ISRG Journal of Arts, Humanities and Social Sciences (ISRGJAHSS)

ISRG PUBLISHERS
Abbreviated Key Title: ISRG J Arts Humanit Soc Sci
ISSN: 2583-7672 (Online)
Journal homepage: https://isrgpublishers.com/isrgjahss
Volume – III Issue-I (January- February) 2025
Frequency: Bimonthly

An Effective Hate Speech Detection using Coral Reef Optimization for Social Media

FHA. Shibly[1*], AR. Mohamed Mahir[2] , and A. Mohamed Jabir[3]

[1] Senior Lecturer (Gr. I), South Eastern University of Sri Lanka, shiblyfh@seu.ac.lk
[2] Senior Assistant Registrar, Eastern University Sri Lanka
[3] Chief Security Officer (Gr. I), South Eastern University of Sri Lanka

| **Received:** 25.12.2024 | **Accepted:** 30.12.2024 | **Published:** 05.01.2025

***Corresponding author:** FHA. Shibly
Senior Lecturer (Gr. I), South Eastern University of Sri Lanka, shiblyfh@seu.ac.lk

Abstract

On social networking sites, online hate speech has become more prevalent due to the quick expansion of mobile computing and Web technology. Previous research has found that being exposed to internet hate speech has substantial offline implications for historically disadvantaged communities. As a result, research into automated hate speech detection has received a lot of interest. Hate speech can have an influence on any population group, but some are more vulnerable than others. An effective automatic hate speech detection ideal, which based on progressive natural language processing and machine learning is not adequate. We need annotated datasets of a size sufficient to train a model. Lack of properly labeled hate speech data, as well as existing biases, have been the biggest obstacles in this field of study for years. To meet these needs, we provide in this paper an unique coral reefs optimization-based method with a transfer learning attitude based BERT (Bidirectional Encoder Representations from Transformers). An optimization approach for coral reefs that simulates coral behaviors for placement and growth in reefs is known as a coral reefs optimization algorithm. In the projected strategy, each solution to the problem is viewed as a coral that is always attempting to be planted and grow in the reefs. Special operators from the coral reefs optimization algorithm are applied to the results at each phase. When all is said and done, the best solution is chosen as the ultimate solution to the issue. A new fine-tuning strategy based on transfer learning is also used to evaluate BERT's ability to capture hateful context in social media posts. To evaluate proposed method, we use datasets that have been annotated for racism, sexism, hate, or objectionable content on Twitter. The results show that our solution outperforms earlier techniques in terms of precision and recall.

Keywords: *Natural Language Processing; Bidirectional Encoder Representations from Transformers; Coral Reefs Optimization; Hate speech Detection; Twitter.*

DOI: 10.5281/zenodo.14600781

Introduction

Methods for problems linked to Opinion Mining & Sentiment Analysis (OM& SA) have become more important in the field of artificial intelligence, particularly in NLP. In most cases, these strategies are used to get customer feedback on a product or to gauge political sentiment [1]. Strong and successful approaches are now possible because of significant improvements in supervised learning technologies, as well as the abundance of user-generated content available online, notably on social media platforms. Recent years have seen an growing interest in jobs connected to social and ethical issues in the NLP community, which has been bolstered by the global commitment to combating extremism, violence, fake news, and other internet plagues. Another is hate speech, a toxic discourse rooted in preconceptions and intolerance that can lead to acts of violence, and persecution, and even to structured policies [2].

The sheer volume of comments posted every second on social media platforms makes it impossible to track the content of such sites. As a result, social media companies have a difficult time balancing the need to limit offensive posts with the right to free speech [3]. Hate speech can also be sparked by the diversity of people, ethnicities, cultures, and beliefs [4]. However, every culture has its own unique clarifications and features of cyber-hate that are unique to that society. It is assumed that every culture acts differently and intervenes in a way that best matches the culture [5].

This endeavor is difficult due to the fact that different hate speech definitions exist. Because of this, some things may offend some people, but not others, depending on how they define it. Yet removing hate speech data by hand is labor-intensive and time-consuming. Automatic hate speech papers is a necessity since of these concerns and the ubiquitous presence of hate speech content on the internet. Using the Coral Reef algorithm and BERT, we propose to identify susceptible communities by detecting hate speech in social media. The evaluation of the proposed method is carried out using Twitter dataset, which is clarified in the upcoming sections. The paper is prearranged as: Section 2 presents the study of related works on hate speech detection. The description of proposed procedure is given in Section 3. The validation of proposed method with existing techniques is provided in Section 4. Finally, the work conclusion with future direction is described in Section 5.

Related works

To this end, Chung et al. [6] constructed an extensive corpus of HS-counterspeech pairs, focusing on positive answers rather than just negative ones. The fact that the annotators are NGOs activists with training and experience in opposing and averting HS adds to the significance of this work. Their expertise could be particularly beneficial in establishing such tools. A example shift in the employment of NLP skills to handle abusive language is called for by Jurgens et al. [7]. Only certain types of abusive content are lectured, while others are ignored because they are either too subtle or too rare. All toxic or abusive language, they say, should be addressed. This includes micro aggressions and insults, which they say also contribute to a poor environment.

To identify cyber hatred on Twitter, Peter Burnap and colleagues [8] used a dictionary-based method. A N-gram feature engineering technique was used in this study to construct numeric vectors from a predetermined vocabulary of hostile phrases. When the authors submitted the produced numeric vector to SVM, they received an F-score of up to 67%. When Njagi Dennis et al. [9] classified hate speech in web opportunities and blogs, they utilized a machine-learning-based classification system. To construct a master feature vector, the authors used a dictionary-based technique. Semantic and subjective traits with a bias towards hate speech were used to create the features. To classify them, they fed a rule-based classifier the master feature vector. It was found that their classifier was 73% accurate in experimental situations when measured by precision performance.

