ssn_nlp at SemEval-2020 Task 12: Offense Target Identification in Social Media using Traditional and Deep Machine Learning Approaches

Thenmozhi D., Nandhinee PR, Arunima S, Amlan Sengupta
Department of CSE, SSN College of Engineering, India
theni_d@ssn.edu.in
{nandhinee16066,arunima17016,amlan17008}@cse.ssn.edu.in

Abstract

Offensive language identification (OLI) in user generated text is automatic detection of any profanity, insult, obscenity, racism or vulgarity that is addressed towards an individual or a group. Due to immense growth and usage of social media, it has an extensive reach and impact on the society. OLI is helpful for hate speech detection, flame detection and cyber bullying, hence it is used to avoid abuse and hurts. In this paper, we present state-of-the-art machine learning approaches for OLI. We follow several approaches which include classifiers like Naive Bayes, Support Vector Machine(SVM) and deep learning approaches like Recurrent Neural Network(RNN) and Masked LM (MLM). The approaches are evaluated on the OffensEval@SemEval2020 dataset and our team ssn_nlp submitted runs for the third task of OffensEval shared task. The best run of ssn_nlp that uses BERT (Bidirectional Encoder Representations from Transformers) for the purpose of training the OLI model obtained F1 score as 0.61 and it is our official submission. The model performs with an accuracy of 0.80 and an evaluation loss of 1.08. The model has a precision rate of 0.72 and a recall rate of 0.58.

1 Introduction

In this day and age, offensive language is used predominantly in social media thereby giving rise to the need for more efficient and faster offensive language identification (OLI) algorithms. OLI is useful for several applications such as hate speech detection, flame detection, aggression detection and cyber bullying. Several workshops such as TA-COS1, TRAC2 (Kumar et al., 2018), Abusive Language Online3, ALW2 (Fišer et al., 2018) and GermEval (Wiegand et al., 2018) have been organized recently in this area of research. In this line, OffensEval@SemEval2020 (Zampieri et al., 2020) shared task focuses on Multilingual Offensive Language Identification in Social Media. It consists of three subtasks namely offensive language detection, categorization of offensive language and offensive language target identification. Sub Task A aims to detect text as offensive (OFF) or not offensive (NOT). Sub Task B aims to classify the offensive type as targeted text (TIN) or untargeted text (UNT). Sub Task C focuses on identifying the target of tweet as individual (IND), group (GRP) or others (OTH). Our team ssn_nlp participated in the third subtask of Multilingual Offensive Language Identification in Social Media.

2 Related Work

A lot of research has been reported in the field of Offensive Language Identification. Wiegand et al. (2018) presented an overview of GermEval shared task on the identification of offensive language that focused on the classification of German tweets from Twitter. A dataset of over 8,500 annotated tweets was provided for a coarse-grained binary classification task in which systems were trained to discriminate between offensive and non-offensive tweets. It also included a second task where the offensive class was subdivided into profanity, insult, and abuse.

There have been a few shared tasks and competitions for the classifications of classes in offensive languages like Kaggle's Toxic Comment Classification Challenge[1] and Automatic Misogyny Identification

[1] https://www.kaggle.com/c/jigsaw-toxic-comment-classification-challenge

Proceedings of the 14th International Workshop on Semantic Evaluation, pages 2155–2160
Barcelona, Spain (Online), December 12, 2020.

(AMI)[2] is jointly run by IberEval4 and EVALITA. The Kaggle competition is a shared task that required the classification of Wikipedia comments. They were to be classified based on the toxicity (i.e. toxic, severe toxic, obscene, insult, identity hate, and threat). The shared task of IberEval4[3] and EVALTIA[4] focused on the identification of misogynist tweets on Twitter.

