Topic: AUTOMATIC DETECTION OF HATE SPEECH

AND OFFENSIVE LANGUAGE IN

SOCIAL MEDIA

USING DEEP LEARNING

Table of Contents

Abstract

A significant challenge in automatically detecting hate speech on social media lies in effectively differentiating hate speech from other instances of offensive language. Lexical detection methods often yield low precision because they classify all messages containing specific terms as hate speech. Previous attempts utilizing supervised learning have struggled to distinguish between these two categories. As a result, developing an automated system capable of detecting and flagging inciteful posts before they escalate is of utmost importance. Artificial intelligence is rapidly advancing, with the latest innovation being chat-GPT3, a conversational bot built upon the deep learning architecture of generative pretrained transformers.

Leveraging such cutting-edge algorithms presents an opportunity to construct a state-of-the-art classification system. This project is centered around the implementation of a deep learning system for text classification, specifically focused on detecting hate speech and offensive language on social media. The project utilized context-independent word embedding techniques, including word2vec, GloVe, and FastText, in combination with BiLSTM—a variant of Recurrent Neural Network (RNN) models. Furthermore, the context-dependent BERT algorithm was also integrated into the system. The performance of these systems was rigorously evaluated and compared.

Remarkably, the GloVe + BiLSTM and BERT algorithms stood out with exceptional scores of 95% accuracy, having been trained on annotated tweets sourced from Twitter. The project's overarching goal is to stimulate discourse and collaborative efforts within the research community, specifically in the realm of natural language processing.

INTRODUCTION

The fundamental human right of freedom of expression was formally recognized by the United Nations General Assembly in 1948 in Paris (UNGA, 1948). However, it's important to note that this right doesn't extend to allowing discrimination against others. Laws such as the New York anti-discrimination law (Berger, 1952) specifically prohibit discrimination based on factors like race, colour, religion, or nationality.

In recent times, the advent of social media has significantly expanded individuals' ability to voice their opinions. Nevertheless, some individuals have misused this newfound freedom, exploiting it to propagate hatred and engage in abusive behaviour. The broad reach and instantaneous nature of social media platforms carry inherent risks, particularly when exploited for negative purposes like disseminating hate speech. A notable example of this was observed during the US 2021 elections when a Twitter post by Donald Trump played a role in sparking the Capitol Hill riot (BBC, 2021).

Addressing the issue of offensive language in text has captured the attention of researchers, prompting a focus on automated detection mechanisms. The progress in artificial intelligence, particularly in the domain of natural language processing, has facilitated the evolution of algorithms designed to automatically identify offensive or toxic content. This task, akin to sentiment analysis, involves classifying text based on its emotional tone.

This project adopts a novel approach by leveraging word embedding techniques, a departure from conventional methodologies that involve converting text into numerical representations for classifier input. Instead, the text corpus is transformed into high-dimensional vectors using contextual and semantic cues associated with individual words. This approach aims to construct models proficient in categorizing offensive textual content.

The desired outcome of this endeavour is to create a model distinguished by its precision and efficiency in identifying offensive text. This model should be equipped to recognize offensive language and maintain its efficacy when integrated into real-time applications. By undertaking this project, the realm of natural language processing stands to benefit from valuable insights into the use of contextually dependent and independent word embedding techniques within the domain of text classification.

Subsequent sections of this report will delve into the background of the research, its objectives, the methodology followed, the conducted experiments, and the resultant findings.

LITERATURE REVIEW

Supervised machine learning methodologies are the dominant strategy employed for automated identification of hate speech (Schmidt and M. Wiegand, "A survey on hate speech detection using natural language processing," in Proceedings of the Fifth International Workshop on Natural Language Processing for Social Media, 2017, pp. 1-10). The detection of hate speech can be represented within machine learning as either binary or multiclass classification challenges, both of which can be effectively addressed through classical or deep learning algorithms. Classical approaches hinge on meticulously crafted features, while deep learning techniques autonomously derive features from raw input data, bypassing the need for manual feature engineering (T. Young, D. Hazarika, S. Poria, and E. Cambria, 2018). The inherent unstructured nature of human language poses numerous inherent difficulties for automated text classification methods.

A significant hurdle encountered by current hate speech detection techniques is their inability to capture extended contextual relationships. This inadequacy leads to the omission of crucial contextual details essential for semantic comprehension. Deep learning models, especially recurrent neural network

(RNN) architectures, have emerged as the standard methodology for managing sequential data like textual content (X. Wang, W. Jiang, and Z. Luo, "Combination of convolutional and recurrent neural network for sentiment analysis of short texts," in Proceedings of COLING 2016, the 26th international conference on computational linguistics: Technical papers, 2016, pp. 2428-2437.). Nevertheless, these models have constraints regarding the length of sequences they can effectively model (Z. Mhammedi, A. Hellicar, A. Rahman, and J. Bailey, "Efficient orthogonal parametrisation of recurrent neural networks using householder reflections," in International Conference on Machine Learning, 2017, pp. 2401-2409).

Transformers have emerged as a promising solution to capture prolonged contextual dependencies within text data. However, the technical obstacles associated with adapting Transformers for automated hate speech detection are far from straightforward.