This approach was also utilized by Stéphan Tulkens and colleagues [10] to automatically detect racism in Dutch social media. They looked at how words were distributed throughout three dictionaries. They passed the created features to the SVM classifier, which then classified the data based on the features. A 0.46 F-Score was derived from their research. A study conducted by Sebastian Köffer et al. [11] classified hate speech messages in German texts using word2vec features and SVM classifiers, achieving an F-score of 67%. In terms of results, word2Vec was the least successful since it requires a large amount of data to understand complicated word semantics.

Proposed System

In this section, the explanation of proposed methodology is given. There are five major parts such as dataset description, pre-processing, feature extraction, feature selection and classification.

Dataset Description

136,052 tweets were collected during a two-month period and 16,914 were tagged [12]. We manually marked and categorized tweets as either racist, sexist, or neither. 16.907 tweet IDs and their labels were published by the authors. Tweepy18's Python library detected just 15,844 tweets, categorized into three classifications. Some tweets were either erased or had their visibility altered — Twitter has the capacity to censor tweets, and users often erase their individual tweets.

Pre-Processing

Table 1: pre-processing procedures

Pre-processing Method
URLs, user-mentions & hashtag symbol
Lower-casing of words
Replace Abbreviations and Slang
Removing Punctuation
Expanding Contractions
Removing Stop-words
Replace Emoticons
Removing Numbers
Lemmatization
Incorrect Spellings
Words Segmentation
Elongated Characters

URLs, user references and hashtag symbols

In order to give users with more information, utmost tweets include URLs, user mentions and hashtag symbols. Although this extra

DOI: 10.5281/zenodo.14600781

info is deemed helpful for humans, most text analytic jobs consider it to be a nuisance and of little use. We deleted all URLs, user references, and hashtag symbols from our research.

Abbreviations and slang

Earlier, I said that Twitter's character limit forces users to utilize alternative acronyms and jargon. If each user writes in his or her own manner and utilizes diverse acronyms, this becomes much more challenging. These are abbreviations and phrases that are sometimes linked with a particular context or a particular group of individuals. Replace these errors with their corresponding meanings in order for a machine to understand them.

Expanding Contractions

As a pre-processing technique, expanding contractions can be very valuable, especially before tokenizing, because tokenizing will turn can't into two different tokens, can and t, which is absurd. Contractions can be preserved by expanding them, because not is an essential part of the sentence for classification purposes.

Removing Numbers

However, numerals do not always deliver useful information for text organisation, hence it is usual repetition to delete them from the corpus of documents. Eliminating statistics too soon, on the other hand, may result in lost information. It's possible to worsen the results by removing the 8 from gr8. Numbers should always be removed when acronyms and slang have been replaced with the corresponding word meanings.

Replace Emoticons

Emoticons are a way for people to express themselves on social media. People can grasp the feelings and sentiments after emoticons, but machines need to be furnished with the word meanings of the emoticons' symbols. We charity the Ekphrasis library to substitute emoticons with their word meanings in order to extract the most info from our trials.

Lemmatization

In lemmatization, root words are substituted for the word being lemmatized. This step was performed using the WordNet lemmatizer15 library in our analysis.

Removing punctuation

To eliminate punctuation is a traditional and widespread pre-processing strategy in text classification. While punctuation helps people grasp opinions and feelings, it has little impact on machine classification. As a result, we deleted all punctuation from our study.

Word Segmentation

Since tweets have a character limit of 140 characters, it encourages users to communicate in an unstructured and informal manner Humans can read and understand these concatenated strings, but machines need a little help to understand them. After deleting the hashtag symbol, we separated the remaining content/phrases.

Lower-casing of words

One of the most used preprocessing techniques is case folding. It reduces the dimensionality of the corpus and helps match words with the same meaning.

Removing Stop words

Natural language articles and prepositions add nuance to the language, but they don't always help classify the content. For example, words such as the, a, am, are, on, at, and so on and so forth. In pre-processing ordering tasks, removing stop words is a frequent method. Stop words were detached from the corpus using the NLTK Stop-word library16.

Elongated Characters

To prevent the beginner from treating extended terms differently from their base words, appeals that are frequent three times in a row are condensed to a single appeal.

Incorrect Spelling

It's not uncommon for social media posts and communications to be misspelled. Occasionally, users will purposely misspell words as a kind of stylization, for example, hav for have. In this investigation, we additionally examine the consequence of correcting spelling errors. This study makes use of Norvig's spell checker17.

Feature Extraction

It is necessary to specify generic properties of the corpus in order for the classification algorithms to conduct an spontaneous detection task, such as hate speech identification. Several of these approaches will be discussed in the following paragraphs.

Dictionaries and Lexicons

In unsupervised machine learning scenarios, this characteristic is commonly used [13]. Wiegand et al. [14] used corpora and lexical resources to detect profane terms. A general-purpose lexical resource and numerous traits were employed to create their lexicon. Since lexicon-based techniques are domain-independent, they do not compete with other features employed in supervised algorithms. It was also utilized by Gitari et al. [15] to aggregate opinions and rate subjective words.

Bag-of-words (BOW) and N-grams

It can be viewed as a feature of word cooccurrence. By giving weight to each word based on its frequency in the tweet and its incidence amongst various tweets, a vectorization procedure is performed on tokenized terms in the corpus (e.g. TF-IDF weight). An alphabetical list will be created and offered as vectors of weights [16]. In n-gram representation, N words are arranged in a row. For hate speech detection, the study looked into the impact of combining a number of variables in conjunction with character N-grams. A character n-gram representation has been found to be a powerful tool for detecting hate speech. This is due to the fact that BOW is limited by the fact that it must be complemented by other characteristics in order to boost performance. In the case of N-grams, the value of N must be carefully selected to avoid a high amount of distance among related words [18].

