Journal of Autonomous Intelligence (2024) Volume 7 Issue 3
doi: 10.32629/jai.v7i3.1122

ORIGINAL RESEARCH ARTICLE

CROA-based feature selection with BERT model for detecting the offensive speech in Twitter data

R. J. Anandhi[1], V. S. Anusuya Devi[1], B. S. Kiruthika Devi[2,*], Balasubramanian Prabhu kavin[3], Gan Hong Seng[4]

[1] *New Horizon College of Engineering, Ring Road, Bellandur Post, Bengaluru 560103, India*

[2] *School of Computing, Sathyabama Institute of Science and Technology, Jeppiaar Nagar, Rajiv Gandhi Salai, Chennai 600119, India*

[3] *Department of Data Science and Business Systems, SRM Institute of Science and Technology, Kattankulathur 603203, India*

[4] *School of AI and Advanced Computing, XJTLU Entrepreneur College (Taicang), Xi'an Jiaotong-Liverpool University, Suzhou 215400, China*

*** Corresponding author:** B. S. Kiruthika Devi, kiruthikadevi.b.s.cse@sathyabama.ac.in

ABSTRACT

Online hate speech has flourished on social networking sites due to the widespread availability of mobile computers and other Web knowledge. Extensive research has shown that online exposure to hate speech has real-world effects on marginalized communities. Research into methods of automatically identifying hate speech has garnered significant attention. Hate speech can affect any demographic, while some populations are more vulnerable than others. Relying solely on progressive learning is insufficient for achieving the goal of automatic hate speech identification. It need access to large amounts of labelled data to train a model. Inaccurate statistics on hate speech and preconceived notions have been the biggest obstacles in the field of hate speech research for a long time. This research provides a novel strategy for meeting these needs by combining a transfer-learning attitude-based BERT (Bidirectional Encoder Representations from Transformers) with a coral reef optimization-based approach (CROA). A feature selection (FC) optimization strategy for coral reefs, a coral reefs optimization method mimics coral behaviours for reef location and development. We might think of each potential answer to the problem as a coral trying to establish itself in the reefs. The results are refined at each stage by applying specialized operators from the coral reefs optimization algorithm. When everything is said and done, the optimal solution is chosen. We also use a cutting-edge fine-tuning method based on transfer learning to assess BERT's ability to recognize hostile contexts in social media communications. The paper evaluates the proposed approach using Twitter datasets tagged for racist, sexist, homophobic, or otherwise offensive content. The numbers show that our strategy achieves 5%–10% higher precision and recall compared to other approaches.

Keywords: natural language processing; bidirectional encoder representations from transformers; coral reefs optimization; hate speech detection; Twitter

ARTICLE INFO

Received: 8 August 2023
Accepted: 4 December 2023
Available online: 3 January 2024

1. Introduction

These days, you can find just about anything on a microblogging site. This is because microblogs allow users to share their opinions on any number of topics instantly, discuss urgent issues, offer constructive criticism, and boast about the benefits of products they regularly employ[1]. Businesses that provide comparable goods have started polling these microblogs to see how customers feel about their products/services. These companies regularly monitor user feedback and engage with customers via microblogs. One challenge is creating systems that can detect and summarise prevalent emotions. Opinion

Mining and Sentiment Analysis (OM & SA) methods have gained prominence in the arena of artificial intelligence, particularly in the area of (NLP). These strategies are typically used to gauge public opinion on a product or issue[2]. The abundance of user-generated content online, especially on social media sites[3], and the recent development of supervised learning technology have made the development of powerful and effective strategies possible. The global effort to tackle extremism, violence, false news, and other digital plagues has led to a surge in interest in NLP careers that focus on social and ethical issues in recent years. Hate speech is another example; it is a poisonous form of discourse that stems from intolerance and prejudice and can lead to physical harm, persecution, and even government regulation[4,5].

The material represents the opinions and feelings of its creators and readers on a certain topic or subject[6]. There are now countless ways to mine this data for insights into people's emotions, which can be used to better predict anything from product sales to stock market swings to the outcomes of upcoming elections. Twitter is a popular microblogging platform with around 300 million active users[7]. Due to its significance in understanding people's feelings and opinions, Twitter Sentiment Analysis (TSA) has garnered great interest in recent years[8]. The topic of SA has been the deal of research, and in recent years, the topic of TSA has received a lot of attention. Some social media users hide their identities and engage in attacks or abuse of others[9] due to the absence of online speech regulation. When people make derogatory comments about others, it's called "hate speech" because they mean to hurt their feelings. No single definition of "hate speech" has gained widespread acceptance. Many definitions of hate speech have been proposed[10], including those that relate it to specific acts of violence or to the creation of a hostile environment that may incite such an act.

