ANALYSIS OF ONLINE COMMENTS USING MACHINE LEARNING ALGORITHMS

Sneha Bushetty¹, Prasanna Thummalacheruvu², Vineetha Ramavath³, C. Jagadeswari⁴, Meghana Devarapalli⁵

¹Department of Computer Science & Engineering, BVRIT Hyderabad
College of Engineering for Women, Rajiv Gandhi Nagar, Hyderabad

¹sneha1779@gmail.com, ²16wh1a05b7@bvrithyderabad.edu.in,
³16wh1a05a7@bvrithyderabad.edu.in, ⁴cjagadeswari3110@gmail.com,
⁵16wh1a0581@bvrithyderabad.edu.in

https://doi.org/10.26782/jmcms.2020.09.00014

Abstract

Online social forums are a great place to express one’s opinions on others’ work. But due to the threat of harassment and abuse online, many people stop expressing themselves and give up on seeking different opinions. This leads to the complete shutdown of the user comments section in many communities. Hence, there is a need to identify an efficient way to detect the level of toxicity in the comments posted online, which will be helpful to the content moderators who monitor the data obtained from the comments section on online forums. In this paper, we train various machine learning and deep learning models like NB-SVM, LSTM, BERT on the toxic comments dataset and analyze which approach is efficient for the task of classification of toxic comments.

Keywords: Classification, NB-SVM, BiLSTMs, BERT, Comments, Content Moderation

I. Introduction

The Internet is an open discussing space for everyone to freely express their opinions. The problem originates from the comments that are posted online. Sometimes the comments may be abusive and insulting or hate-based which may discourage people from sharing their opinions. So, it is important to ensure that positive, healthy conversations take place. Hence, there is a need to build technology to protect voices in a conversation. Organizations are still in search of the best solution that determines whether a comment is insulting, hate-based, or not. The aim is to generate a model that can categorize comments into various classes efficiently. Commented by a user (a group of sentences or paragraphs), the model makes predictions to determine the class (es) of a comment: toxic, obscene, insult, severe_toxic, threat, identity_hate. The models developed are trained on a dataset consisting of Wikipedia comments rated with the toxic levels by human raters. The
models return the probabilities of a comment belonging to a certain class label, i.e., a toxic level. In this paper, various machine learning and deep learning algorithms are used to classify the toxicity of the comments and are compared to identify the efficient approach.

II. Literature Review

In recent years, many kinds of research have been conducted involving the toxicity of language. This research is done in the context of social media. In a broad sense, toxic comment classification falls under sentiment analysis. Early work in this field began at Lehigh University in 2009 where a group of researchers combined TF-IDF with sentiment features. Over time, more and more companies have taken up this task with the most prominent among them being Google who established the company Jigsaw, a company dedicated to stopping online harassment. Toxic comment detection can be tackled by using machine learning architectures like XGBoost, SVM (Support Vector Machine), Logistic regression, and deep learning architectures like ANNs (Artificial Neural Networks), RNN (Recurrent Neural Networks), etc. Mukul Anand and R. Eswari used LSTM (long short term memory cell) and CNN (Convolution neural network) for the purpose in their paper [III]. Julian Risch and Ralf Krestel used deep learning techniques and explained toxicity, their levels, architectures used in their paper [IV]. They considered the task as binary classification rather than multi-label classification i.e., the comments are classified into toxic or non-toxic classes. In this paper, we have used BERT, BiLSTMs to classify the online comments into various toxicity levels which give deeper and specific information about the toxicity of the text rather than just a binary classification of the comment being toxic or non-toxic.