Latent Dirichlet Allocation (LDA)

A probabilistic topic modeling technique, it is based on probabilities. These latent topics will serve as features as an alternative of words. For unsupervised and semi-supervised machine learning environments, LDA is a viable option. When it comes to abusive text recognition in twitter, Xiang et al. [19] say that BOW does not perform well. As an alternative to the supervised approaches, they include extremely sensitive topical characteristics and other lexical aspects by employing LDA model.

Word embedding and Word2Vec

Although data sparsity has been addressed by word embedding, a semantic element has been added by producing distributed representations that introduce reliance between words. To construct word embedding, Word2Vec is a technique. Lilleberg et al. [20] report that researchers in the field of text mining are very interested

DOI: 10.5281/zenodo.14600781

in word2vec because of its compatibility with machine-learning prototypes.

Coral reefs optimization method

This meta-heuristic program replicates coral growth in reefs. As its corals, this method starts with a population of various coded solutions. Randomly generated corals are then arranged into a square grid, which forms the reef. In the beginning, the square grid's cells are empty and the corals of the original population are arbitrarily positioned in the grid's cells at random. A few cells in the square grid should be left empty, allowing new corals to grow in the following step when the sum of solutions is less than the sum of square grid cells. It is also necessary to have a health function in order to check the solutions and come up with better ones. Optimizing for health might be the same objective function. On the basis of coral reproduction and reef formations, the algorithm operates. Different operators are used to reproduce corals again in order to find superior solutions. Randomly picked corals are used for external reproduction. They are then selected for reproduction in pairs. The cross-over operator will create new solutions from these pairings of solutions. A roulette wheel, for example, can be used to select pairs of solutions at random. In this phase, each pair of solutions yields only one new solution. There is no instant placement of the new solutions in the grid and they are released into the water. An additional fraction of the solutions will be used for internal reproduction. In this step, new solutions are created by applying random mutations to selected solutions. At this point, just like in Step 1, all of the solutions are released into the water. Coral reef formation is a kind of competition for space. In each phase of the algorithm, the corals produced strive to be placed on the reef. Here, success depends on the coral's power (how much it can optimize an issue) or its ability to identify an empty spot. As a result, the internal and outside reproduction operators' solutions are applied to the reef. First, for each new solution, the health function value is determined.

Each answer is then assigned a randomly picked square grid cell. If this cell is vacant, the solution will be placed in it if it is empty. If the cell is already occupied, and the new solution's health function is larger than the old solution's health function, the new solution will be placed in the cell. The novel solution cannot be placed in the cell if it is not sterile. For each solution, a maximum number of placement tries is defined, and if the maximum number of efforts is exceeded, the solution is considered obsolete. This is followed by an operation called asexual reproduction, which is applied to all of the grid solutions. This is accomplished by sorting all solutions based on their health function values, and a percentage of the best solutions reproduce themselves. Similar to the previous step, new solutions try to attach themselves to the reef. It is necessary to ensure that there are enough vacant areas on the reef for the following stage by implementing the coral depredation process (to exclude incorrect solutions) at the end of each algorithmic step. For this aim, a certain percentage of the poorest solutions is held aside. Thereafter, the procedure is repeated until a conclusion is reached. As an example, generating a given sum of generations set by the user can be regarded the algorithm's termination condition. As the algorithm progresses, a health function determines whether or not the solutions are fit. The following are the steps of the CRO technique.

Initialization:

It starts with a matrix R that has N rows and M columns (N M). This is followed by a randomization process that creates a random population of solutions that are randomly located in the matrix' cells. One solution is allowed per cell. No more than or no less than the amount of reef cells should make up a population. All cells of the matrix are engaged in this situation, and novel solutions have a low chance of being placed and growing in the matrix in the next steps. When each random solution has been created and placed in the matrix, a ratio of unfilled cells to occupied cells is determined. As soon as this ratio falls below 0.4, initial population generation will be stopped. For instance, a 10 x 10 matrix with 100 cells can yield 72 random solutions (28/72 0:4). After each solution is generated, it is added to a list and given an identifier. Randomly, this identification is inserted in a matrix cell. Moreover, after making any solution, the procedure calculates the value of its health function.

External reproduction operator:

To solve the problems, this operator is applied in two steps. As a first step, a random selection of matrices solutions is made. Fb is a user-specified parameter that determines how many solutions should be selected. As it turns out, Fb is equal to the ratio of all the matrix's solutions divided by the sum of designated solutions for external reproduction. For the external reproduction to work, the sum of designated solutions should be even. There is a distinct list for the picked and non-selected solutions in this stage. A roulette wheel approach is then used in the second phase to select two separate pairs from the list of designated solutions and apply the cross-over operator to each of them in order to generate novel solutions. If you want to apply the cross-over operator to each pair of solutions, you need to consider three separate random points (between 1 and n-1). A pair's solution can be broken down into four sections based on these three points. A novel solution is created by combining two larger chunks of the better solution with two lesser parts of the weaker one. If the sizes of the pieces are equal, the better option is the one that gets the nod. Out of each pair of selected solutions, a new solution is formed. In this step, new solutions are not added to the matrix. When they are released into the ocean, they are placed on a list of novel solutions so that they can subsequently be placed on the reef.

Internal reproduction operator:

A solutions list that weren't used by the outside reproduction operator is used by this operation to undertake operations. Defined this way, the ratio of solutions that are reproduced internally to all other solutions is 1-Fb. A random mutation is applied to each solution via the internal reproduction algorithm. One bit of the solution is inverted for this purpose, and a new solution will be formed as a result.

Setting new solutions:

Here, each solution tries to fit into one of the cells in the matrix. This is done by calculating each solution's value of the health function, and then examining how likely each answer is to be positioned in a cell of the matrix. A new solutions is found by selecting a random cell from the matrix. A solution identification will be positioned in the empty cell and the solution will be additional to population if the cell is empty. It's possible to replace the old solution in a cell if the value of the new solution's health function is greater than that of its predecessor's health function. That cell will not be used if a new solution is not applied. In the matrix, each solution can attempt to be inserted h times, where h is a user parameter.