Tracking the content of media sites is impossible due to the high volume of posts made every second. Therefore, social media companies have a hard time finding a happy medium between preventing offensive speech and protecting users' right to free expression[11]. The world's variety of people, nations, cultures, and beliefs may also give rise to hate speech. But the concept and features of cyber-hatred vary greatly depending on the culture. It is believed that each culture reacts and intervenes in its own unique way[12]. This unruly behaviour has increased over the past decade, and it is labour-intensive to identify and remove harmful messages and comments from social media platforms by hand[13]. Quickly classifying hate speech and shielding internet users from social hate speech abuse is possible using automated techniques[14]. Over the past ten years, scientists have investigated the feasibility of using automatic algorithms to identify hate speech. Even it is still challenging to identify some hate phrases. Twitter's difficult-to-annotate vocabulary and emotions provide a problem for automatic hate speech recognition[15]. Overused word abbreviations and idioms, such as character redundancy (e.g., kind, caaaaar), unnecessary tone, and excessive exclamation (e.g., Coming...!!!!, yes???). Therefore, the source material requires a significant preprocessing method that is consistent with other similar postings in order to generate a format that keeps the original meaning[16]. Since it is from other potentially destructive material that does not meet the hate tweet category[17], it is problematic for both robots and humans to recognize hate speech from tweets. Word, deep learning, Term Frequency-Inverse Document, and character n-grams are among the common machine learning algorithms and surface characteristics that have been used in other research to identify hate speech.

This is difficult work since "hate speech" can be interpreted in a diversity of ways. Therefore, some people may find certain elements offensive while others do not. Given the challenges of manually removing hate speech data and the widespread nature of such data, automatic hate speech detection methods are deemed necessary. It proposes utilizing the Coral Reef algorithm and BERT to monitor social media for instances of hate speech and to locate at-risk populations. As can be shown in the next sections, the projected method is tested on the Twitter dataset. This is the style adopted for the paper: The second section analyses related projects concerning the classification of hate. In Section 3, it features the proposed workflow. Validation of the suggested method in light of current approaches is discussed in Section 4. In the last section, it discusses what this effort amounts to and where it wants to go from here.

2. Related works

To that aim, Almaliki et al.[18] present a strategy for detecting hate speech in Arabic across social media platforms like Twitter. Therefore, this research presents the Arabic BERT-Mini Model (ABMM) to identify online hate speech. The bidirectional encoder representations from transformers (BERT) model was utilized to classify Twitter data into three distinct categories: normal, abuse, and hate speech. It conducted a series of experiments using Twitter data to evaluate our model against state-of-the-art approaches.

Along with a comprehensive analysis of previous efforts and plans for recognizing and censoring hostile content on Twitter, del Valle-Cano et al.[19] provide a novel classification approach based on individuals' profiles. This work makes three contributions. Our work on the publicly available dataset from HaterNet using a BERT-inspired model called HaterBERT shows promising results towards this end; (ii) A method for building a user database in the form of a relational network, with the aim of deducing textual and centrality characteristics. Furthermore, (ii) a third model, SocialHaterBERT, which combines the first two approaches by analyzing features outside of those inherent in the text, has been independently tested with a number of standard Machine Learning and Deep Learning procedures, demonstrating its efficacy in identifying trolls. Experiment results demonstrate that this last input considerably improves the outcomes, paving the way for future research into connected models and establishing a new field within the bounds of texts.

Bilal et al.[20] employed a transformer-based model to classify hate speech in Roman Urdu because of the model's ability to capture text context. The first pre-trained BERT model in Roman Urdu, BERT-RU, was also developed by us. It retrained BERT from scratch using the largest Roman Urdu dataset available, which had 173,714 text messages, to improve hate speech classification. LSTM, Attention Layer, and CNN were used as baselines. It also investigated transfer learning by utilizing deep learning models and pre-trained BERT embeddings. The model improved in all four metrics (accuracy, precision, recall, and F-measure) by a greater margin than before. The transformer-based model also shows better generalization on a cross-domain dataset.

Puteri et al.[21] developed a method to recognize hate speech using a CNN model trained with FastText word embedding. The performance of a CNN classification model with FastText as the word embedding yields good results for recognizing hate speech when the K-Fold Cross Validation procedure is used with an appropriate dropout value. The accuracy value achieved can be used to evaluate a model's effectiveness in stopping the spread of hate speech via social media.

Addressing recent instances of anti-immigrant hate speech and Castillo-López et al.[22] describe the findings of a comprehensive examination of the presentation of hate speech detection algorithms on various Spanish dialects. It constructed a Twitter data collection that includes these nuances. Our results using mBERT and Beto, a language perfect, as the foundation of our transfer learning architecture show that models for detecting hate speech in a specific variety of Spanish are negatively impacted when other varieties of the language are ignored. Even within a single language's spoken territory, there may be dialectal differences in the expression of hate.

To efficiently execute hate speech recognition in low-resource tongues, Awal et al.[23] present HateMAML, a model-agnostic meta-learning (MAML)-based system. With a self-supervision approach, HateMAML is able to create a barrier and generate superior LM initialization for rapid adaptation. Extensive tests are performed on eight different low-resource languages and five different datasets. According to the findings, HateMAML achieves better outcomes than the multilingual transfer situation by a margin of more than 3%. In addition, we do ablation research to better understand HateMAML's peculiarities.

To divide online comments into hate speech and non-hate speech, Dwivedy and Roy[24] describe a multimodal architecture made up of a combined Long Short-Term Memory (LSTM)-based model. In order to predict whether or not a post contains hate speech, the suggested model takes into account both the text and the accompanying pictures. To determine if a social media post contains hate speech or not, the architecture combines visual and text features to determine if a social media post contains hate speech or not. The results of comparing the base model to those generated by separate text and picture models showed that the combined model produced more accurate predictions.

Based on Ali et al.'s[25] creation of an Urdu language hate lexicon, we generate an annotated dataset to conduct studies using many speech identification as a baseline. And since we already have pre-trained BERT embeddings, we can apply transfer learning to make the most of them for our work. Finally, we test four distinct BERT models and compare them to several other models to evaluate their performance.

