विश्वजीवनामृतं ज्ञानम्

ATAL BIHARI VAJPAYEE INDIAN INSTITUTE OF INFORMATION TECHNOLOGY AND MANAGEMENT GWALIOR

INFORMATION TECHNOLOGY

Minor Project

TOXIC COMMENT DETECTION

2021-IMT-041 - Harsh
Sharma

Under the supervision of
Dr.Sunil-Kumar

Contents

1 INTRODUCTION

Over the past decade, social media platforms such as Twitter, Instagram, and LinkedIn have evolved from mere pastimes into primary sources of information consumption, especially for news and content. They're now preferred over traditional web searches due to their vast user base, facilitating instant sharing and resharing of information as events unfold. Comments on these platforms serve to enhance user interaction and expression. However, they can also be exploited, particularly through the use of bots, to disseminate fake news, propaganda, or hurtful content. This includes messages that incite panic, promote racial discrimination, issue threats, or engage in other forms of abuse, collectively termed as toxic comments or information. This paper examines various types of comments and seeks to analyze their toxic nature, employing accessible language to convey the complexities involved.

In response to the aforementioned challenge, we've devised a Toxic Comment Classification System employing Deep Learning techniques. This system is engineered to identify and categorize toxic comments effectively.It aids individuals in abstaining from utilizing negative or vulgar language during their interactions with others.

The dataset utilized in this endeavor originates from Kaggle. The chosen model for this system is LSTM, which stands for Long Short-Term Memory. LSTM is kind of recurrent neural networks (RNNs) and outperforms traditional RNNs in memory retention. Similar to other neural networks, LSTM can comprise multiple hidden layers. However, it excels in preserving pertinent information while discarding irrelevant data as it progresses through each layer in every cell.

In the realm of text representation for Machine Learning applications, word embeddings play a pivotal role. They involve mapping words to vectors of real numbers, providing a numerical representation of textual data. The inception of the first word embedding model, Word2Vec, utilizing Neural Networks dates back to 2013, pioneered by researchers at Google. Since then, word embeddings have become universal in almost every Natural Language Processing (NLP) model utilized in practical applications. This widespread adoption is primarily attributed to their effectiveness.

By translating words into embeddings, it becomes feasible to capture the semantic significance of a word in a numeric format, enabling mathematical operations on it. In 2018, Google researchers introduced Bidirectional Encoder Representations from Transformers (BERT), a profoundly bidirectional, unsupervised language representation method capable of generating word embeddings that encapsulate the semantics of words within their contextual usage.

In contrast, context-free models like Word2Vec is used to generate a single embedding representation for each word in the vocabulary, Disregarding of the context of the word. Within this configuration, the paper focuses on designing and evaluating various deep learning models fed by word embeddings to discern levels of toxicity in the textual comments, employing medium-level language and rephrasing where necessary.(1)

2 MOTIVATION

Toxic comment detection models using Natural Language Processing (NLP) in deep learning are created to make online communities like Instagram, facebook etc. safer and more user friendly. As the amount of content shared online keeps increasing, trolling, abusiveness etc. keep growing too . These models automatically detect and filter out toxic comments like hate speech and harassment. They use advanced NLP techniques and deep learning to understand and flag harmful content accordingly, which helps to maintain community rules and safeguard users. This not only saves time and resources but also ensures that online platforms comply with laws against sharing harmful or illegal content on internet. Ultimately, the aim of this model is to ensure safety at online world for users all around.(2)

3 CONTRIBUTION

1. Ankit Kumar Singh(2021IMT-012)-Implemented tokenization and padding.
2. Harsh Sharma(2021IMT-041)- Worked on Traditional Model , Data Preprocessing and report preparation.
3. Shruti Murthy(2021IMT-094)-Designed the LSTM model architecture and report preparation, Handled out-of-vocabulary words.
4. Shail Pujan(2021IMT-088)-Trained and tested the LSTM model on the dataset, vocabulary words handling and report preparation.

4 LITERATURE SURVEY

The Kaggle Toxic Comment Classification Challenge attracted various contributions, showcasing diverse approaches to toxic comment classification using deep learning methods. Among the notable works are:

B. ”LSTM-based deep learning models for non-toxic and toxic comment classification in community question-answering platforms” (2020) by A. S. Shahin, M. K. Hossain, and M. K. A. Mollah. This paper focused on employing Long Short-Term Memory (LSTM) networks to identify toxic comments in community question-answering platforms, shedding light on the application of recurrent neural networks for this task.