DOI: 10.5281/zenodo.14600781

Asexual replica operator:
This operator uses some of the reef's best solutions. Here, the matrix solutions are arranged by their health function values. This is followed by a random selection from the beginning members of the sorted list to be reproduced. Each of these answers is duplicated, resulting in a new solution that is identical to its predecessor. A user-specified component of the problem, parameter Fa, determines how much of asexual reproduction occurs. It is next attempted to insert the new solutions, identical to what was done before, in a matrix similar to the prior stage.

Depredation:
Many of the reef's weakened solutions are removed at this point in order to create more empty cells in the matrix, giving fresh solutions a chance to grow in the matrix. Use the ordered list from step one, and choose a fraction Fd of worst solutions from it. Fd is also a user-specified parameter that can be overridden by the user. An integer between 0 and 1 is generated for each of these solutions, and if this number is less than Pd, then this solution is removed from the matrix and its consistent cell will be liberated. The user-defined probability of depredation is specified by the Pd parameter. After a few repetitions, the ratio of old solutions should be low and stable. A large number of vacant cells in the matrix allow new solutions to be put and grown. A bigger amount of weak solutions should be eliminated after numerous iterations. 0.1/k is added to Pd after each iteration. Depredation operates on the worst solutions and asexual reproduction operates on the best solutions.

CRO will be discontinued after k iterations of producing populations, as defined by the user. Final solution of the CRO is determined by the best matrices. FIG. 1 shows an overview of the proposed technique and a summary of each phase's results.

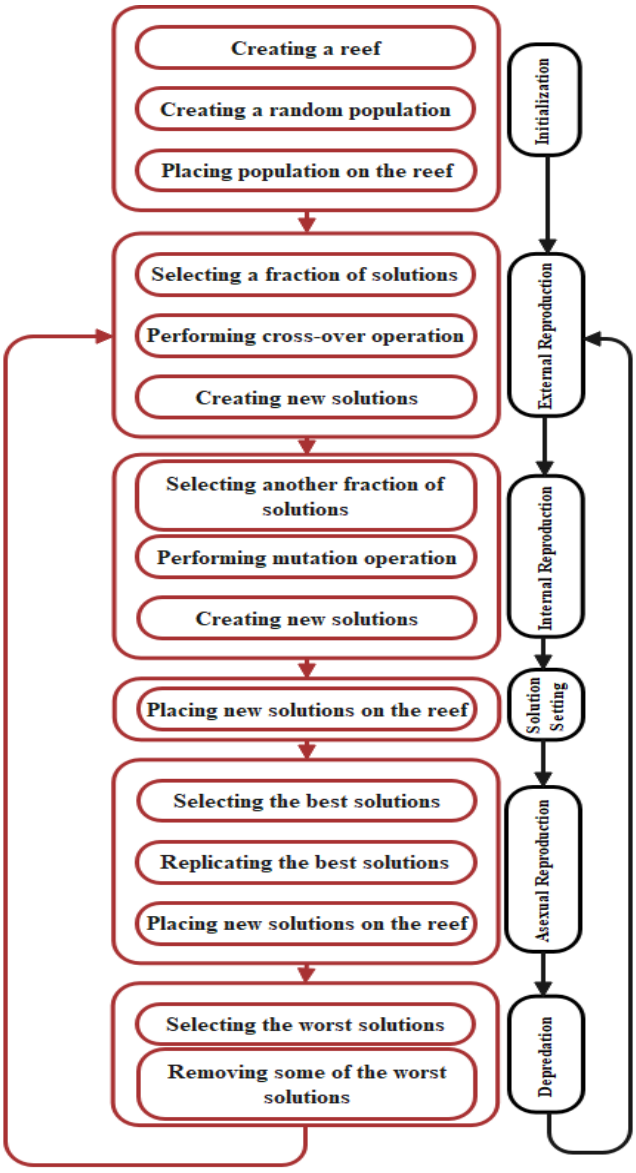
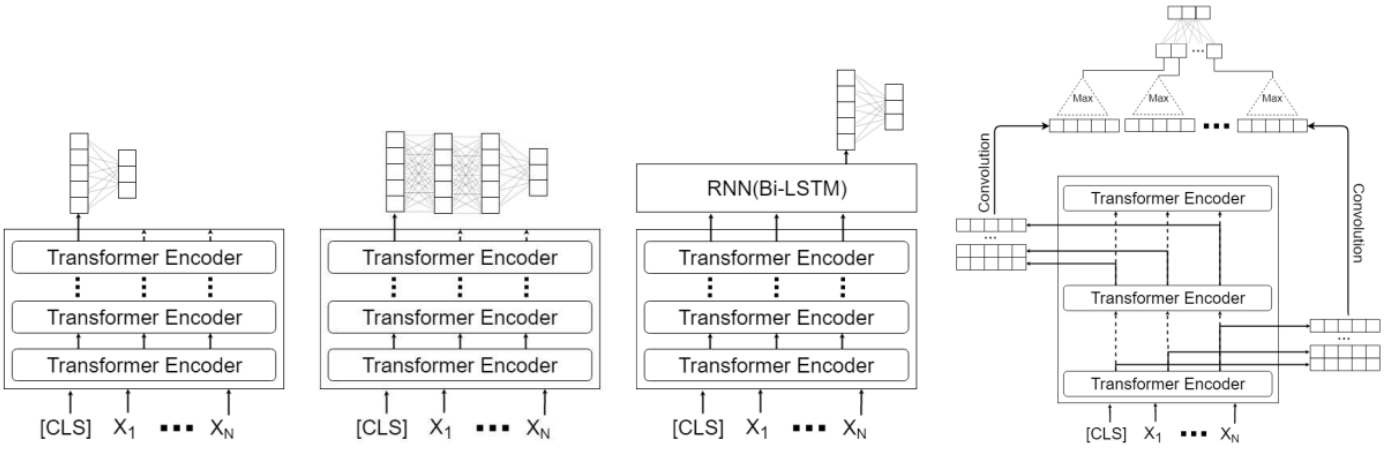