III. System Architecture

In this paper, we implemented 3 different algorithms namely NB-SVM (Naive Bayes Support Vector Machine), LSTM (Long Short Term Memory) and Google’s BERT (Bidirectional Encoder Representation From Transformers). NBSVM can be defined as SVM that uses Naive Bayes log-count ratios in feature values and is proved to be the best baseline method. LSTMs are special variants of RNN. They learn long-term dependencies. The sequences of data can be processed by LSTM as it has feedback connections. Google’s BERT uses a multi-layer bidirectional Transformer encoder. BERT performs self-attention in both directions. For evaluation, four different metrics are chosen: accuracy, hamming-loss, log-loss, and f1_score.
IV. Implementation

IV.i. Dataset and Data Preprocessing

The dataset used to train models is the Jigsaw/Conversation AI dataset provided for the Kaggle Toxic Comment Classification Challenge. The dataset contains Wikipedia comments and each comment has a label/toxic level associated with it. There are nearly 160,000 comments in the dataset. The comments are categorized into one or more of the six classes namely toxic, obscene, insult, severe_toxic, threat, identity_hate. Comments are largely in English except for a few of them being in Chinese, German, Arabic languages. The comments are related to various topics. The text in the comments contains many non-alphabetical symbols which can be removed before training to avoid unnecessary training of models on such symbols. Some algorithms from the NLP (Natural Language Processing) library NLTK (Natural Language Toolkit) are used to remove punctuation marks and stop words. Words in the comment text are also stemmed and lemmatized before passing it for training.

IV.ii. Approaches

NB-SVM (Naive Bayes - Support Vector Machine) For our first model, we considered a combination of SVM and Naive Bayes. In this algorithm, a linear classifier SVM uses log-count ratios as its features. The Log-count ratio is calculated separately for each class label and the model is trained on the feature values multiplied by the log-count ratio for that class label. Thus, we obtain 6 different trained models for 6 different class labels. To obtain the feature values, we have used the TF-IDF vectorizer to convert the comment text into vector sparse matrices. Our main model variants are linear classifiers [VI], where the nth test case prediction is

\[ y^{(n)} = \text{sign}(w^T x^{(n)} + b) \]  

(1)

assume \( fc^{(0)} \) belongs to \( R^{|V|} \) is the feature count vector for \( j^{th} \) training sample with class label \( y^{(j)} \). The range of \( y^{(j)} \) is \( \{-1, 1\} \). \( V \) is the feature set, and \( fc^{(0)} \) represents the
frequency of feature $V_i$ in the $j^{th}$ training sample. The count vectors for the smoothing parameter $\beta$ are:

$$p = \beta + P_{j=1} f_c(j)$$  \hspace{1cm} (2)

$$p = \beta + P_{j=1} f_c(j)$$  \hspace{1cm} (3)

$$\log - countRatio = \log((p/\|p\|)/(q/\|q\|))$$  \hspace{1cm} (4)

$$x^{(a)} = (\log - countRatio) \cdot f_c^{(a)}$$  \hspace{1cm} (5)

where the element-wise product is:

IV.iii. LSTM (Long Short Term Memory)

Our second approach is using LSTM, a special kind of RNN. In sequential data like sentences, dependencies may not just be in one direction but also in both directions. Hence, we used BiLSTMs (Bi-directional LSTMs) that can obtain dependencies by traversing in forward and backward directions simultaneously. We used Keras tokenizer to convert the comment text into an array of vectors.

We used Glove (Global Vectors For Word Representations) pre-trained word embedding by the Stanford NLP group to create an embedding matrix. A matrix is a dictionary that consists of the words from the set of unique words from comment text which are present in the pre-trained embeddings and the corresponding vectors assigned. There are 3 fully connected layers in the model, in which two are with relu activation functions and another one is the last layer with sigmoid activation function which is used to get the output predictions in the form of probabilities instead of binary values. We used 3 dropout layers with a frequency rate of 0.1. Adam optimizer, binary cross-entropy loss are used for training the BiLSTM model.

IV.iv. BERT (Bidirectional Encoder Representation From Transformers)

BERT performs efficiently for eleven NLP tasks. Tasks like Question Answering and Sentence classification can be approached through only fine-tuning the last layer. Next Sentence Prediction and masked language model (MLM) are used for pre-training BERT. This will help the encoder in learning the context of a word in the sentence by parsing both left and right surroundings of it and provides significant support for various NLP tasks. The results from the paper [3] depict that bidirectional language models perform efficiently when compared to single-direction language models because they understand language context well.