C. ”Convolutional Neural Networks for Toxic Comment Classification” (2017) by Yoon Kim. Kim’s study delved into the utilization of convolutional neural networks (CNNs) for toxic comment classification, demonstrating the effectiveness of CNNs in capturing local features within text data.

D. ”Hate Speech Detection with Comment Embeddings” (2019) by Pranav Goel and Arjun Mukherjee. Although not specifically targeting toxic comments, this research explored the use of comment embeddings for hate speech detection, presenting concepts and methods relevant to the broader context of identifying offensive and harmful content.

A.N.M. JuBaerl’s ”Bangla Toxic Comment Classification” proposes a method that combines multiple algorithms to achieve the classification goal. They utilized techniques such as the Binary Relevance Method with Multinomial NB Classifier and supervised machine learning algorithms like SVMs. Comments were categorized into subdivisions such as obscene, severe, threat, clean, and identity hate. These features were converted into arrays, tokenized using CountVectorizer, and stop/violated words were removed. Among the tested classifiers, BP-MLL emerged as the best with a respectable accuracy of 60%.

Muhammad Husnain introduced ”A Novel Pre-processing Technique for Toxic Comment Classification” aiming to address toxic comments. The research emphasized the need for policies to govern which comments are considered harmful. It employed a mix of approaches, including training separate classifiers for each dataset and tackling multi-labeled classification problems. Various machine learning techniques like Na¨ıve Bayes, Decision tree classification, and Logistic Regression were utilized. The model exhibited very high accuracy, ranging from 94-99%, even after conducting 10-fold cross-validation.

Nadira Boudjani proposed ”Toxic Comment Classification for French Online Comments,” which employed a supervised method to identify toxic comments in the French language. Due to a lack of classifiers in French, English classifiers were utilized. The study employed selective features such as insulting words and expressions documented in the dictionary. Multiple machine learning algorithms like SVMs and decision Trees were utilized, resulting in a relatively lower accuracy compared to English toxic detection. However, an F1 score of 78

Kazi Saeed Alam presented ”Cyberbullying Detection: An Ensemble-Based Machine Learning Approach,” highlighting the urgency to address online hatred, bullying, racism, and harassment. The study proposed single and double ensemble-based voting models to classify information into offensive and non-offensive categories. It employed four ML classifiers along with three ensemble models, outperforming the proposed DLE and SLE voting classifiers with a commendable performance of 96

5 METHODOLOGY

5.1 Installing dependencies

Installing dependencies and importing data is an essential step in environment setup and dataset loading. To install the necessary Python packages, use the!pip install command.

In order to manage file operations and data manipulation, libraries like os, pandas, and numpy are imported. In order to train the model, the dataset (train.csv) is put into a Pandas DataFrame.

5.2 Pre Processing

To make sure the data is in the right format, preprocessing steps are conducted before feeding it into the model. TensorFlow Keras' TextVectorization layer is used to transform unprocessed textual data into numerical representations. The prepared data for the input (X) and output (y) is a DataFrame with labels denoting the toxicity categories in y and the comment text in X. TextVectorization layer settings include output_sequence_length (length of output sequences after padding or truncation) and max tokens (maximum vocabulary size). .
In order to construct the vocabulary based on the remarks, the layer is adjusted to the data. The TextVectorization layer vectorizes the comment text, turning each remark into a string of integers that represent words.
Finally, a TensorFlow dataset is created from the vectorized text and labels, applying optimizations such as caching, shuffling, batching, and prefetching to improve training efficiency. (3)

5.3 Deep Learning Models

This step involves building the deep learning model architecture. The model is constructed using TensorFlow Keras' Sequential API, allowing for a linear stack of layers.

5.3.1 Embedding Layer

This layer converts integer-encoded words into dense vectors of fixed size. It helps represent words in a continuous vector space where similar words have similar representations.