Khan et al.[26] provide a new deep learning model, HCovBi-Caps, built on Convolutional, BiGRU, and Capsule networks to categorize hate speech. When tested on two suggested models achieves the best results on the latter. The suggested model achieves an accuracy of 0.93 during training and 0.90 during validation on the imbalanced dataset. When compared with other methods results on the imbalanced dataset, HCovBi-Caps stands out as the clear winner. We also examine how varying hyperparameters affect HCovBi-Caps' performance to learn more about how to optimize their settings. We observed that increasing the sum of routing iterations negatively impacts the model's performance, while increasing the capsule dimension has the opposite effect.

3. Proposed system

In this section, an explanation of the proposed procedure is given. There are five major parts such as dataset description, preprocessing, feature extraction, and classification.

3.1. Dataset description

136,052 tweets were composed during a two-month period, and 16,914 were tagged[27]. 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. **Table 1** provides the list of the procedures used in the preprocessing phase.

3.2. Preprocessing

Table 1. Preprocessing procedures.

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

3.2.1. 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 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 preferences, and hashtag symbols from our research.

3.2.2. Abbreviations and slang

Earlier, I said that Twitter's character limit forces users to utilize alternative acronyms and jargon. If each user manner and utilizes diverse acronyms, this becomes much more challenging. These are abbreviations and phrases that are occasionally 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.

3.2.3. Expanding contractions

As a preprocessing 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.

3.2.4. Removing numbers

However, numerals do not always deliver useful information for text organization, 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.

3.2.5. 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 info from our trials.

3.2.6. Lemmatization

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

3.2.7. Removing punctuation

Eliminating punctuation is a traditional and widespread preprocessing 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.

3.2.8. 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.

3.2.9. 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 similar meaning.

3.2.10. Removing stop words

Natural 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 preprocessing ordering tasks, removing stop words is a frequent method. Stop words were detached from the corpus using the NLTK Stop-word library16.

3.2.11. Elongated characters

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

3.2.12. Incorrect spelling

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

3.3. Feature extraction

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

3.3.1. Dictionaries and lexicons

In unsupervised machine learning scenarios, the use of dictionaries and lexicons is commonly employed[28]. Wiegand et al.[29] 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.[30] to aggregate opinions and rate subjective words.

3.3.2. Bag-of-words (BOW) and *n*-grams

It can be viewed as a feature of word cooccurrence. A vectorization procedure is performed on tokenized terms in the corpus by considering their incidence in tweets, using techniques such as TF-IDF weighting. An alphabetical list will be created and offered as vectors of weights[31]. In *n*-gram representation, where N represents the number of words arranged in a row. For hate speech findings, the study looked into the impact of combining a number of variables in N-grams[32]. A character n-gram representation has been found to be a powerful tool for detecting hate speech. This is because BOW has limitations and needs to be complemented by other characteristics 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[33].

3.3.3. Latent dirichlet allocation (LDA)

A modelling technique, it is based on probabilities. These latent topics will serve as features as an alternative to words. For unsupervised and semi-supervised machine learning environments, LDA is a viable option. When it comes to abusive text recognition on Twitter, Xiang et al.[34] 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 the LDA model.

3.3.4. Word embedding and Word2Vec

While word embedding addresses data sparsity, it adds a semantic element by creating distributed representations that establish connections between words. To construct word embedding, Word2Vec is a technique. Lilleberg et al.[35] report that researchers in the field of text mining are particularly interested in Word2Vec due to its compatibility with various machine-learning models.

3.4. Feature selection using coral reefs optimization method

The “Coral Reefs Optimisation” (CRO) method is not associated with the deterioration of coral reefs; rather, it is an optimisation technique inspired by nature. The Coral Reefs Optimisation algorithm was created by a team of researchers who were motivated by the ecological relationships seen in coral reefs. It is utilised in the field of computer science and mathematics to solve optimisation problems. This meta-heuristic program replicates coral growth in reefs. As its corals, this method starts with a population of 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 arbitrarily positioned in the grid’s cells at random. A few cells in the square grid should be left empty to allow new corals to grow in the following feature selection 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 improve them. Optimizing for health might have 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 a condition or problem) or its ability to identify an empty spot. As a result, the operators’ solutions are applied to the reef. First, for each new solution, the health function value is strongminded.

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 engaged, and the new solution’s larger than the old solution’s health function, the new solution will cell if it is not sterile. For each solution, a maximum number of placement tries is defined, and if the maximum number of if the maximum number of attempts 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 preceding 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 coral depredation process, that incorrect solutions are eliminated or excluded in order to guarantee that there is sufficient space on the reef for the subsequent algorithmic step. 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 started the procedure termination disorder. As the algorithm progresses, a health function determines whether or not the solutions are fit. The following are the steps of the CRO technique[36].

3.4.1. Initialization

It starts with a matrix R that has ($N \times M$). 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 matrix in the next steps. When each random solution has been created and placed in the matrix, a ratio of unfilled cells to engaged cells is determined. As soon as this ratio falls below 0.4, initial population is complete. After each solution is generated, it is added to a list and given an identifier. This identification is inserted in a matrix cell. Moreover, after making any solution, the procedure calculates the worth of its health function.