ElSherief et al. (2018) performed linguistic and psycholinguistic analysis to detect that hate speech is either "directed" towards a target or "generalized" towards a group. Several datasets like Key phrase-based dataset, Hashtag-based dataset, Public datasets and General dataset were combined to form a final dataset which was evaluated by incorporating human judgment using Crowdflower. The dataset was lexically analyzed using SAGE and frame-semantics was applied to identify the hate speech and classify it into the datasets.

Zampieri et al. (2019) used the OLID dataset to classify the data into various categories by utilizing various models like SVM, BiLSTM, and CNN and has performed a comparison study among the respective macro- F1 scores and accuracy. Gambäck and Sikdar (2017) used deep learning using CNN models to detect hate speech as "racism", "sexism", "both" and "not hate-speech". They used character 4-grams, word vectors based on word2vec, randomly generated word vectors, and word vectors combined with character n-grams as features in their approach. The system first generates feature embeddings which are constructed using word embeddings and character n-grams. There are two types of word embeddings used i.e continuous-bags-of-words (CBOW) and skip-gram models.

Zhao et al. (2016) used an Enhanced Bag of Words (EBow) model on a real-world cyber-bullying dataset and an experimental dataset. The real-world dataset is a public bullying traces dataset, which consists of tweets sent on Twitter. They have then presented a study of the recall, precision and F1 scores between various models such as BoW, LSA, LSD and EBoW.

Chen et al. (2012) used a Lexical Syntactic Feature (LSF) architecture to detect offensive content and identify potential offensive users in social media. They distinguish the contribution of pejoratives/profanities and obscenities in determining offensive content and introduce hand-authoring syntactic rules in identifying name-calling harassments.

3 Data

In our approach, we have used the English dataset out of the multilingual dataset given by OffensEval@ SemEval2020 shared task. The dataset for task C is given in .tsv file format with columns namely, id, text, average_ind, average_grp, average_oth, std_ind, std_grp, std_oth. It consists of the labels Individual (IND), Group (GRP) and Other(OTH).

The column id contains distinct tweet id, text consists of the tweets, average_ind, average_grp, average_oth consists of the average score of the tweet for IND, GRP and OTH labels respectively, std_ind, std_grp, std_oth consists of the standard score of the tweet for IND, GRP and OTH labels respectively.

The IND label refers to offensive tweets that are targeted towards a single person who may be well known, a named individual or an unnamed participant in the conversation. These threats and insults are regarded as cyberbullying. The GRP label identifies tweets that are targeting a group of people based on their gender, ethnicity, political affiliation or other common characteristics. This is commonly called hate speech. The OTH refers to tweets that do not belong to the previous two categories i.e an organisation, an event, or a social issue.

All the instances provided in the dataset were considered for the Sub_task_C. We have preprocessed the data by removing the emojis, URLs and the text "@USER" from the tweets. The columns id, std_ind, std_grp, std_oth are dropped and the labels are computed by assigning the maximum value from the values average_ind, average_grp, average_grp for each tweet. Labels 0,1,2 correspond to average_ind, average_grp and average_oth respectively. After the assignment of labels, the 3 average columns are dropped and the label column is inserted.

[2] https://amievalita2018.wordpress.com
[3] https://sites.google.com/view/ibereval-2018
[4] http://www.evalita.it/2018

4 Methodology

We have employed traditional machine learning, deep learning and transformer approaches to identify the offensive language in social media.

4.1 Traditional Learning Approach

In the traditional machine learning approach, we use the classifiers Naive Bayes and Support Vector Machine(SVM) to build the models. Naive Bayes[5] is a supervised learning algorithm for classification of tasks. It find the class of observation (data point) given the values of features. It is based on Bayes theorem i.e. it assumes that features are independent of each other and there is no correlation between features. Support Vector Machine (SVM)[6] is responsible for finding a hyperplane in an N-dimensional space(N — the number of features) that distinctly classifies the data points. The objective is to find a plane that has the maximum margin, i.e the maximum distance between data points of all classes. The maximization of the margin distance provides some reinforcement so that future data points can be classified with more confidence.