RNN models such as long short-term memory (LSTM) (S. Hochreiter and J. Schmidhuber, "Long short-term memory," Neural computation, vol. 9, no. 8, pp. 1735-1780, 1997.) and gated recurrent units (J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, 2014) were intentionally designed to address the challenge of representing long-term dependencies—issues that other machine learning algorithms encounter. The Vanilla RNN operates by assigning greater importance to preceding data points in a sequence, rendering it suitable for classifying textual content in a manner that enhances semantic analysis (K. Kowsari et al., 2019). Nonetheless, RNNs are susceptible to challenges like gradient explosions and vanishing gradients during backpropagation training (Y. Bengio et al., 1994).

It is worthy to know that dealing with textual data requires a distinct strategy compared to numerical data, as computers cannot directly process text but rather

deal with numbers. This requires the transformation of non-numerical data (text) into numerical format, which is achieved through a process known as vectorization. The Term Frequency Inverse Document Frequency (TFIDF) is one such vectorization method that takes into account both the frequency of words or n-grams and the importance of tokens by assigning weights. Tokens that appear in numerous documents are considered less significant than those appearing in a select few (Dinakar et al., 2011). In their study on identifying instances of cyberbullying in text, Dinakar et al. utilized the TFIDF technique for vectorization and employed Support Vector Machines as the classifier, achieving an accuracy rate of 79%.

More recent work by Abro et al. (2020) adopted the same vectorization technique to discern hate speech in textual data. They employed the JRip model, which is a rule-based approach utilizing sequential covering, and achieved an accuracy rate of 73%. This investigation employed the bag-of-words (BOW) method in conjunction with TFIDF, representing textual data as vectors of word frequencies (the occurrence of words or n-grams). Koushik et al. (2019) applied this approach to automatically detect instances of hate speech on Twitter, achieving an accuracy rate of 94% using the Logistic Regression model as the classifier.

However, the vectorization techniques employed in these studies have limitations when it comes to understanding the nuanced meanings and context of words. To address these limitations, modern methods such as word embeddings have emerged. These methods represent text as high-dimensional vectors, capturing the semantic, syntactic, and contextual meanings of individual tokens in the text. Notable algorithms using word embeddings include word2vec, GloVe, and FastText.

In their research on identifying hate speech on Twitter, Robinson et al. (2018) compared the performance of models with meticulously crafted features using

machine learning algorithms (SVM) and models that automatically learn features through DNNs (CNN and GRU). Their research demonstrated that the ability of DNNs to process hierarchical data representations led to superior performance compared to the machine learning algorithms.

The capacity of DNNs to process hierarchical data representations makes them particularly well-suited for learning high-dimensional vector outputs generated by word embedding models. DNNs can be trained on these vectors, capturing the nuanced meaning and context of the text. Common DNN architectures include Convolutional Neural Networks (CNN) and Recurrent Neural Networks (RNN).

This research endeavors to evaluate the effectiveness of word embedding techniques (word2vec, GloVe, FastText) in tandem with RNN classifiers and transformer architecture (BERT). It compares these modern techniques with traditional approaches and machine learning models in the context of text classification

OBJECTIVE

The aim of this project is to construct an automatic text classification system utilizing advanced deep learning methodologies such as Recurrent Neural Networks (RNNs) and pre-trained transformer models. The objective is to create a classifier that can effectively label a given text as either offensive or nonoffensive. Multiple word embedding techniques will be investigated to enhance the system's ability to understand the underlying context and semantics of the text. The primary focus lies in evaluating and contrasting the performance of various classifier models within this framework.

METHODOLOGY

This segment outlines the proposed system designed for categorizing text into either "offensive" or "not offensive" categories. The complete research approach is visually represented in Figure 1. The research methodology is organized into six distinct phases: data collection, data preprocessing, visualization, vectorization, model architecture, and model evaluation.