3.4.2. External reproduction operator

To solve the problems, this operator is functional 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 of 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 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, "Ocean" usually refers to the search space or solution space where possible solutions are explored in order to resolve an optimisation issue. they are placed on a list of fresh solutions so that they can subsequently be placed on the reef.

3.4.3. Internal reproduction operator

A solutions list that wasn'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 key is upturned for this drive, and a new solution will be formed as a result.

3.4.4. Setting new solutions

Here, each solution tries to fit into one of the cells in the matrix. This is done by calculating each answer value of the health function and then examining how likely each answer is to be positioned in a cell of the matrix. A new solution 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 the empty population. It's possible to replace the old solution in a cell if the value of the function is greater than that of its predecessor's health function. That cell will not be used if a new solution is not selected. In the matrix, each key can attempt to be inserted h times, where h is a user parameter, the term "key" indicates an essential component or characteristic influencing the way potential solutions based on algorithms function.

3.4.5. Asexual replica operator

This operator uses some of the reef's best keys. Here, the matrix keys are arranged by their health function values. This is followed by a random selection from the start 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. Next, attempt to insert the new solutions, identical to what was done before, in a matrix similar to the prior stage.

3.4.6. 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 the worst solutions from it. Fd is also a limit 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 likelihood of depredation is specified by the Pd parameter. After a few repetitions, the stable large number of vacant cells in the matrix allow new solutions to be put and grown. A larger amount of weak solutions must be eliminated after frequent iterations. 0.1/k is extra 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. The final solution of the CRO is determined by the best matrices.

3.5. Classification using BERT

In this study, we investigate the performance of the BERT transformer perfectly in identifying hate speech. BERT comes in two sizes: BERT base and BERT large. BERT base is a multi-layer neural network that was trained on Wikipedia, which contains 2500 M and 800 M tokens, respectively. In comparison to BERT large, BERT base has a 24-layer encoder, 340 million parameters, and 16 attention heads. According to their retrieved embeddings, BERTbase embeddings contain 768 hidden sizes. In order to identify hate speech, we must first examine the contextual information produced from layers and then fine tune it using BERT with annotated datasets. It accepts a 512-token sequence as input and returns a 768-dimensional sequence. BERT inserts [CLS] and [SEP] segments into each input sequence. In hate speech ordering tasks, we employ a special classification embedding in [CLS] as a representation of the entire sequence; we use [CLS] embedding as the first token in the final hidden layer. The [SEP] is not used as a segment separator in our classification task. Each tweet in our datasets is classified using the BERTbase model. As a result, our focus is on fine-tuning the BERTbase settings that have been pre-trained. This entails fine-tuning a classifier with layers of 768 scopes on top of the BERTbase transformer to minimize task-specific factors.

3.5.1. Fine-tuning strategies

It is possible to capture multiple degrees of syntactic and semantic info 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. The pre-trained parameters that capture broad syntactic and semantic information are referred to as the “general layer”. The “task-specific layer” is where specific parameters, like learning rates, are adjusted to fine-tune the model for a BERT classification task so that it can adjust to subtleties in the labelled datasets. As a result of our classification work, four different fine-tuning procedures have been established, utilizing pre-trained BERT base 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, for fine-tuning. As shown in **Figure 1**, where Xi is represented as the vector illustration of token I in the tweet sample, diverse fine-tuning attitudes for the hate speech identification job are described as follows:

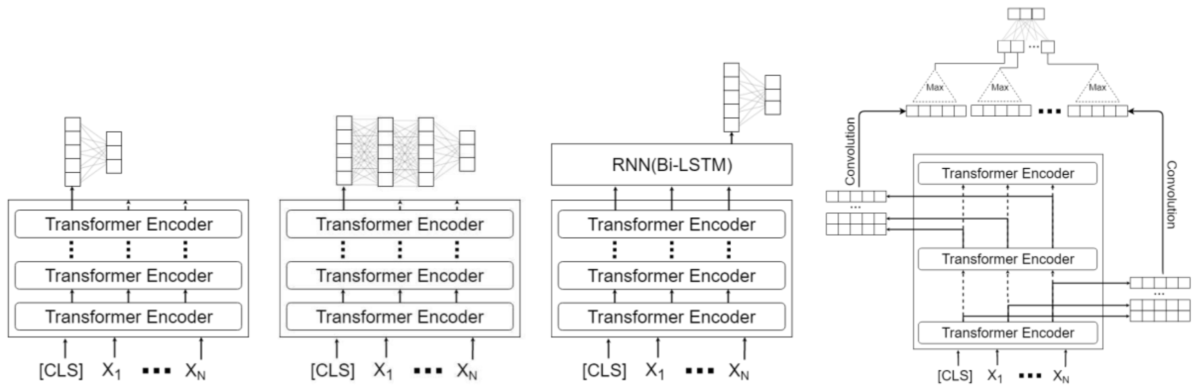

Figure 1. Fine-tuning plans. **(a)** BERTbase fine-tuning; **(b)** nonlinear sheets; **(c)** Bi-LSTM layer; **(d)** CNN coating.

3.5.2. BERT based fine-tuning

When using the first strategy, which is seen in **Figure 1a**, 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] encoder.

3.5.3. Insert nonlinear layers

As an alternative to using a fully linked network, a system with two layers of size 768 is used. A fully linked network is a neural network that is fully connected, meaning that all nodes in one layer are connected to all nodes in the layer below it, resulting in dense connections across the network. However, as shown in **Figure 1b**, the last layer, which is the initial construction, employs the softmax activation function.