The data consists of two columns i.e. text and label. For these two models, we first remove all the blank spaces from the data, change all the text to lowercase. We then tokenize every tweet, remove the stop words and the non-alpha text. We finally perform word lemmatization. Every tweet now consists of a list of independent strings. All this preprocessing is done using the nltk package present in python.

The dataset is split with 70 percent for training and 30 percent for testing. Word vectorization is done using TF-IDF scores and a maximum of 5,000 unique features are obtained. The models are then put to train. The Naive Bayes model has an accuracy of 0.79 and the SVM model has an accuracy of 0.80.

4.2 Recurrent Neural Network

In the deep learning approach, we use the Recurrent Neural Network(RNN)[7] using the Long Short Term Memory (LSTM) architecture. Recurrent Neural Networks can remember the previous output and use it as input for it's next set of recurring operations. The idea behind RNN is the usage of sequential information. It is called recurrent since it repeats the same operation on every element of the sequence.

LSTM is a modified version of RNN which makes it easier to remember past data in memory and resolves the gradient problem. LSTM models are trained by back-propagation.

For the LSTM Modelling, we first vectorize the tweets by turning them into a sequence of integers. We then truncate and pad the input sequence so that they are all in the same length for modeling.

We use a train-test split ratio of 90%-10% to divide the data into training and validation sets. We have used 1 layer of LSTM in this model. The output is passed to the softmax layer. We have trained the model with a batch size 64 and a dropout of 0.2 for over 5 epochs. The model has an accuracy of 0.91 and a loss of 0.22.

4.3 Transformer Approach

BERT(Bidirectional Encoder Representations from Transformers) introduced by Devlin et al. (2018) uses transformer, an attention mechanism that learns contextual relations between words (or sub-words) in a text. The transformer encoder used is bidirectional as opposed to directional models, which read the text input sequentially. This bidirectional behavior is responsible for allowing the model to learn the context of a word based on all of its surroundings (left and right of the word).

[5] https://towardsdatascience.com/naive-bayes-classifier-explained-50f9723571ed
[6] https://machinelearningmastery.com/support-vector-machines-for-machine-learning
[7] https://towardsdatascience.com/understanding-rnn-and-lstm-f7cdf6dfc14e

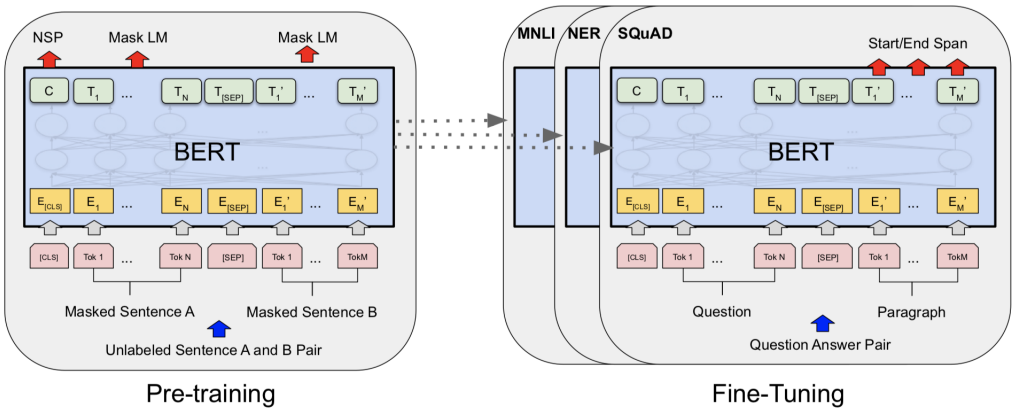

Figure 1: Pre-training and fine-tuning procedures for BERT.