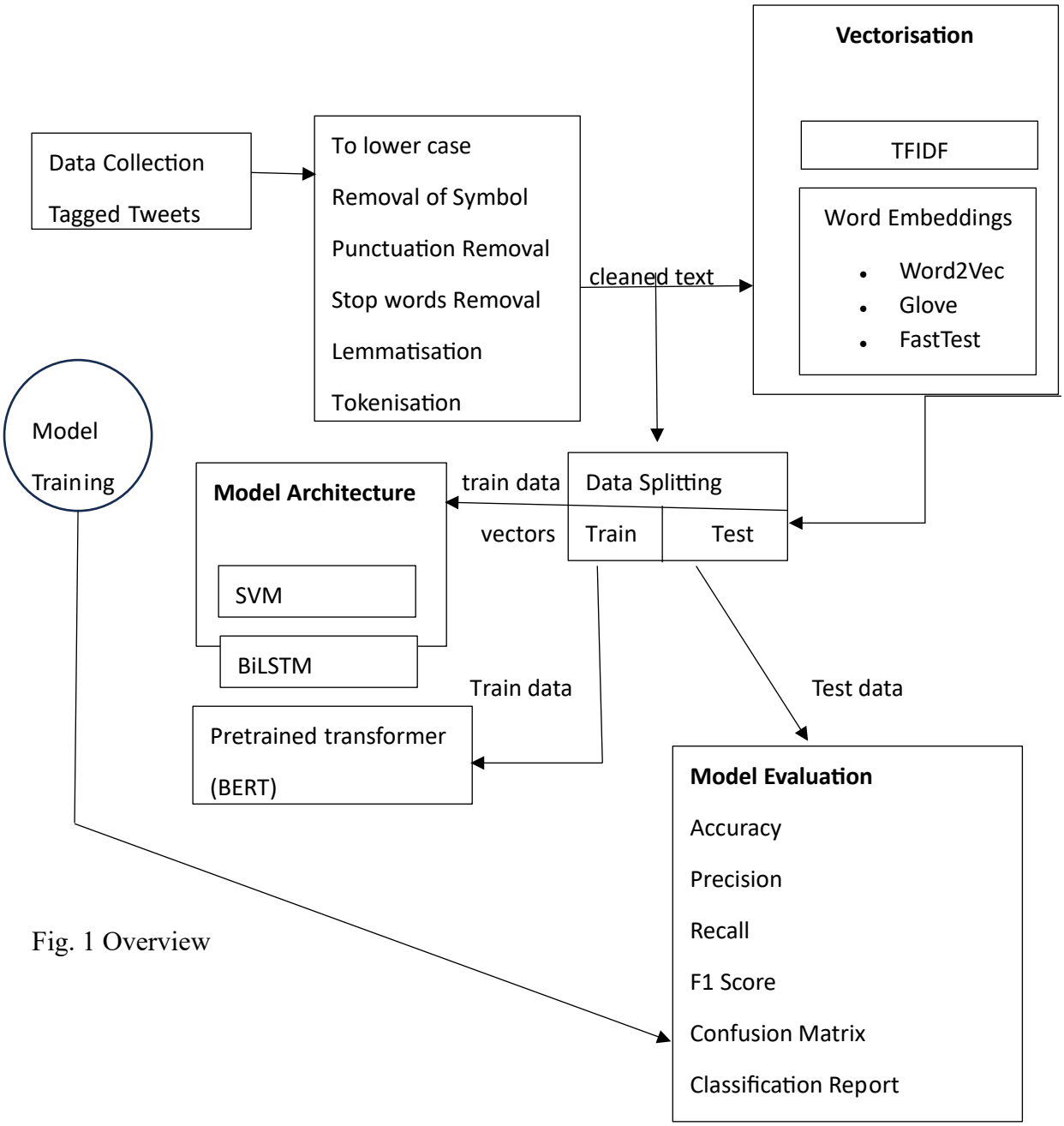


Fig. 1 Overview

Stage 1: DATA COLLECTION

This study utilized two distinct datasets for its research: the "tweet sentiments" dataset (Yasser, 2022) and the "Twitter hate and offensive language" dataset (Samoshyn, 2020). These datasets were sourced from Kaggle and come with associated labels.

Stage 2: DATA PREPROCESSING

The collected data underwent preprocessing to ensure a clean and uniform structure. This entails various tasks such as converting text to lowercase, eliminating punctuation, removing special characters, filtering out stop words, and applying lemmatization. During the punctuation and special character removal process, regular expressions (re) were utilized to identify and eliminate items adhering to specified patterns. This step effectively removed elements like usernames, mentions, hashtags, and white spaces.

For the removal of stop words, the NLTK's stopwords library was employed. This step was taken since stop words lack significant meaning in the context of natural language processing. Lemmatization was preferred over stemming, as it preserves the fundamental meaning of each word. To perform lemmatization, the WordNetLemmatizer library from NLTK was employed to process the words in the text.

The outcome of the preprocessing stage is a refined text that contains less noise and retains informative characteristics. These enhanced features provide valuable information for the classifier to learn during its training phase.

Stage 3: DATA VISUALIZATION

Following preprocessing, the data is subjected to visualization. This step involves presenting the distribution of the classes to assess class balance. Detecting an imbalance within the classes is crucial as it can cause the model to exhibit bias toward the dominant class. In Figure 2, the class distribution within the "Twitter hate and offensive language" dataset is displayed. This visualization underscores a substantial class imbalance, which has the potential to impact the model's efficacy. Consequently, an alternative dataset ("tweet sentiments") was considered to address this imbalance.