3.5.4. Insert Bi-LSTM layer

Instead of only [CLS], a Bi-LSTM receives all yields of the newest transformer encoder as shown in **Figure 1c**. The last layer or representation produced by the most recent transformer encoder in the model architecture is referred to as the "newest transformer encoder". It sends the concealed state to a fully connected network, which classifies the input using a function.

3.5.5. Insert CNN layer

Instead of using the yield of the most recent encoder, this construction (shown in **Figure 1d**) uses the yields of altogether encoders. The term "altogether encoders" refers to the combined outputs of multiple encoders in the technique. 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 the encoder by smearing output. This creates a vector that can be fed into a fully connected network. The categorization operation is approved by smearing softmax to the input.

4. Results and discussion

This was done using a PC with an Intel 2410 M 2.3 GHz processor, 4 GB of DDR3 RAM, anda 600 GB SATA hard drive. Evaluations do not take into account limitations (such as memory constraints).

4.1. Performance metrics

The proposed model is evaluated by using following Equations (1)–(4), which is mentioned as follows:

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

$$Precision = \frac{|TP|}{|TP| + |FP|} \tag{2}$$

$$Recall = \frac{|TP|}{|TP| + |FN|} \tag{3}$$

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

4.2. Performance analysis of proposed technique for Binary classification

In this analysis, two types of trials are carried out to test the performance of proposed CRO and BERT for binary classification, i.e., Normal speech and Hate speech. **Table 2** expresses the comparative analysis of projected CRO with existing meta-heuristic techniques in terms of accuracy and F1-Score. **Figure 2** demonstrates the comparative analysis of feature selection process. **Figure 3** demonstrates the graph of the comparative analysis.

Table 2. Comparative investigation of projected feature selection process.

Feature Selection Methodology	Parameter Evaluation			
	Accuracy (%)	**Precision (%)**	**Recall (%)**	**F-measure (%)**
PSO	80.89	79.12	80.92	85.27
GA	82.30	72.50	73.69	73.07
ABC	85.25	65.07	88.06	69.28
ACO	87.26	92.04	93.17	94.08
Proposed CRO	96.10	97.64	98.20	98.67

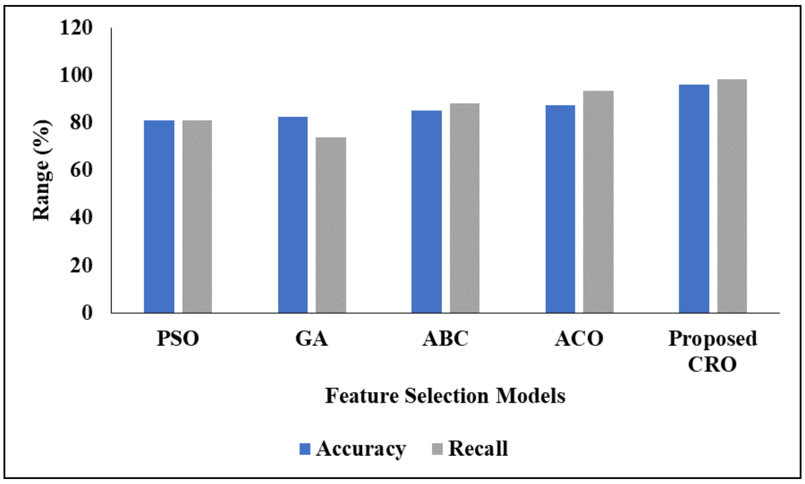

Figure 2. Performance analysis of proposed CRO for binary class.

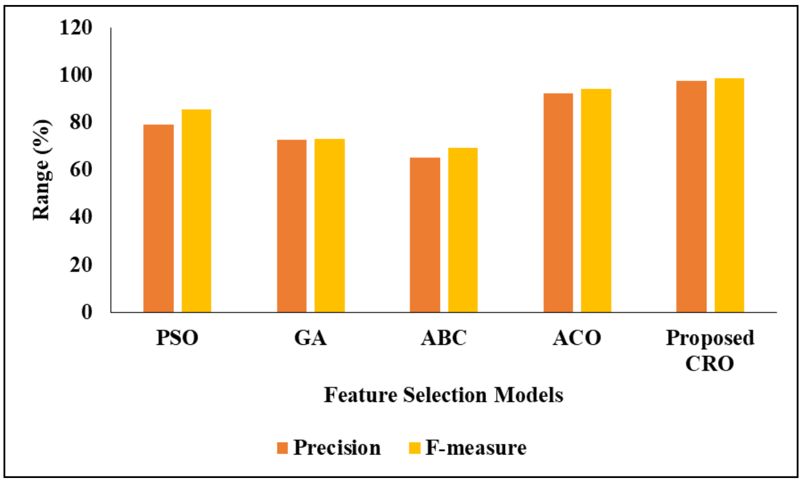

Figure 3. Graphical description of the projected model in terms of numerous metrics.

From the table and graph analysis, it is demonstrated that the proposed CRO achieved better performance than other techniques such as PSO, GA, ABC and ACO. For accuracy analysis, the GA and ACO achieved nearly 80%, PSO and ABC achieved nearly 85%–87%, but the proposed CRO achieved 96.10% of accuracy. When compared with all techniques, ABC achieved low precision (65.07%), whereas PSO and GA achieved nearly 75% of precision. However, ACO achieved 92.04% of precision, and the proposed CRO achieved 97.64% of precision as the proposed model provides higher precision value in speech detection. In addition, the proposed CRO achieved 98.20%, whereas the existing techniques achieved nearly 80% to 93% of recall and F1-Score. The reason for the proposed CRO is that the fitness is determined. The next **Table 3** shows the presentation of the classifier for binary class.