5.3.2 LSTM Model

LSTM layers are employed for a binary classification task on input text. Unlike traditional Recurrent Neural Networks (RNN), LSTMs are an extended version tailored for sequences. They utilize memory blocks to retain the computation state, enabling the learning of temporal relationships within data. This mechanism binds the current data being processed with previously processed data chunks, facilitating the inference of semantic patterns that encapsulate the data's historical context. This addresses a limitation of standard RNNs, which primarily rely on the most recent input data, thus enhancing the model's ability to discern relevant information across the entirety of the input sequence.

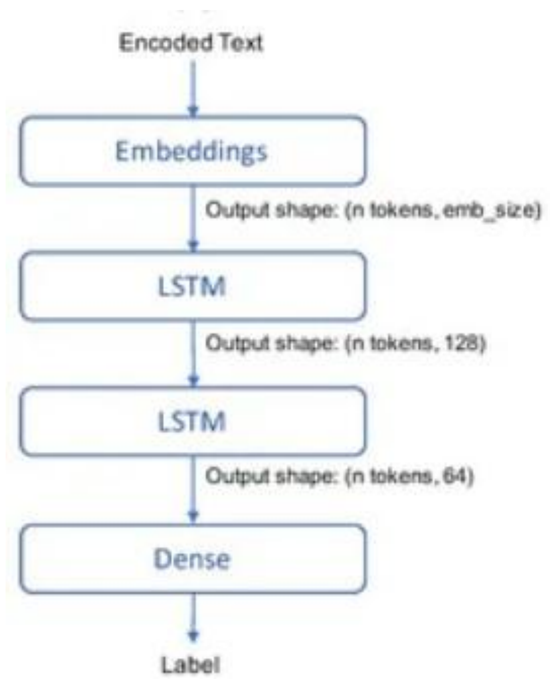

Encoded Text
Embeddings
Output shape: (n tokens, emb_size)
LSTM
Output shape: (n tokens, 128)
LSTM
Output shape: (n tokens, 64)
Dense
Label

Figure 1: Long-Short Term Memory (LSTM)

5.3.3 Bidirectional LSTM

LSTM is a type of recurrent neural network (RNN) which has potential of understanding long-term dependencies. Bidirectional LSTM is used to process the input sequence in both forward and backward directions, to capture contextual information efficiently.

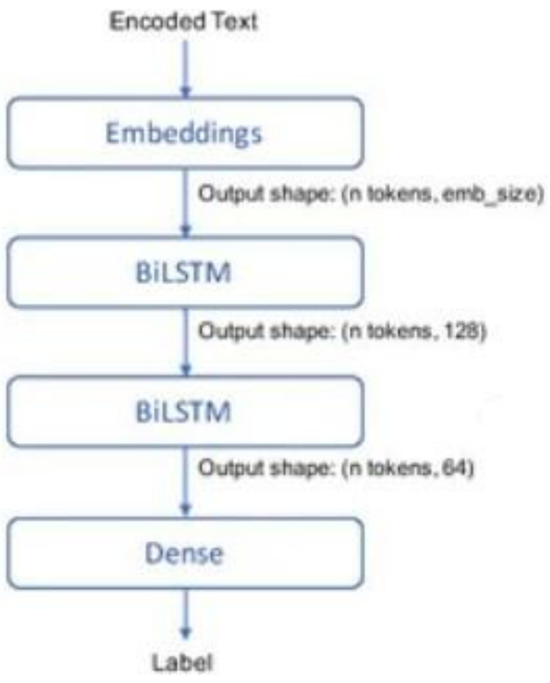

Encoded Text
Embeddings
Output shape: (n tokens, emb_size)
BiLSTM
Output shape: (n tokens, 128)
BiLSTM
Output shape: (n tokens, 64)
Dense
Label

Figure 2: Bidirectional LSTM

5.3.4 Dense Model

Fully connected layers responsible for feature extraction. ReLU (Rectified Linear Unit) activation functions are used for introducing non-linearity.(4)

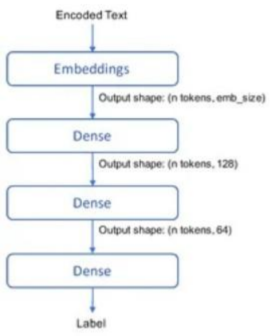

Encoded Text
Embeddings
Output shape: (n tokens, emb_size)
Dense
Output shape: (n tokens, 128)
Dense
Output shape: (n tokens, 64)
Dense
Label

Figure 3: Dense Layer

5.3.5 Final Dense Model

This layer produces the output probabilities for each toxicity category using a sigmoid activation function, enabling multi-label classification.