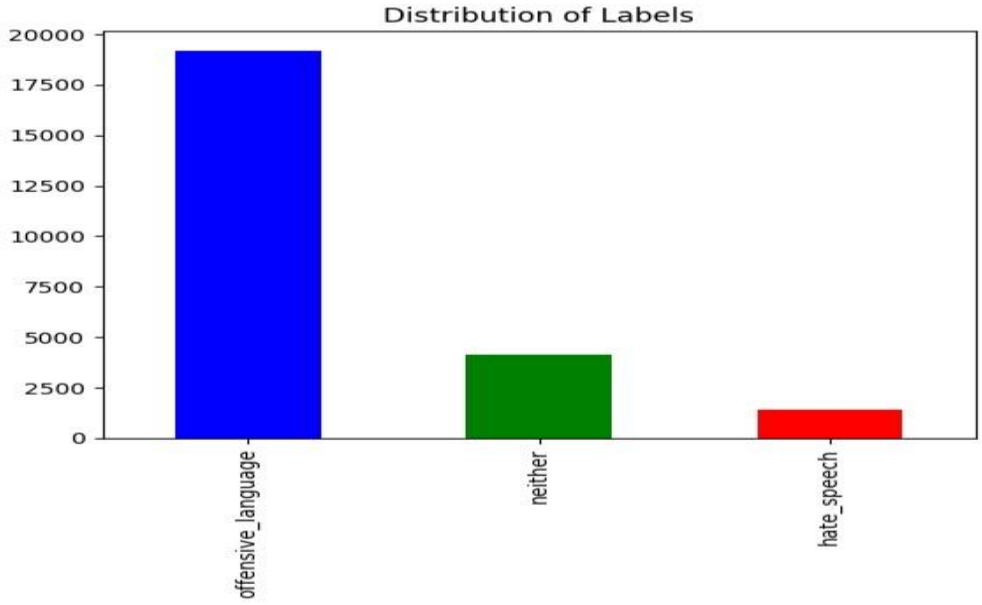

Distribution of Labels
20000
17500
15000
12500
10000
7500
5000
2500
0
offensive_language
neither
hate_speech

FIG. 2

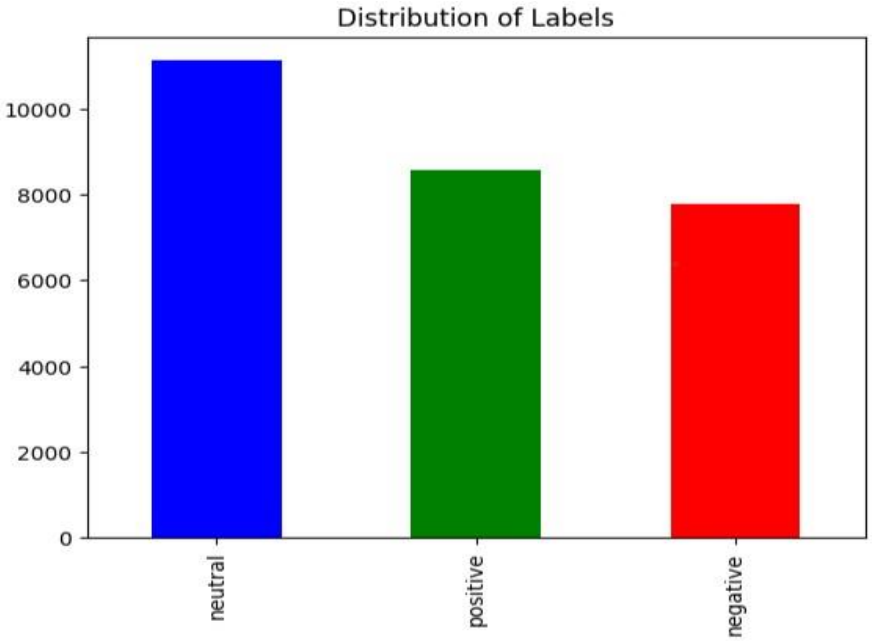

Distribution of Labels
10000
8000
6000
4000
2000
0
neutral
positive
negative

FIG. 3

The categories labeled as "offensive language" and "hate speech" were concatenated with over 17,000 randomly picked instances from the "neutral" and "positive" categories of dataset2. This merging resulted in a new dataset, depicted in the figure below.