We first process the data to have four columns; id, label, same character and the text to be classified. We then use a train-test split ratio of 80%-20% to divide the data into training and validation sets. The data is then converted to features that the BERT uses. The word sequences that are fed into BERT are first run through Masked LM (MLM). In MLM, 15 percent of the words per sequence is replaced with a [MASK] token. Based on the context provided by the non-masked words remaining in the sequence, the model tries to predict the masked words. We use 128 as the maximum sequence length after tokenization. We then load the pre-trained tokenizer for BERT, BERT-base-cased model and train our model over 1 epoch. The transformer model performs better than the machine learning and the deep learning approaches. The BERT model has a training loss of 0.11.

For evaluating the performance of the model, we use the 37,795 validation tweets. The model performs with a MCC score of 0.74 and an evaluation loss of 0.19. The model has a validation accuracy of 0.92, a precision score of 0.82 and a recall score of 0.76.

5 Results

In the OffensEval@SemEval2020 challenge for the Subtask C, our team ssn_nlp achieved a top submission rank of 15, with a score of 0.61. We followed a transformer approach that uses BERT. The model was tested against 850 tweets from the OffensEval@SemEval2020 Subtask C test data. The model performed on the test set with an accuracy of 0.80, F1-Score of 0.61, and an evaluation loss of 1.08. A precision score of 0.72 and a recall score of 0.58 was encountered.

	Precision	**Recall**	**F1 - Score**	**Support**
IND	0.83	0.94	0.88	580
GRP	0.71	0.63	0.67	190
OTH	0.62	0.19	0.29	80
Accuracy			0.80	850
Macro-average	0.72	0.58	0.61	850
Weighted average	0.78	0.80	0.78	850

Table 1: Results for Subtask C

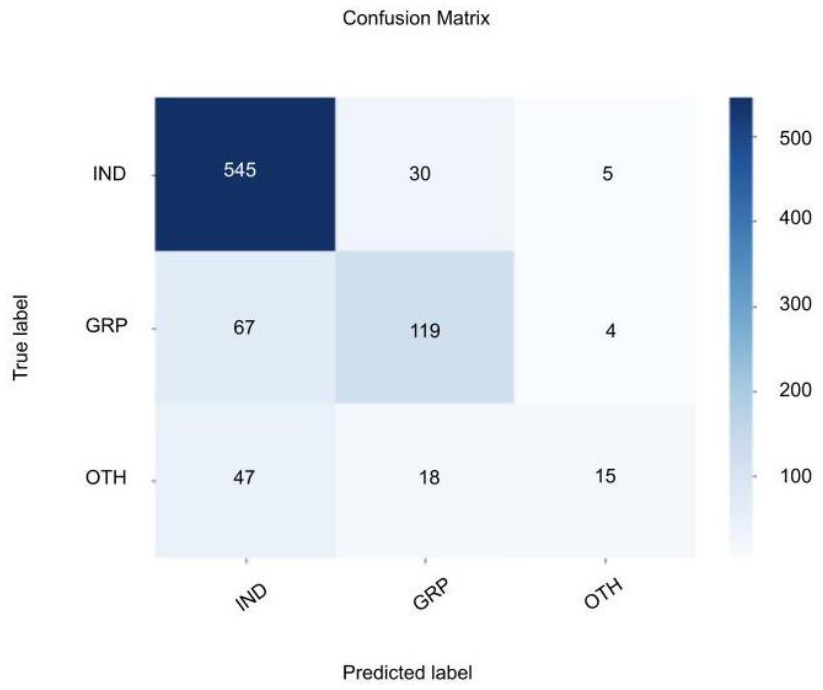

Figure 2: Confusion matrix for Subtask C

A macro-average will compute the metric independently for each and every class and then select the average (hence treating all classes equally). Therefore a macro-average score is used to obtain the system performance across various sets of data.

In calculating a weighted average, each number in the data set is multiplied by a predetermined weight before the final calculation is made. Therefore the precision, recall, and F1 scores are calculated for each label, and then it is weight is used as support to compute the average - which is the number of true instances for each label thereby obtaining the weighted average scores.