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	96.10

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

4.3. Performance analysis of proposed techniques for multi-class

In this analysis, two types of trials are carried out to test the performance of proposed CRO and BERT for multi-classification, i.e., racism, sexism, hate, or offensive content speech. **Table 4** demonstrates the appropriate study of the proposed CRO with existing meta-heuristic procedures. **Figure 4** shows the graph of the comparative analysis. **Figure 5** shows the graph of the comparative analysis of FS models.

Table 4. Comparative investigation of projected feature selection practice.

Feature Selection Methodology	Parameter Evaluation			
	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
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	89.90	88.24	88.47	89.20

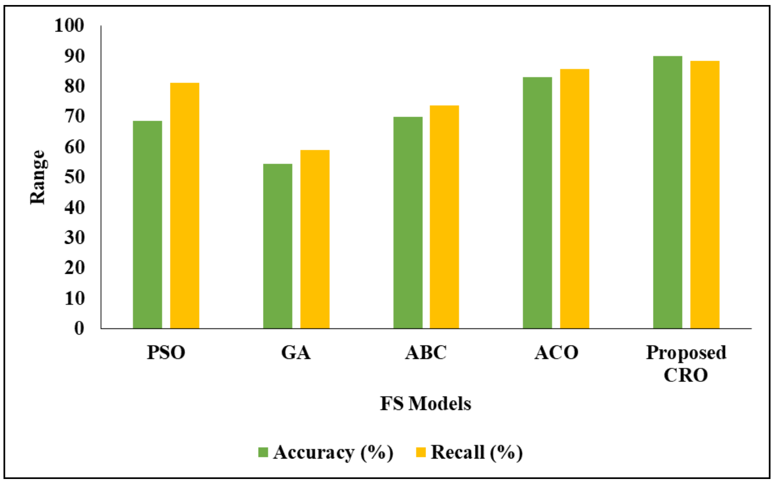

Figure 4. Performance analysis of proposed CRO for multi-class.

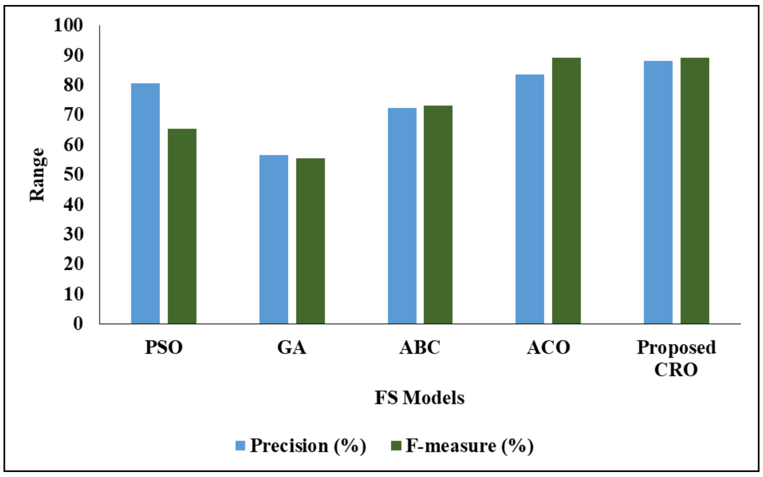

Figure 5. Comparative analysis of various FS models.

When associating with all techniques, GA performs the lowest performance in terms of all parameters; for instance, GA achieved nearly 55% to 58% of very less accuracy and F1-Score. This is because GA can't handle the multi-class data due to its fast convergence. ABC and PSO accomplished nearly 70% of accuracy, 80% of precision and recall, and nearly 70% of F1-Score. However, ACO achieved nearly 81% to 89% of accuracy. Even though the proposed CRO achieved better performance (<95%) for binary class, the technique achieved only 86% of accuracy, 88.24% of precision, 88.47% of recall, and 89.20% of F1-Score. This shows that when tested with multi-class data, the proposed CRO achieved low performance but higher performance than existing techniques. **Table 5** shows the relative analysis of the 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	89.90

Here, the existing technique called SVM is tested on our dataset for multi-class data, and the results are presented. 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, whereas CNN and Bi-LSTM layer achieved nearly 83% of accuracy. But, BERT based fine-tuning achieved 86.90% of accuracy, which proves that it classifies the multi-class more effectively than other layers and existing techniques. However, the classifiers show low performance when compared with other models. **Figure 6** shows the comparative analysis of accuracy for both classifications.