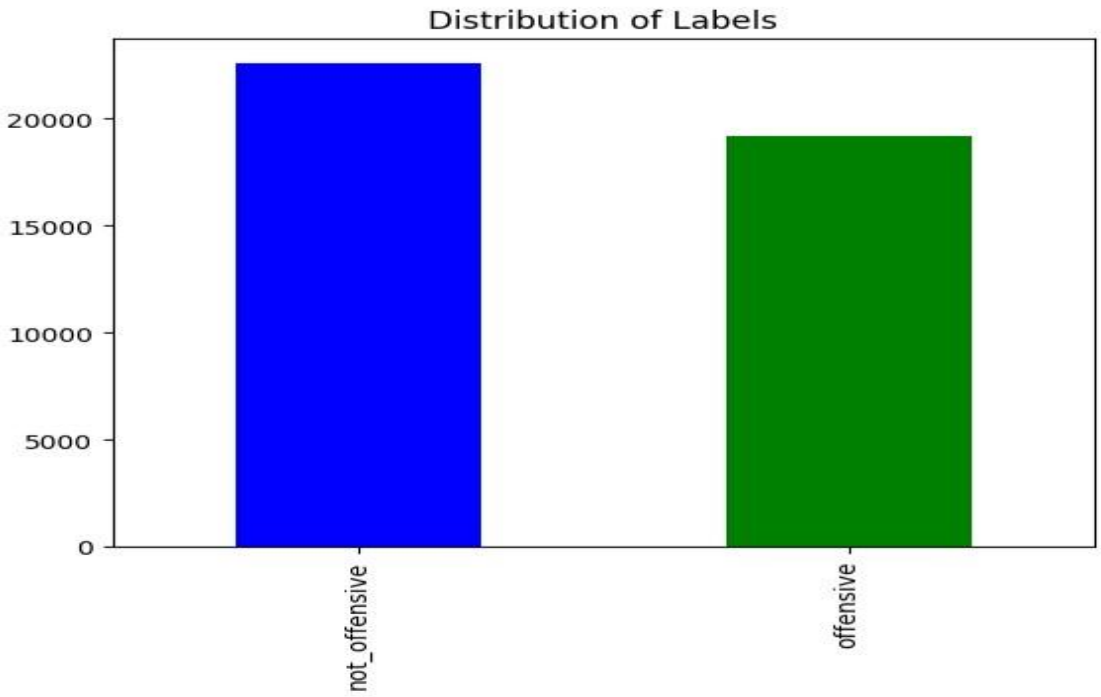


Fig. 4

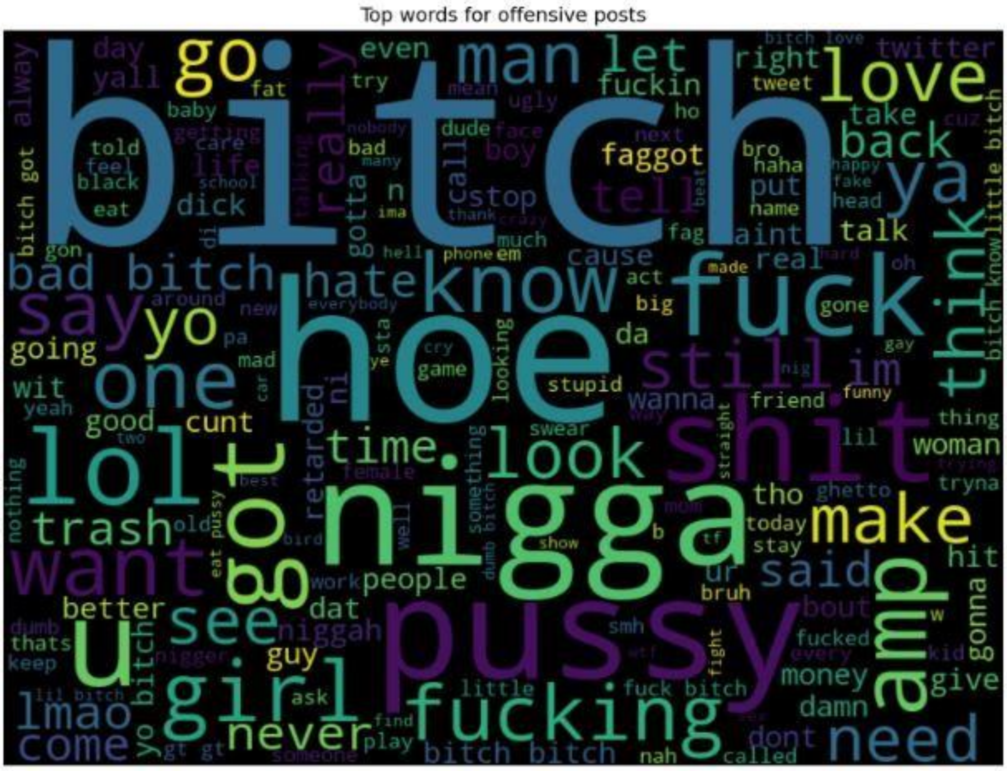

Top words for offensive posts
bitch
hoe
nigga
pussy
fuck
shit
lol
got
u
love
know
go
fucking
girl
one
want
see
amp
think
still
say
man
let
make
need
look
time
trash
yo
back
ya
really
hate
said
im
come
lmao
never
bad bitch

Fig. 5


Top words for Non_Offensive posts
love
good
u
lol
trash
thank
know
one
go
today
day
got
happy
think
need
new
going
back
hope
great
time
work
look
make
see
want
still
bird

Fig.6

Stage 4: VECTORISATION

In this phase, the cleaned text was transformed into numbers using word embedding techniques. For the baseline model (SVM), the data was turned into numbers using the TFIDF method. The word embedding methods used in this study include word2vec, GloVe, and FastText. Word2Vec, proposed by Mikolov et al. (2013), is a technique to represent large datasets as vectors. It's computationally efficient and performs better than previous approaches. GloVe, developed by Stanford University researchers, combines global matrix factorization with local context windows, resulting in better performance on tasks measuring similarity (Pennington et al., 2014). FastText was created to handle limitations of other models when dealing with unlabeled data (Bojanowski et al., 2017). These methods produce high-dimensional vectors for the neural network classifiers.

Schmidt and M. Wiegand, "A survey on hate speech detection using natural language processing," in Proceedings of the Fifth International Workshop on Natural Language Processing for social media, 2017, pp. 1-10.

Stage 5: MODEL ARCHITECTURE

The structure of the classifier models is established. The three models employed in this study are SVM (as the baseline model), BiLSTM, and the pre-trained transformer known as BERT.

The BiLSTM architecture is composed of two Long Short-Term Memory (LSTM) layers. One layer processes the input in the forward direction, while the other layer processes it in reverse. This configuration enables the network to grasp both past and future contextual information from the input data. These two layers are then combined at the output, and a dense layer is appended to generate the final output.