From Table 1, we can see that both recall and precision are high for the IND label (83% and 94%), but is significantly lower for GRP label (71% and 63%) and OTH label(62% and 19%). This is explained by the class imbalance in the test data and class IND having more number of instances than the other two classes.

The confusion matrix of BERT approach for Subtask C is shown in Fig. 1. From this we can see that the 679 instances were predicted correctly out of the 850 total instances, i.e True positives(TP). The total number of falsely classified instances is 171.

6 Source Code and Computational Resources

We have uploaded the source code for pre-processing the dataset and fine-tuning the BERT architecture to our GitHub repository[8]. The resources we used for our final submission were a 2.3 GHz Intel core i5 CPU and Intel Iris Plus Graphics 640 1536 MB GPU.

7 Conclusion

We have implemented both traditional and deep learning approaches for identifying if the offensive language is targeted at a group, individual or others.

The approach we submitted was a deep learning transformer model called Bert. The approach is evaluated on OffensEval@SemEval2020 dataset. The BERT model that is trained in the pytorch environment, gave a validation accuracy of 92% as compared to the validation accuracy of 91% given by the RNN model which also uses deep learning approaches.

The traditional learning approaches like Naive Bayes and SVM gave a validation accuracy of 79% and 80% respectively. These results suggest that the deep transfer learning approaches are capable of much

[8] https://github.com/prnan4/BERT_text_classification/tree/master

deeper and more complex representations, such that they can utilize previously learned features for newer documents, even when the type of document differs significantly in key properties such as length and vocabulary, therefore enabling BERT as our best choice.

As a deep learning approach , BERT is more effective on larger datasets like the one provided by OffensEval@SemEval2020 (Zampieri et al., 2020). BERT uses word piece embeddings which makes it more robust to new vocabularies when compared to classical approaches like Naive Bayes and SVM. There is not much additional preprocessing required for BERT making it easier to use when compared to traditional approaches where multiple steps such as removing stop words, blank spaces and non alpha text, converting text to lower case and word lemmatization is required. BERT applies bidirectional training which has a deeper sense of language context and flow when compared to single-direction models thereby making it a desirable deep learning approach. In future, we intend to apply techniques that work well with the given data and record higher scores of accuracy, precision and recall.

References

Y. Chen, Y. Zhou, S. Zhu, and H. Xu. 2012. Detecting offensive language in social media to protect adolescent online safety. In *2012 International Conference on Privacy, Security, Risk and Trust and 2012 International Confernece on Social Computing*, pages 71–80.

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*.

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. In *Twelfth International AAAI Conference on Web and Social Media*.

Darja Fišer, Ruihong Huang, Vinodkumar Prabhakaran, Rob Voigt, Zeerak Waseem, and Jacqueline Wernimont. 2018. Proceedings of the 2nd workshop on abusive language online (ALW2). Brussels, Belgium, October. Association for Computational Linguistics.

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.

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 Cyberbullying (TRAC-2018)*, pages 1–11, Santa Fe, New Mexico, USA, August. Association for Computational Linguistics.

Michael Wiegand, Melanie Siegel, and Josef Ruppenhofer. 2018. Overview of the germeval 2018 shared task on the identification of offensive language. 09.

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. *arXiv preprint arXiv:1902.09666*.

Marcos Zampieri, Preslav Nakov, Sara Rosenthal, Pepa Atanasova, Georgi Karadzhov, Hamdy Mubarak, Leon Derczynski, Zeses Pitenis, and Çağrı Çöltekin. 2020. SemEval-2020 Task 12: Multilingual Offensive Language Identification in Social Media (OffensEval 2020). In *Proceedings of SemEval*.

Rui Zhao, Anna Zhou, and Kezhi Mao. 2016. Automatic detection of cyberbullying on social networks based on bullying features. In *Proceedings of the 17th International Conference on Distributed Computing and Networking*, ICDCN '16, New York, NY, USA. Association for Computing Machinery.