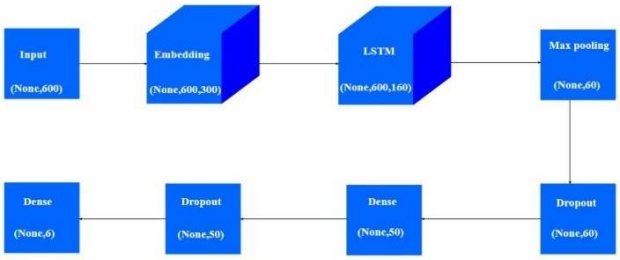

Figure.3. Flow of Model with number of Trainable Parameters at each

4. Results

The text Classification using the below classification approaches was be used and the accuracies for each classification model was determined. Finally, the model using binary cross-entropy loss, Adam optimizer and evaluation metric of Area Under Curve (AUC) was compiled in this project.

Area Under Curve (AUC) is the area under the Receiver Operating Characteristics (ROC) curve, drawn by plotting true positive rate against false positive rate. AUC is essentially the probability that the classifier/model will rank a randomly drawn positive case higher than a randomly drawn negative case, all assuming "positive" ranks higher than "negative."[13]. By fitting the model with a mini-batch size of 256 and over ten epochs. 10% of data was used as a training data out as a validation set. This validation set is drawn before model training. The reason behind this is that, to control the splitting with a set random seed so that our result is somewhat reproducible. Early stopping was be used on validation loss, so that training is stopped after the epoch that it detects the model is overfitting the training set. The performance as shown in Table 2.

Table 2: Accuracy for various classification approaches

	Accuracy
RNN(GRU)	98.87
CNN	0.895
KNN	0.697
LDA	0.808
NB	0.719
SVM	0.811
Bidirectional LSTM+ Attention	0.9778

5. Conclusion

To automate the classification of toxic comments. In this paper still, by using basic preprocessing, with minimal processing can be able to achieve significant accuracy. Started with basic machine learning models like svm, probabilistic model naive bayes, it does not give good results which proves that probabilistic approaches are not suitable for text classification. By taking a step forward, tried with deep learning approaches, these techniques will automatically find the features which increases the possibility of reaching higher accuracies and finally can able to get approximately 99% accuracy. Gated recurrent units layer proves to be more efficient at training but performs slightly worse than the baseline model with LSTM layer. One of the significant challenges researchers in machine learning face is the limitation of "high quality" data. On a closing note, further improvements can be made to improve the model, is to perform additional hyperparameter tuning, which will most definitely prove beneficial.

Acknowledgement

The authors wish to thank the Ministry of Higher Education of Malaysia (MOHE) and Universiti Teknologi Malaysia (UTM) for financing this research with Vote No. 20H62 and 08G47.

References

[1] C. C. Aggarwal and C. X. Zhai, *A survey of text classification algorithms*, in Mining Text Data, vol. 9781461432, Springer US, 2012, pp. 163–222.

[2] X. Quan, G. Liu, Z. Lu, X. Ni, and L. Wenyin, *Short text similarity based on probabilistic topics*, Knowl. Inf. Syst., vol. 25, no. 3, pp. 473–491, Dec. 2010, doi: 10.1007/s10115-009-0250-y.

[3] Maeve Duggan. 2014. Online harassment. Pew Research... - Google Scholar. .

[4] E. Wulczyn, N. Thain, and L. Dixon Jigsaw, Ex Machina: *Personal Attacks Seen at Scale*, dl.acm.org, pp. 1391–1399, 2017, doi: 10.1145/3038912.3052591.

[5] Y. Kim, *Convolutional Neural Networks for Sentence Classification*, EMNLP 2014 - 2014 Conf. Empir. Methods Nat. Lang. Process. Proc. Conf., pp. 1746–1751, Aug. 2014.

[6] Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. *Character-level convolutional networks for text classification. In Advances in neural information processing systems*. 649–657. - Google Search.".

[7] C. Xu et al., *Recurrent convolutional neural network for sequential recommendation*, in The Web Conference 2019 - Proceedings of the World Wide Web Conference, WWW 2019, May 2019, pp. 3398–3404, doi: 10.1145/3308558.3313408.

[8] H. Hosseini, S. Kannan, B. Zhang, and R. Poovendran, *Deceiving Google's Perspective API Built for Detecting Toxic Comments*, 2017.

[9] https://towardsdatascience.com/neural-network-embeddings-explained-4d028e6f0526 - Google Search." .

[10] R.-M. Karampatsis and C. Sutton, "*SCELMO: SOURCE CODE EMBEDDINGS FROM LANGUAGE MODELS*," 2020.

[11] N. Srivastava, G. Hinton, A. Krizhevsky, and R. Salakhutdinov, *Dropout: A Simple Way to Prevent Neural Networks from Overfitting*, J. Mach. Learn. Res., vol. 15, pp. 1929–1958, 2014, Accessed: Feb. 08, 2021. [Online]. Available: https://www.jmlr.org/papers/volume15/srivastava14a/srivastava14a.pdf?utm_campaign=buffer&utm_content=buffer79b43&utm_medium=social&utm_source=twitter.com.

[12] D. P. Kingma and J. L. Ba, Adam: *A method for stochastic optimization*, 2015.

[13] Fawcett T., *An introduction to ROC analysis*, Pattern Recognit. Lett., vol. 27, no. 8, pp. 861–874, 2006, Accessed: Feb. 08, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S016786550500303X?casa_token=K4syYHYx1UEAAAAA:NcvOFseB67ESYnUPGRi5qMdzfkkCjc1Pjx6IKosrHKdBGn7DQRn2AGognE--O4WJWfzn6IPIqnM.