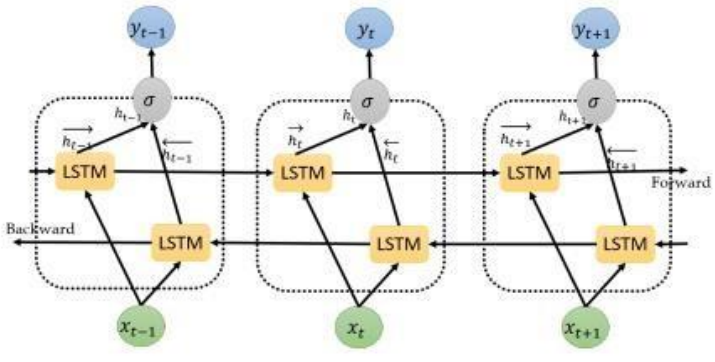


Fig.7: The structure of BiLSTM cell

The Transformer model architecture incorporates an attention mechanism (Vaswani et al., 2017) and stands out due to how it differentially weighs input sequences, unlike convolutional or recurrent models. In this study, the model of choice is BERT (Bidirectional Encoder Representations from Transformers). BERT employs the attention mechanism of the transformer to understand the contextual significance of words in relation to the sentences they belong to when forming word embeddings. This contrasts with conventional NLP models that typically focus only on preceding or subsequent words (Devlin & Chang, 2018). BERT's approach enables it to more effectively capture the meaning and intricacies of natural language.

BERT is a pre-trained model, extensively trained on a massive collection of text, including the complete Wikipedia corpus and the BookCorpus dataset. Its training involves a masked language modeling objective, allowing it to grasp intricate language patterns and nuances.

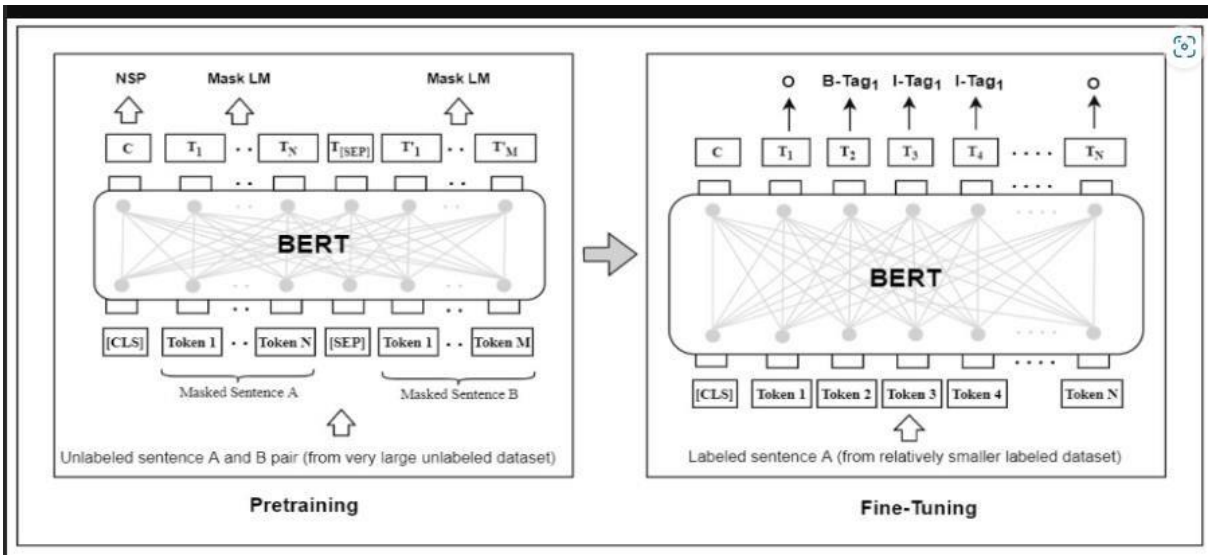


Fig.8: The structure of BERT

In the context of this research, the model was fine-tuned by connecting its output to a dense layer tailored for classification purposes.

Subsequently, the model was trained on the training dataset and optimized by evaluating its performance on the testing dataset.

Stage 6: MODEL EVALUATION

The model was evaluated using the test data. The model predicted the corresponding class ("offensive" or "not offensive") for each instance in the test set. To assess its performance, the results are evaluated through the calculation of confusion metrics.

The models' performances were assessed based on the following; accuracy, precision, recall, and F1 measure.

Accuracy: This metric quantifies the proportion of correct predictions relative to all predictions made by the model.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN},$$

Precison: measures the proportion of correctly predicted positive instances among all instances that were predicted as positive by the model. It focuses on the accuracy of the positive predictions.

$$Precision = \frac{TP}{TP + FP}.$$

Recall: This is also known as Sensitivity or True Positive Rate, gauges the ratio of correctly predicted positive instances to the total number of actual positive instances. It assesses the model's ability to identify positive instances correctly.

$$Recall = \frac{TP}{TP + FN}.$$

F1 Score: is a metric that combines both precision and recall into a single value, providing a balanced assessment of a model's performance. It is particularly useful when dealing with imbalanced classes.

$$F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall}.$$

This metric balances the trade-off between precision and recall, giving equal weight to both. It ranges between 0 and 1, where a higher value indicates better model performance.

EXPERIMENT

The dataset employed for this study was the final dataset that emerged after the concatenation of dataset1 and a subset of dataset1, yielding a class-balanced dataset as detailed in the methodology. The division between training and testing data was performed at an 80:20 ratio.