The model is compiled with appropriate loss function (BinaryCrossentropy) and optimizer (Adam), and its summary is printed to visualize the architecture. Training is performed for one epoch on the training dataset while validating on the validation dataset. (5)

5.4 Make Predictions

When our model is trained, it can predict the new data provided.
Input text is prepared for prediction by vectorizing it using the same TextVectorization layer. The model predicts the probability of each toxicity category for the input text.
Thresholding is applied to convert probabilities into binary predictions (toxic or non-toxic).

5.5 Test and Gradio

This step involves deploying the model for real-world use using Gradio, a library for building UIs for machine learning models.

The model is saved and then loaded back for deployment. A function is defined to score comments using the loaded model, providing toxicity scores for each category.

Gradio interface is created, allowing users to input comments and receive toxicity scores interactively.

The interface is launched for testing and sharing, enabling easy access to the model's functionality.

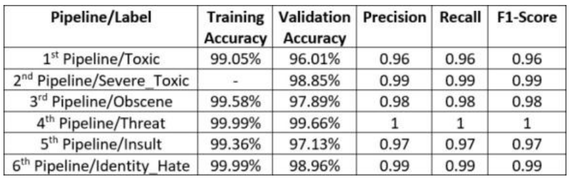
6 RESULT AND DISCUSSIONS

6.1 Individual Results

Our Accuracy obtained on training set is described by training accuracy.’ Our Accuracy obtained on Test set is described by Validation accuracy.
We define Precision as the ratio of correctly predicted positive observations to the total predicted positive observations.
The formula for Precision is shown in equation.
Precision = TP/TP +FP
We have defined Sensitivity or Recall as the proportion of correctly identified positives.
The formula for Recall is given in equation.
Recall = TP/TP+FN
The Harmonic Mean of Precision and Recall is F1-Score.

After training, the pipelines are being used for testing or validating the remaining 20000 samples. All the pipelines give their predictions independently. But if the label is severe toxic, it is evident that unless a comment is detected to be toxic, it does not have a chance of being severe toxic. So based on the predictions made by the 1st Pipeline for the toxic label, those test instances that are not said to be toxic, are labelled 0, for the label severe toxic i.e., not detected as severe toxic. Hence, only for the label severe toxic, a 2nd check is made with reference to prediction made by the 1st Pipeline for the label toxic i.e., only those instances that are detected positive (Toxic) by 1st Pipeline are fed to the 2nd Pipeline for predictions. So, no Training Accuracy has been shown for 2nd Pipeline for the label severe toxic.

The Training Accuracy, Validation Accuracy, Precision, Recall and F1-Score for all the pipelines/labels are tabulated in Table 1.



Pipeline/Label	Training Accuracy	Validation Accuracy	Precision	Recall	F1-Score
1st Pipeline/Toxic	99.05%	96.01%	0.96	0.96	0.96
2nd Pipeline/Severe_Toxic	-	98.85%	0.99	0.99	0.99
3rd Pipeline/Obscene	99.58%	97.89%	0.98	0.98	0.98
4th Pipeline/Threat	99.99%	99.66%	1	1	1
5th Pipeline/Insult	99.36%	97.13%	0.97	0.97	0.97
6th Pipeline/Identity_Hate	99.99%	98.96%	0.99	0.99	0.99

Figure 4: Bidirectional LSTM

Training Accuracy, Precision,Validation Accuracy, Recall and F1-Score for all the pipelines/labels.
Area Under Receiver Operator Characteristic Curve (AUROC).
We plot ROC Curve as True Positive Rate vs False Positive Rate. ROC Curve should have Area greater than 0.5 that is termed as an Acceptable Test. Here,