In our approach, we used the BERT-Base model and modified the output layer to work for our toxic comment classification. The pre-trained model is cased and has 12 layers with 768-hidden, 12-heads, and 110M total parameters. We used Bert tokenizer to tokenize the words in the comment texts and converted the text into the form Bert accepts by adding “CLS” and “SEP” tokens.

Sneha Bushetty et al
V. Results

The evaluation results are shown in Table 1.

| Description | Accuracy(%) | F1_score | Hamming-loss(%) | Log-loss |
|-------------|-------------|----------|-----------------|----------|
| NBSVM       | 91.88%      | 0.6981   | 1.8599%         | 1.6319   |
| LSTM        | 91.01%      | 0.6590   | 2.1407%         | 1.3727   |
| BERT        | 92.73%      | 0.7896   | 1.5342%         | 1.1183   |

Comparing the models based on hamming-loss, the BERT classifier is observed as the efficient one with the least hamming-loss 1.5342%, BiLSTM as the less efficient one with a Hamming-loss of 2.1407%. Comparing the models based on log-loss resulted in, BERT as efficient with the least loss of 1.118300 and NBSVM as the least efficient one with the highest loss of 1.6319. Based on accuracy scores, Bert scored the highest accuracy of 92.7309. Comparing the f1 scores of the models, BERT achieved the highest f1 score of 0.7896.

![Fig. 2: F1_score results](image-url)
VI. Conclusion

To give an outline of our work, we implemented three models, namely the Naive Bayes-SVM model, LSTM, and BERT. We used information from data visualization to preprocess data. After comparing all the results, comparing models...
based on all the 4 evaluation metrics, i.e; F1_score, Hamming-loss, Accuracy, and log-loss, the BERT classifier is observed as the efficient classifier with good accuracy, f1_score, and less hamming-loss and log-loss values.

VII. Future Scope

In the future, this work can be extended by dealing with the code languages like leetspeak words (Ex: $#!+ which is leetspeak word for Shit) or comments originating from mobile devices, use of acronyms, or intentionally obfuscating words to avoid filters by inserting spurious characters, using phonemes, dropping characters, etc. Also instead of removing punctuation marks, assigning them with the sentiment values improves the efficiency of the classification.

Conflict of Interest:
Authors declared: No conflict of interest regarding this article.

References
I. Amir Moradibaad, Ramin Jalilian Mashhoud Use Dimensionality Reduction and SVM Methods to Increase the Penetration Rate of Computer Network, J. Mech. Cont. & Math. Sci., Vol.-14, No.-4, July-August (2019), pp 8-26.
II. Arif Ullah, Umeriqbal, Ijaz Ali Shoukat, Abdul Rauf, O Y Usman, Sheeraz Ahmed, Zeeshan Najam, An Energy-Efficient Task Scheduling using BAT Algorithm for Cloud Computing, Vol.-14, No.-4, J. Mech. Cont.& Math. Sci., July-August (2019), pp 613-627.
III. Classification of Abusive Comments in Social Media using Deep Learning, Published in 2019 Proceedings of the Third International Conference on Computing Methodologies and Communication (ICCMC 2019) IEEE Xplore
IV. Is preprocessing of text worth your time for toxic comment classification, Int'l Conf. Artificial Intelligence | ICAI'18 |
V. Julian Risch and Ralf Krestel Hasso: Toxic Comment Detection in Online Discussions, Plattner Institute, University of Potsdam.
VI. Jacob Devlin, Ming-Wei Chang, Kenton Lee, Kristina Toutanova, Google AI Language: BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding
VII. Sidawang and Christofer D. Manning: Baselines and Bigrams: Simple, Good Sentiment and Topic Classification, Department of Computer Science Stanford University