Three model architectures were used for the classification, namely SVM, BiLSTM, and fine-tuned BERT (pre-trained transformer) and four vectorization techniques were used; TFIDF, word2vec, GloVe, and FastText. The BERT model was fine-tuned by stacking it into a network of dense layers. The input to the model is the cleaned text.

To enhance the models' ability to generalize effectively, dropout layers of 20% were incorporated. The BiLSTM model was compiled using a learning rate of 1e3, while the fine-tuned BERT utilized a learning rate of 3e-5 (higher values led to suboptimal performance). The baseline model chosen for this study is the SVM, along with additional models from related research. The dataset was vectorized using the TFIDF technique.

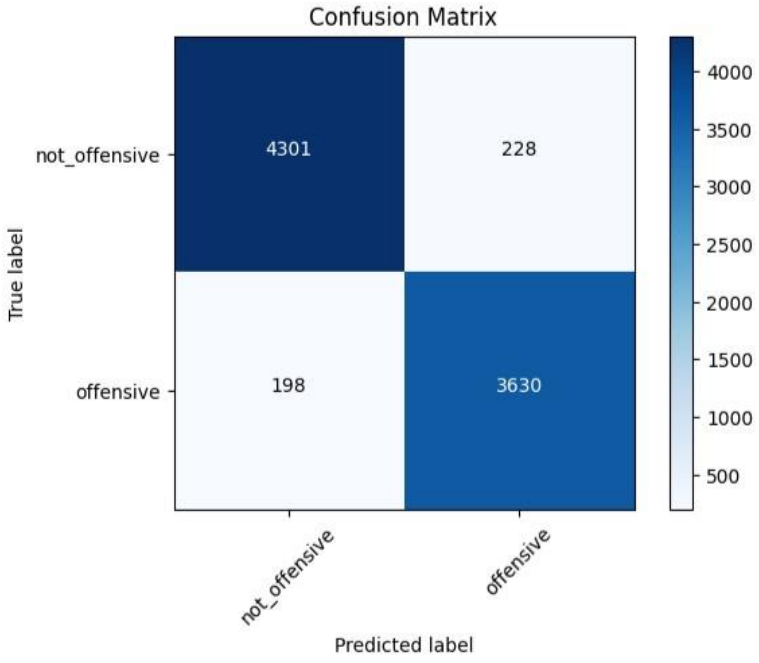
The evaluation metrics of accuracy, precision, recall and F1 were employed to compare the effectiveness of the models.

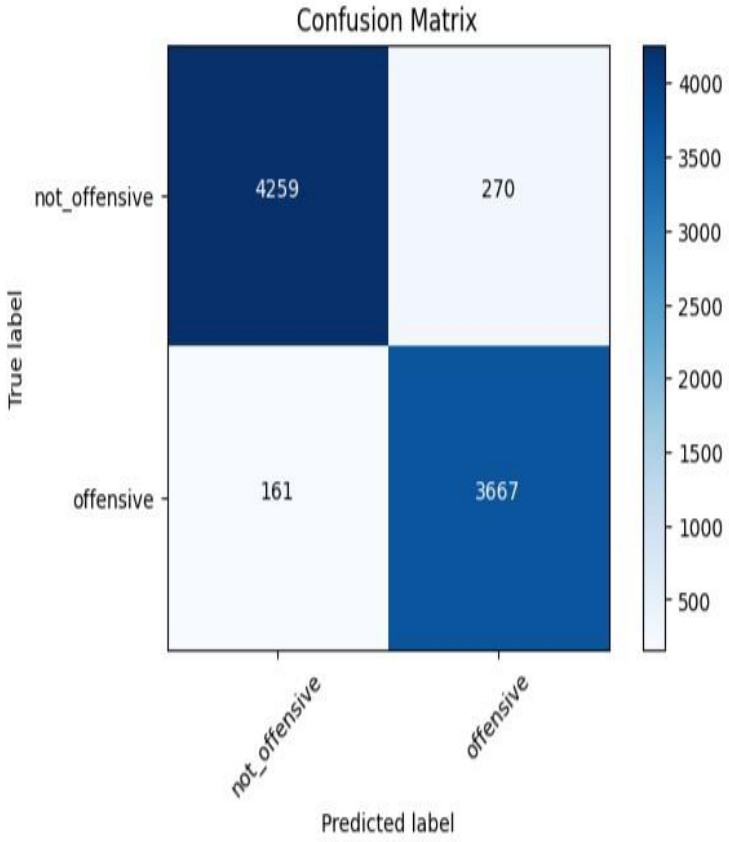
RESULTS

From the table below, the BERT ranked the highest while the BiLSTM using Word2Vec and FastText scored the least.

MODEL		SVM	BiLSTM			BERT
Embedding Techniques		TFIDF	Word2Vec	Glove	FastText	
metric	Accuracy	0.94	0.90	0.95	0.90	0.95
	Precision	0.94	0.89	0.95	0.89	0.95
	Recall	0.94	0.89	0.95	0.89	0.95
	F1-Score	0.94	0.89	0.95	0.89	0.95

Performance Metrics





The experimental results highlighted that both the BERT and GloVe models exhibited superior performance. Although the difference in performance is not extensive, BERT's success can be attributed to its status as a state-of-the-art model, having been pre-trained on an extensive volume of text and incorporating an advanced attention mechanism. This positions BERT advantageously for addressing considerably complex natural language processing tasks.