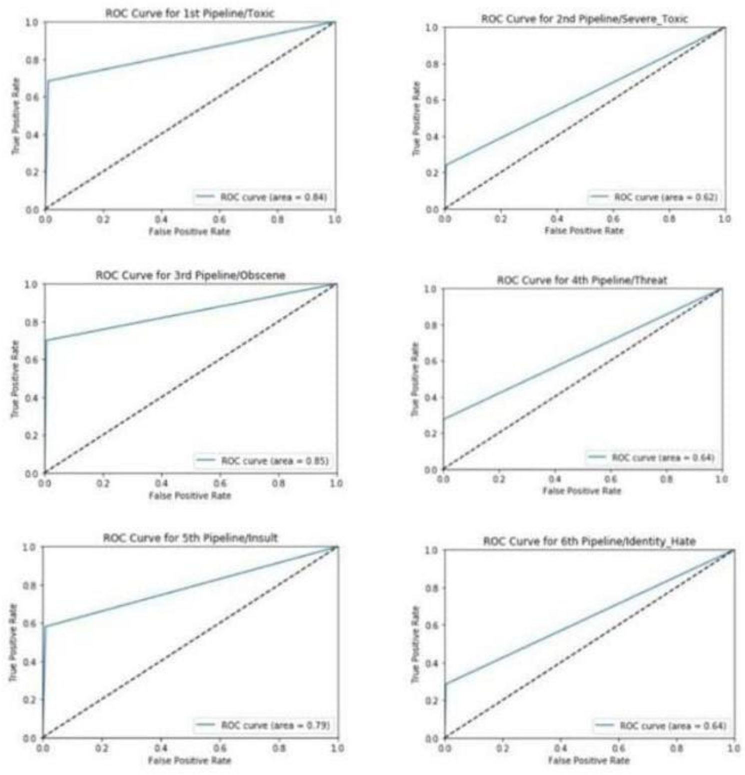
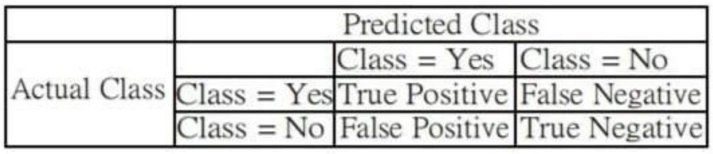


Fig 5. ROC Curves and Area Under ROC Curves for all the pipelines/labels

the ROC Curves are plotted by making the predictions specified by the pipelines which is on the Validation Set, rather than considering the distances of the predicted probabilities from the decision boundary.

6.2 Confusion Matrix Structure



		Predicted Class	
		Class = Yes	Class = No
Actual Class	Class = Yes	True Positive	False Negative
	Class = No	False Positive	True Negative

7 CONCLUSION

Our research involved a thorough examination of various deep learning models fueled by diverse word embedding representations to spot toxicity within textual comments. Our findings unequivocally demonstrate the effectiveness of machine and deep learning techniques, utilizing both syntactic and semantic cues from text, in identifying toxic content. Specifically, our study highlights the LSTM-based model as the most effective choice for detecting toxicity among the models we investigated.

As online communication increasingly influences our interactions, it's imperative to employ tools capable of curtailing the spread of toxicity and nurturing a more positive digital environment. By grasping and implementing these insights, we can endeavor to establish safer and more inclusive online platforms for all users.

8 FUTURE SCOPE

As we understand, we have put-up to perform a deep assessment of deep learning models by using and combining different kind of layers, which detect patterns in a better way and on the real scenarios where classes are unbalanced. Overall, we would like to investigate other contextualized word embedding representation, such as ELMO for the toxic comment detection. It is an breakdown of the proposed approach in which arrangements, parameter settings and heuristic approaches are added which could tackle the same problems but in presence of highly unbalanced datasets is definitely a research direction we would like to investigate as well.
Sooner, we would like to research the effect of using different embeddings in the same word, since it might be the cause of failure of BERT embeddings in our experiments. We also think that an ensemble strategy of the proposed approaches which should result in entire better results.

References

[1] "Conversation ai team." https://conversationai.github.io/.

[2] M. Kohli, E. Kuehler, and J. Palowitch, "Paying attention to toxic comments online." https://web.stanford.edu/class/cs224n/reports/6856482.pdf.

[3] "Jigsaw toxic comment classification challenge data." https://www.kaggle.com/c/jigsaw-toxic-comment-classification-challenge/data.

[4] S. V. Georgakopoulos, S. K. Tasoulis, A. G. Vrahatis, and V. P. Plagianakos, "Convolutional neural networks for toxic comment classification," in *10th Hellenic Conference on Artificial Intelligence*, 2018.

[5] T. Chu, K. Jue, and M. Wang, "Comment abuse classification with deep learning." https://web.stanford.edu/class/cs224n/reports/2762092.pdf.