Fig. 1 The overall process of CRO

Classification using BERT
In this paper, we examine the BERT transformer model's performance in detecting hate speech. There are two versions of BERT: BERTbase and BERTlarge. BERTbase is a multi-layer, which trained on the the Book Corpus and English Wikipedia, which contain 2,500M tokens and 800M tokens, respectively. When compared to BERTlarge, BERTbase contains an encoder with 24 layers, 340 million parameters and 16 attention heads. BERTbase embeddings have 768 hidden dimensions, according to their extracted embeddings. Our hate speech identification work requires us to assess the contextual information derived from BERT's pre-trained layers, and then fine-tune it using annotated datasets. It takes a sequence of tokens with a maximum length of 512 as input and outputs a 768-dimensional vector representation of that sequence. [CLS] and

DOI: 10.5281/zenodo.14600781

[SEP] are the only segments that BERT inserts into each input sequence. There's a specific classification embedding in [CLS] that we use as a representation of the entire sequence in hate speech ordering tasks; we use [CLS] embedding as a first token in the final hidden layer. As a segment separator, the [SEP] will not be used in our classification task. The BERTbase model classifies each tweet in our datasets as Racism, Sexism, Neither, or As Hate, Offensive, Neither. So we attention on fine-tuning the BERTbase parameters that have already been taught. This means training a classifier with layers of 768 dimensions on top of the BERTbase transformer to minimalize task-specific parameters, which is what we call fine-tuning.



(a) BERTbase fine-tuning (b) nonlinear layers (c) Bi-LSTM layer (d) CNN layer

Fig. 2: Fine-tuning plans

Fine-Tuning Strategies

It is possible to capture multiple degrees of syntactic and semantic information using different layers of neural networks. This model has two layers: a general layer and a task-specific layer, which can be fine-tuned by changing the learning rates. As a result of our classification work, four different fine-tuning procedures have been established, utilizing pre-trained BERTbase transformer encoders. The architectures of these transformer encoders are described in further detail. At this stage, the model is initialized with pre-trained parameters, before being refined with the use of labelled datasets, which are used to fine-tune it. As shown in Figure 2, where Xi is represented as the vector representation of token I in the tweet sample, diverse fine-tuning attitudes for the hate speech identification job are described as follows:

BERT based fine-tuning:

When using the first strategy, which is seen in Figure 2a, the BERTbase undergoes very few changes. A single token output from BERT is used in this architecture: output of [CLS] tokens provided by BERT With no hidden layer, the [CLS] output is given as input, which is equivalent to the [CLS] token output of the 12th transformer encoder.

Insert nonlinear layers:

Instead of employing a completely linked network without a hidden layer network with two hidden layers of size 768 is employed. However, the last layer, which is the first construction, uses softmax activation function as illustrated in Figure 2b.

Insert Bi-LSTM layer:

A Bi-LSTM receives all outputs of the newest transformer encoder, as illustrated in Figure 2c, instead of just [CLS]. It delivers the concealed state to a fully connected network that uses the softmax activation function to classify the input.

Insert CNN layer:

Instead of using the output of the most recent transformer encoder, this architecture (shown in Figure 2d) uses the outputs of altogether transformer encoders. Each transformer encoder output vector is concatenated, and a matrix is created. As a result of the convolutional process, the extreme value is obtained for each transformer encoder by smearing max pooling on the convolution output. This creates a vector that can be fed into a fully connected network. The categorization operation is carried out by applying softmax to the input.

Results and Discussion

Performance Metrics

In the disciplines of data mining and data retrieval, evaluating the accuracy of machine learning classifiers is one of the most important phases. Error rate and F-measure are widely used to determine the accuracy of a classifier's ability to locate the proper category or class of unknown cases. The error rate is the instances of the test set that were erroneously categorised. We'll call this set of data "X" and let "m" represent how many occurrences were misclassified by a classification model C. You can calculate the accuracy of C in selecting the correct classes of X instances using the following formula:

$$Accuracy(C) = \frac{m}{n} \qquad (1)$$

DOI: 10.5281/zenodo.14600781

The error rate approach ignores the cost of inaccurate predictions in machine learning. For the most part, F-measure is used to solve this problem. To determine the value of F-measure, two basic metrics are used: precision and recall. Imagine that some of the data in the test set belong to a certain class or category S. It assigns a category label to each test data. There will be four kinds of forecasts for the test set S:

Percentage of accurately forecast data for category S is known as precision. Percentage of correctly forecast real data for category S is known as recall. It is possible to calculate the F-measure based on precision and recall (2-4).

$$Precision = \frac{|TP|}{|TP|+|FP|} \quad (2)$$

$$Recall = \frac{|TP|}{|TP|+|FN|} \quad (3)$$

$$F - measure = 2 \cdot \frac{Precision.Recall}{Precision+Recall} \quad (4)$$

Performance Analysis of Proposed Technique for Binary Class

In this analysis, two types of trials are carried out to test the performance of proposed CRO and BERT for binary class, i.e. Normal speech and Hate speech. Table 2 shows the comparative analysis of proposed CRO with existing meta-heuristic techniques in terms of accuracy, recall, precision and F-measure. Figure 3 shows the graphical representation of the comparative analysis.