The notable performance of the GloVe word embedding technique can potentially be attributed to its capacity to capture the broader contextual statistics of word occurrences within the text corpus.

CONCLUSION

In summary, this research aimed to evaluate the efficacy of various word embedding methods and model architectures for the classification of offensive language in tweets. The outcomes demonstrated that BERT and the combination of GloVe with BiLSTM yielded the most favourable results.

The exceptional performance of BERT, a model pre-trained on extensive text data, underscores the significance of pre-training in natural language processing tasks. On the other hand, the commendable performance of GloVe can be attributed to its capability to capture comprehensive occurrence statistics of words throughout the text corpus. The findings offer valuable insights into the effectiveness of diverse techniques and models when it comes to categorizing offensive language in tweets. Importantly, they emphasize how the strategy of pre-training models and subsequently fine-tuning them for specific tasks can substantially enhance the performance of AI systems in real-world applications.

Future investigations could delve into the effectiveness of alternative pre-trained models and different word embedding techniques, particularly exploring the capabilities of generative pre-trained transformers such as GPT. Moreover, extending the study to other languages and diverse types of textual content, like comments and reviews, would provide a broader perspective on the generalizability of the findings.

REFERENCES

Al-Hassan and H. Al-Dossari, “Detection of hate speech in social networks: a survey on multilingual corpus,” in 6th International Conference on Computer Science and Information Technology, vol. 10, 2019.

A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever, “Language models are unsupervised multitask learners,” openai blog, vol. 1, no. 8, p. 9, 2019.

A. Assiri, A. Emam, and H. Al-Dossari, “Towards enhancement of a lexiconbased approach for Saudi dialect sentiment analysis,” Journal of information science, vol. 44, no. 2, pp. 184–202, 2018.

A. Arango, J. P´erez, and B. Poblete, “Hate speech detection is not as easy as you may think A closer look at model validation,” in Proceedings of the 42nd international acm sigir conference on research and development in information retrieval, 2019, pp. 45–54.

A. M. Founta, D. Chatzakou, N. Kourtellis, J. Blackburn, A. Vakali, and I. Leontiadis, “A unified deep learning architecture for abuse detection,” in Proceedings of the 10th ACM conference on web science, 2019, pp. 105–114.

B. Gamb¨ack and U. K. Sikdar, “Using convolutional neural networks to classify hate speech,” in Proceedings of the first workshop on abusive language online, 2017, pp. 85–90.

B. Ross, M. Rist, G. Carbonell, B. Cabrera, N. Kurowsky, and M. Wojatzki, “Measuring the reliability of hate speech annotations: The case of the European refugee crisis,” arxiv preprint arxiv:1701.08118, 2017.

B. Cheang, B. Wei, D. Kogan, H. Qiu, and M. Ahmed, "Language Representation Models for Fine-Grained Sentiment Classification," arxiv preprint arxiv:2005.13619, 2020

D. Chatzakou, N. Kourtellis, J. Blackburn, E. De Cristofaro, G. Stringhini, and A. Vakali, “Mean birds: Detecting aggression and bullying on twitter,” in Proceed ings of the 2017 ACM on web science conference.

E. Cambria, S. Poria, A. Gelbukh, and M. Thelwall, "Sentiment analysis is a big suitcase," IEEE Intelligent Systems, vol. 32, no. 6, pp. 74-80, 2017.

G. K. Pitsilis, H. Ramampiaro, and H. Langseth, "Effective hate-speech detection in Twitter data using recurrent neural networks," Applied Intelligence, vol. 48, no. 12, pp. 4730-4742, 2018.

J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," arxiv preprint arxiv:1810.04805, 2018.

J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," arxiv preprint arxiv:1412.3555, 2014.

K. Kowsari, K. Jafari Meimandi, M. Heidarysafa, S. Mendu, L. Barnes, and D. Brown, "Text classification algorithms: A survey," Information, vol. 10, no. 4, p. 150, 2019.

M.S. Islam, S. S. S. Mousumi, S. Abujar, and S. A. Hossain, "Sequence-tosequence Bangla sentence generation with LSTM recurrent neural networks," Procedia Computer Science, vol. 152, pp. 51-58, 2019.

N. Djuric, J. Zhou, R. Morris, M. Grbovic, V. Radosavljevic, and N. Bhamidipati, “Hate speech detection with comment embeddings,” in Proceedings of the 24th international conference on world wide web, 2015, pp. 29–30.

Z. Mhammedi, A. Hellicar, A. Rahman, and J. Bailey, "Efficient orthogonal parametrisation of recurrent neural networks using householder reflections," in International Conference on Machine Learning, 2017, pp. 2401-2409.

Z. Zhang, D. Robinson, and J. Tepper, "Detecting hate speech on twitter using a convolution-gru based deep neural network," in European semantic web conference, 2018: Springer, pp. 745-760.

Z. Waseem and D. Hovy, "Hateful symbols or hateful people? Predictive features for hate speech detection on twitter," in Proceedings of the NAACL student research workshop, 2016, pp. 88-93.