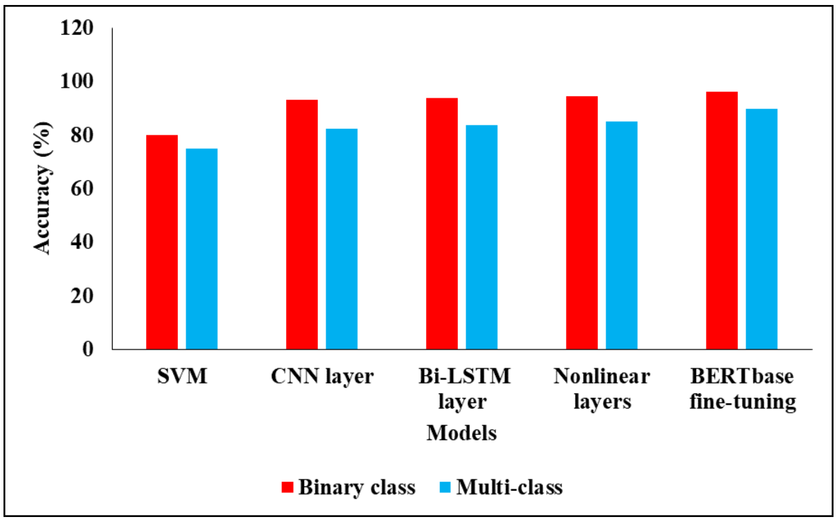

Figure 6. Comparative analysis of classifiers for binary and multi-class data.

5. Conclusion

Due to the confusion between hate speech and offensive or innocuous words, hate speech detection software flags user-generated content incorrectly. Because of this, platforms and users may suffer significant consequences, such as a decline in platform reputation or user abandonment. A unique CRO with a transfer learning approach that takes advantage of the pre-trained language model BERT can be used to improve hate speech detection systems. Therefore, it present novel fine-tuning methods for assessing the effect of several BERT layers on the hate speech finding task. It found that by using information stored in several transformer encoder layers, our model outperformed in accuracy prior attempts using a CNN-based approach for fine-tuning the BERT model. The findings also show that our algorithm can identify biases introduced during data collection or annotation. The pre-trained BERT model's potential to minimize bias in datasets suggests it may be useful in future research.

Author contributions

Conceptualization, RJA and VSAD; methodology, BSKD; software, BSKD; validation, BPK and GHS; formal analysis, RJA; investigation, VSAD; resources, BSKD; data curation, BSKD; writing—original draft preparation, RJA and VSAD; writing—review and editing, BSKD; visualization, BPK; supervision, GHS; project administration, BPK and GHS; funding acquisition, GHS. All authors have read and agreed to the published version of the manuscript.

Conflict of interest

The authors declare no conflict of interest.

References

1. Saeed Z, Ayaz Abbasi R, Razzak I. EveSense: What Can You Sense from Twitter? Advances in Information Retrieval. 2020, 491-495. doi: 10.1007/978-3-030-45442-5_64
2. Poletto F, Basile V, Sanguinetti M, et al. Resources and benchmark corpora for hate speech detection: a systematic review. Language Resources and Evaluation. 2020, 55(2): 477-523. doi: 10.1007/s10579-020-09502-8
3. Waseem Z, Hovy D. Hateful Symbols or Hateful People? Predictive Features for Hate Speech Detection on Twitter. Proceedings of the NAACL Student Research Workshop. 2016. doi: 10.18653/v1/n16-2013
4. Watanabe H, Bouazizi M, Ohtsuki T. Hate Speech on Twitter: A Pragmatic Approach to Collect Hateful and Offensive Expressions and Perform Hate Speech Detection. IEEE Access. 2018, 6: 13825-13835. doi: 10.1109/access.2018.2806394
5. Al-Hassan A, Al-Dossari H. Detection of hate speech in social networks: A survey on multilingual corpus. Computer Science & Information Technology (CS & IT). 2019. doi: 10.5121/csit.2019.90208