Table 2: Comparative Analysis of Proposed Feature Selection Technique

Feature Selection Methodology	Parameter Evaluation			
	Accuracy (%)	Precision (%)	Recall (%)	F-measure (%)
PSO	87.89	79.12	80.92	85.27
GA	72.30	72.50	73.69	73.07
ABC	81.25	65.07	88.06	69.28
ACO	77.26	92.04	93.17	94.08
Proposed CRO	**95.20**	**97.64**	**98.20**	**98.67**

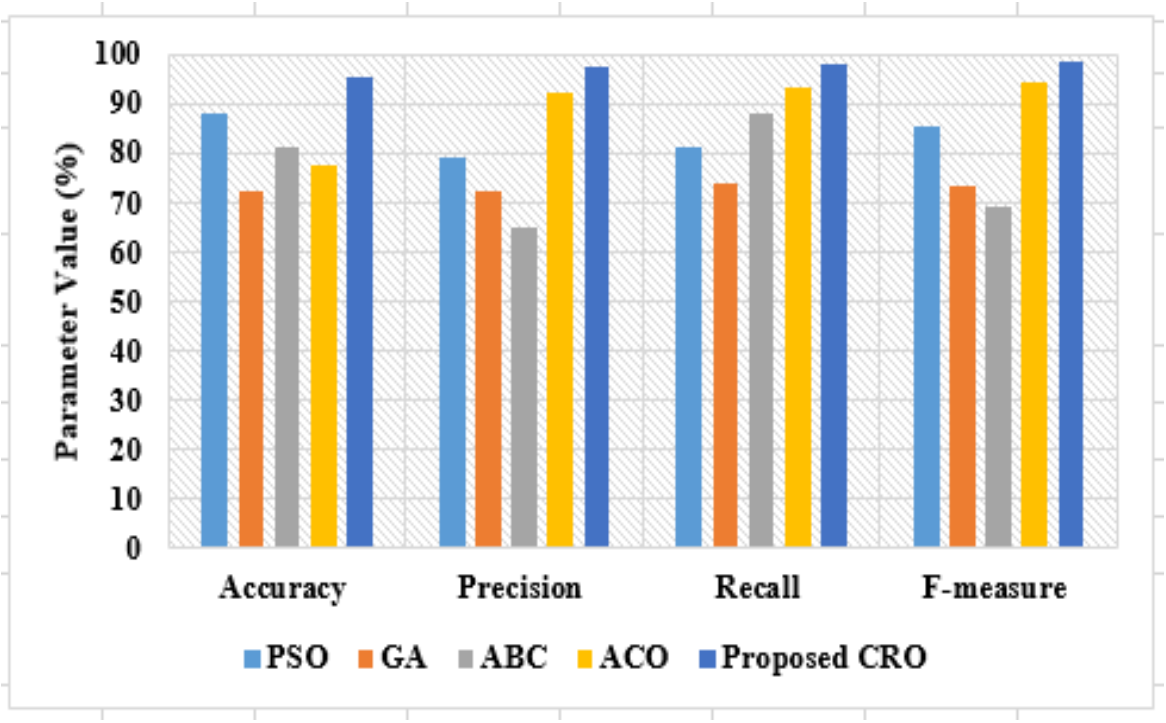


Figure 3: Performance Analysis of Proposed CRO for binary class

From the table and graph analysis, it is proved that proposed CRO achieved better performance than other techniques such as PSO, GA, ABC and ACO. For accuracy analysis, the GA and ACO achieved nearly 75%, PSO and ABC achieved nearly 83%, but the proposed CRO achieved 95.20% of accuracy. When comparing with all techniques, ABC achieved low precision (65.07%), where PSO and GA achieved nearly 75% of precision. However, ACO achieved 92.04% of precision and proposed CRO achieved 97.64% of precision. In addition, proposed CRO achieved 98.20% of recall and 98.67% of F-measure, where the existing techniques achieved nearly 80% to 93% of recall and F-measure. The reason for proposed CRO is that the fitness is determined. The next table 3 shows the performance of classifier for binary class.

DOI: 10.5281/zenodo.14600781

Table 3: Comparative Analysis of Classification Technique

Technique	Accuracy (%)
SVM	80.15
CNN layer	93.15
Bi-LSTM layer	93.87
Nonlinear layers	94.48
BERTbase fine-tuning	**95.20**

Here, the existing technique called SVM is tested on our dataset and the results are tabulated. From the table 3, it is clearly proved that our BERT technique achieved better classification accuracy. For instance, SVM achieved only 80.15% of accuracy, where CNN and Bi-LSTM layer achieved nearly 93.50% of accuracy. But, BERT base fine-tuning achieved 95.20% of accuracy, which proves that it classifies the binary class effectively than other layers and existing techniques. The next section will show the performance of proposed technique for multi-class classification using Twitter dataset.

Performance Analysis of Proposed Techniques for Multi Class

In this analysis, two types of experiments are carried out to test the performance of proposed CRO and BERT for multi class, i.e. racism, sexism, hate, or offensive content speech. Table 4 shows the comparative analysis of proposed CRO with existing meta-heuristic techniques in terms of accuracy, recall, precision and F-measure. Figure 4 shows the graphical representation of the comparative analysis.

Table 4: Comparative Analysis of Proposed Feature Selection Technique

Feature Selection Methodology	Parameter Evaluation			
	Accuracy (%)	Precision (%)	Recall (%)	F-measure (%)
PSO	68.73	80.56	81.32	65.54
GA	54.39	56.64	58.91	55.65
ABC	70.01	72.50	73.69	73.07
ACO	83.14	83.56	85.75	89.23
Proposed CRO	**86.90**	**88.24**	**88.47**	**89.20**