6. Chung YL, Kuzmenko E, Tekiroglu SS, et al. CONAN-COunter NArratives through Nichesourcing: a Multilingual Dataset of Responses to Fight Online Hate Speech. Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics. 2019. doi: 10.18653/v1/p19-1271
7. Jurgens D, Hemphill L, Chandrasekharan E. A Just and Comprehensive Strategy for Using NLP to Address Online Abuse. Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics. 2019. doi: 10.18653/v1/p19-1357
8. Burnap P, Williams ML. Us and them: identifying cyber hate on Twitter across multiple protected characteristics. EPJ Data Science. 2016, 5(1). doi: 10.1140/epjds/s13688-016-0072-6
9. Gitari ND, Zhang Z, Damien H, et al. A Lexicon-based Approach for Hate Speech Detection. International Journal of Multimedia and Ubiquitous Engineering. 2015, 10(4): 215-230. doi: 10.14257/ijmue.2015.10.4.21
10. Tulkens S, Hilte L, Lodewyckx E, et al. A dictionary-based approach to racism detection in Dutch social media. arXiv. 2016, arXiv:1608.08738. doi: 10.48550/arXiv.1608.08738
11. Köffer S, Riehle DM, Höhenberger S, Becker J. Discussing the value of automatic hate speech detection in online debates. Multikonferenz Wirtschaftsinformatik (MKWI 2018): Data Driven X-Turning Data in Value; Leuphana, Germany.
12. Akuma S, Lubem T, Adom IT. Comparing Bag of Words and TF-IDF with different models for hate speech detection from live tweets. International Journal of Information Technology. 2022, 14(7): 3629-3635. doi: 10.1007/s41870-022-01096-4
13. William P, Gade R, Chaudhari R esh, et al. Machine Learning based Automatic Hate Speech Recognition System. 2022 International Conference on Sustainable Computing and Data Communication Systems (ICSCDS). 2022. doi: 10.1109/icscds53736.2022.9760959
14. Malik JS, Pang G, Hengel AVD. Deep learning for hate speech detection: a comparative study. arXiv. 2022, arXiv:2202.09517.
15. Turki T, Roy SS. Novel Hate Speech Detection Using Word Cloud Visualization and Ensemble Learning Coupled with Count Vectorizer. Applied Sciences. 2022, 12(13): 6611. doi: 10.3390/app12136611
16. Khan S, Fazil M, Sejwal VK, et al. BiCHAT: BiLSTM with deep CNN and hierarchical attention for hate speech detection. Journal of King Saud University-Computer and Information Sciences. 2022, 34(7): 4335-4344. doi: 10.1016/j.jksuci.2022.05.006
17. Patil H, Velankar A, Joshi R. L3cube-mahahate: A tweet-based marathi hate speech detection dataset and bert models. Proceedings of the Third Workshop on Threat, Aggression and Cyberbullying (TRAC 2022). 2022, 1-9.
18. Almaliki M, Almars AM, Gad I, et al. ABMM: Arabic BERT-Mini Model for Hate-Speech Detection on Social Media. Electronics. 2023, 12(4): 1048. doi: 10.3390/electronics12041048
19. del Valle-Cano G, Quijano-Sánchez L, Liberatore F, et al. SocialHaterBERT: A dichotomous approach for automatically detecting hate speech on Twitter through textual analysis and user profiles. Expert Systems with Applications. 2023, 216: 119446. doi: 10.1016/j.eswa.2022.119446
20. Bilal M, Khan A, Jan S, et al. Roman Urdu Hate Speech Detection Using Transformer-Based Model for Cyber Security Applications. Sensors. 2023, 23(8): 3909. doi: 10.3390/s23083909
21. Puteri FN, Sibaroni Y, Fitriyani F. Hate Speech Detection in Indonesia Twitter Comments Using Convolutional Neural Network (CNN) and FastText Word Embedding. Jurnal Media Informatika Budidarma. 2023, 7(3): 1154-1161.
22. Castillo-lópez G, Riabi A, Seddah D. Analyzing Zero-Shot transfer Scenarios across Spanish variants for Hate Speech Detection. Tenth Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial 2023). 2023. doi: 10.18653/v1/2023.vardial-1.1
23. Awal MR, Lee RKW, Tanwar E, et al. Model-Agnostic Meta-Learning for Multilingual Hate Speech Detection. IEEE Transactions on Computational Social Systems. 2023, 1-10. doi: 10.1109/tcss.2023.3252401
24. Dwivedy V, Roy PK. Deep feature fusion for hate speech detection: a transfer learning approach. Multimedia Tools and Applications. 2023, 82(23): 36279-36301. doi: 10.1007/s11042-023-14850-y
25. Ali R, Farooq U, Arshad U, et al. Hate speech detection on Twitter using transfer learning. Computer Speech & Language. 2022, 74: 101365. doi: 10.1016/j.csl.2022.101365
26. Khan S, Kamal A, Fazil M, et al. HCovBi-Caps: Hate Speech Detection Using Convolutional and Bi-Directional Gated Recurrent Unit With Capsule Network. IEEE Access. 2022, 10: 7881-7894. doi: 10.1109/access.2022.3143799
27. Hovy D, Waseem Z. 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; 2016; San Diego California, USA. pp. 88-93.
28. Assiri A, Emam A, Al-Dossari H. Towards enhancement of a lexicon-based approach for Saudi dialect sentiment analysis. Journal of Information Science. 2017, 44(2): 184-202. doi: 10.1177/0165551516688143
29. Wiegand M, Ruppenhofer J, Schmidt A, et al. Inducing a Lexicon of Abusive Words—a Feature-Based Approach. Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers). 2018. doi: 10.18653/v1/n18-1095

30. Gitari ND, Zhang Z, Damien H, et al. Incorporating lexical knowledge via WordNet to latent dirichlet allocation in offensive message detection. Journal of Computational and Theoretical Nanoscience. 2016, 13(5): 215-230. doi: 10.1166/jctn.2016.5243
31. George KS, Joseph S. Text Classification by Augmenting Bag of Words (BOW) Representation with Co-occurrence Feature. IOSR Journal of Computer Engineering. 2014, 16(1): 34-38. doi: 10.9790/0661-16153438
32. Tsai CF. Bag-of-Words Representation in Image Annotation: A Review. ISRN Artificial Intelligence. 2012, 2012: 1-19. doi: 10.5402/2012/376804
33. Burnap P, Williams ML. Us and them: identifying cyber hate on Twitter across multiple protected characteristics. EPJ Data Sci. 2016, 5(1).
34. Xiang G, Fan B, Wang L, et al. Detecting offensive tweets via topical feature discovery over a large scale twitter corpus. Proceedings of the 21st ACM international conference on Information and knowledge management. 2012. doi: 10.1145/2396761.2398556
35. Lilleberg J, Zhu Y, Zhang Y. Support vector machines and Word2vec for text classification with semantic features. 2015 IEEE 14th International Conference on Cognitive Informatics & Cognitive Computing (ICCI*CC). 2015. doi: 10.1109/icci-cc.2015.7259377
36. Venkatasubramanian S, Suhasini A, Vennila C. Cluster Head Selection and Optimal Multipath detection using Coral Reef Optimization in MANET Environment. International Journal of Computer Network and Information Security. 2022, 14(3): 88-99. doi: 10.5815/ijcnis.2022.03.07