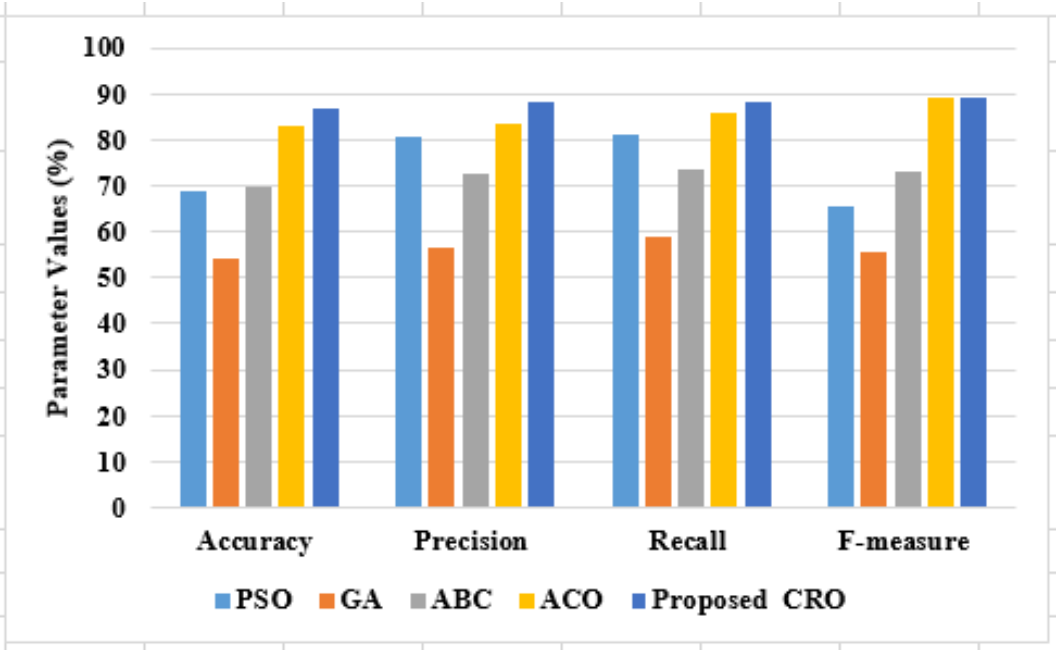


Figure 4: Performance Analysis of Proposed CRO for Multi-class

When comparing with all techniques, GA performs lowest performance in terms of all parameters, for instance, GA achieved nearly 55% to 58% of accuracy, precision, recall and f-measure. This is because, GA can't handle the multi-class data due to its fast convergence. ABC and PSO achieved nearly 70% of accuracy, 80% of precision and recall, nearly 70% of F-measure. However, ACO achieved nearly 81% to 89% of accuracy, precision, recall and F-measure. Even though, the proposed CRO achieved better performance (<95%) for binary class, the technique achieved only 86% of accuracy, 88% of precision and recall, 89.20% of F-measure. This shows that when tested with multi-class data, the

DOI: 10.5281/zenodo.14600781

proposed CRO achieved low performance but higher performance than existing techniques. Table 5 shows the comparative analysis of proposed BERT with SVM in terms of classification accuracy.

Table 5: Comparative Analysis of Classification Technique

Technique	Accuracy (%)
SVM	75.12
CNN layer	82.40
Bi-LSTM layer	83.64
Nonlinear layers	85.14
BERTbase fine-tuning	**86.90**

Here, the existing technique called SVM is tested on our dataset for multi-class data and the results are tabulated. From the table 5, it is clearly proved that our BERT technique achieved better classification accuracy. For instance, SVM achieved only 75.12% of accuracy, where CNN and Bi-LSTM layer achieved nearly 83% of accuracy. But, BERT base fine-tuning achieved 86.90% of accuracy, which proves that it classifies the multi class effectively than other layers and existing techniques. However, the classifiers shows low performance, when comparing with binary class classification. Figure 5 shows the comparative analysis of accuracy for both binary and multi-class classification.

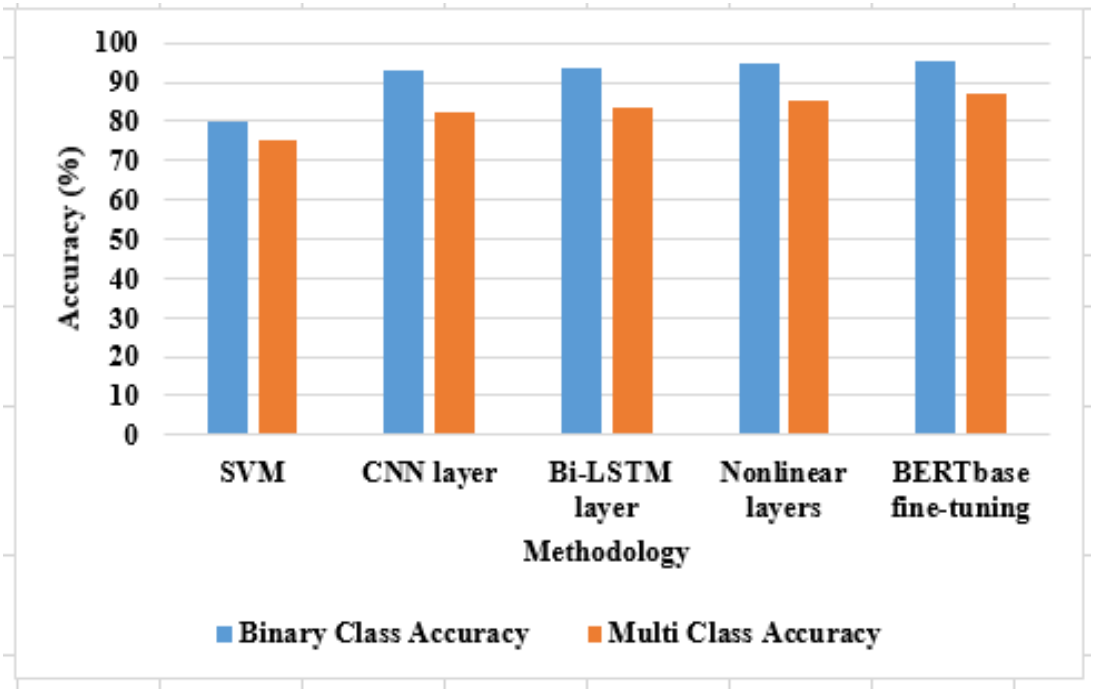


Figure 5: Comparative Analysis of Classifiers for Binary and Multi-class Data

Conclusion

Due to the confusion between hate speech and offensive or innocuous words, hate speech detection software flags user-generated content wrongly. Because of this, platforms and users may suffer significant repercussions, such as a decline in platform reputation or users abandonment. An improved hate speech detection system can be improved by using a novel CRO with transfer learning approach that takes advantage of the pre-trained language model BERT. Therefore, we present new fine-tuning procedures to assess the influence of different layers in BERT in hate speech detection task. A CNN-based technique for fine-tuning the BERT model has shown that our model outperforms prior works by exploiting syntactical and contextual information inherent in distinct transformer encoder layers. Moreover, our model's ability to detect biases in the process of gathering or annotating datasets is demonstrated by the results. Using the pre-trained BERT model to lessen bias in hate speech datasets in future studies could be a significant indication.

References

1. Saeed Z, Abbasi RA, Razzak I (2020) Evesense: what can you sense from twitter? Adv Inform Retr 12036:491.
2. Poletto, Fabio, Valerio Basile, Manuela Sanguinetti, Cristina Bosco, and Viviana Patti. "Resources and benchmark corpora for hate speech detection: a systematic review." *Language Resources and Evaluation* (2020): 1-47.
3. Z. Waseem and D. Hovy, “Hateful Symbols or Hateful People? Predictive Features for Hate Speech Detection on Twitter,” Proc. NAACL Student Res. Work., pp. 88–93, 2016.
4. H. Watanabe, M. Bouazizi, and T. Ohtsuki, “Hate Speech on Twitter: A Pragmatic Approach to Collect Hateful and Offensive Expressions and Perform Hate Speech Detection,” IEEE Access, vol. 6, pp. 13825–13835, 2018.
5. Al-Hassan, Areej, and Hmood Al-Dossari. "Detection of hate speech in social networks: a survey on multilingual

DOI: 10.5281/zenodo.14600781

corpus." In *6th International Conference on Computer Science and Information Technology*, vol. 10. 2019.
6. Chung, Y. L,. Kuzmenko, E., Tekiroglu, S. S., & Guerini, M. (2019). CONAN—COunter NArratives through Nichesourcing: A Multilingual Dataset of Responses to Fight Online Hate Speech. In Proceedings of the 57th annual meeting of the Association for Computational Linguistics, Association for Computational Linguistics (pp. 2819–2829).
7. Jurgens, D., Chandrasekharan, E., & Hemphill, L. (2019). A just and comprehensive strategy for using nlp to address online abuse. In Proceedings of the 57th annual meeting of the association for computational linguistics, Association for Computational Linguistics (ACL) (pp. 3658–3666).
8. Burnap, P. and M.L. Williams, Us and them: identifying cyber hate on Twitter across multiple protected characteristics. EPJ Data Science, 2016. 5(1): p. 11.
9. Gitari, N.D., et al., A lexicon-based approach for hate speech detection. International Journal of Multimedia and Ubiquitous Engineering, 2015. 10(4): p. 215-230.
10. Tulkens, S., et al., A dictionary-based approach to racism detection in dutch social media. arXiv preprint arXiv:1608.08738, 2016.
11. Köffer, S., et al., Discussing the value of automatic hate speech detection in online debates. Multikonferenz Wirtschaftsinformatik (MKWI 2018): Data Driven X-Turning Data in Value, Leuphana, Germany, 2018.
12. Hovy D, Waseem Z (2016) Hateful symbols or hateful people? predictive features for hate speech detection on twitter. In: Proceedings of the student research workshop, SRW@HLT-NAACL 2016, The 2016 conference of the north american chapter of the association for computational linguistics: human language technologies, San Diego California, USA 12-17, 2016, pp 88–93.
13. A. Assiri, A. Emam, and H. Al-Dossari, "Towards enhancement of a lexicon-based approach for Saudi dialect sentiment analysis," J. Inf. Sci., vol. 44, no. 2, pp. 184–202, 2018.
14. M. Wiegand, J. Ruppenhofer, A. Schmidt, and C. Greenberg, "Inducing a Lexicon of Abusive Words – a Feature-Based Approach," in Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers), 2018, pp. 1046–1056.
15. N. D. Gitari, Z. Zuping, H. Damien, and J. Long, "A lexicon-based approach for hate speech detection," Int. J. Multimed. Ubiquitous Eng., vol. 10, no. 4, pp. 215–230, 2015.
16. S. George K and S. Joseph, "Text Classification by Augmenting Bag of Words (BOW) Representation with Co-occurrence Feature," IOSR J. Comput. Eng., vol. 16, no. 1, pp. 34–38, 2014.
17. C.-F. Tsai, "Bag-of-Words Representation in Image Annotation: A Review," ISRN Artif. Intell., vol. 2012, pp. 1–19, 2012.
18. P. Burnap and M. L. Williams, "Us and them: identifying cyber hate on Twitter across multiple protected characteristics," EPJ Data Sci., vol. 5, no. 1, 2016.
19. G. Xiang, B. Fan, L. Wang, J. Hong, and C. Rose, "Detecting offensive tweets via topical feature discovery over a large scale twitter corpus," Proc. 21st ACM Int. Conf. Inf. Knowl. Manag. - CIKM '12, p. 1980, 2012.
20. J. Lilleberg, Y. Zhu, and Y. Zhang, "Support vector machines and Word2vec for text classification with semantic features," Proc. 2015 IEEE 14th Int. Conf. Cogn. Informatics Cogn. Comput. ICCI*CC 2015, pp. 136–140, 2015.

DOI: 10.5281/zenodo.14600781