A DATA MINING APPROACH FOR FILTERING OUT SOCIAL SPAMMERS IN LARGE-SCALE TWITTER DATA COLLECTIONS

Waheeda Almayyan and Asaad Alzayed

Computer Information Department, Collage of Business Studies, PAAET, Kuwait

ABSTRACT

Social networking services – such as Facebook.com and Twitter.com – are fast-growing enterprise platform that has become a prevalent and essential component of daily life. Due to its popularity, Twitter draws many spammers or other fake accounts to post malicious links and infiltrate legitimate users' accounts with many spam messages. Therefore, it is crucial to recognize and screen spam tweets and spam accounts. As a result, spam detection is highly needed but still a difficult challenge. This article applied several Bio-inspired optimization algorithms to reduce the features' dimensions in the first stage. Then we used several classification schemes in the second stage to enhance the spam detection rate in three real Twitter data collections. The performance of the chosen classifiers also revealed that Random Forest and C4.5 classifiers achieved the highest Accuracy, Precision, Recall, and F1-score even on class imbalance.

KEYWORDS

Twitter, spam, machine learning, classification, PSO, Cuckoo, Bat.

1. INTRODUCTION

Nowadays, online social networks have become increasingly crucial in disseminating information worldwide. Users of social networking sites share their thoughts and engage in discussions about many issues. Users typically join social networks to keep in touch with friends and family or coworkers through various means, including direct messaging, publishing and sharing thoughts, ideas, and other media, such as images and videos, and receiving and disseminating information [1]. Twitter is one of the social networks that has risen to become a top microblogging platform since it is free and allows users to send messages up to 280 characters in length to people worldwide [2].

Unfortunately, Twitter's user popularity, accessibility, and convenience have led to a rise in spam activity over the last few years. Spammers employ a variety of techniques in order to spread their spam messages. These techniques include posting malicious links, sending unsolicited messages to genuine users, and following aggressive behavior to gain attention. It also includes misusing the reply or mentions feature to post unwanted messages, making numerous accounts, which can be done manually or with the help of specific automated systems. It also involves posting duplicate updates on multiple platforms, publishing URLs with entirely irrelevant content, and hijacking popular topics to gain attention [3]. Thus, Twitter has been forced to deploy various detecting algorithms to counteract this behavior, and it has developed mechanisms for users to report unwanted and suspect tweets [4]. Users can also mark and report accounts that transmit duplicate or identical content to many people and content that contains a URL [4], [5].

Although. The majority of the available spam detection tools on Twitter are often good at identifying spammers and blocking their accounts [6], [4], [7]. On the other hand, Spammers may establish new accounts and propagate the spam again. As a result, vital spam detection techniques need to be applied, requiring real-time detection due to the enormous volume of messages submitted every hour. Therefore, research in the literature focusing on tweet-level spam detection is still needed [8–10].

Researchers are using machine learning (ML) approaches to discover the underlying patterns of spammer behavior. [11]. ML-based detection methods use raw data to derive information and make predictions. The first step in this process involve collecting data through the Twitter Application Programming Interface (API). There are some drawbacks to using the Twitter API, such as researcher access to tweet data. Though the most recent 3,200 tweets and the tweets from the last 7-9 days can be extracted through the API, Twitter does not allow researchers to publish the content of the tweets [12]. One may, however, use previous Twitter datasets collected and published by other researchers to satisfy study objectives. Such data must be preprocessed and normalized to eliminate duplication and address missing values. Biased datasets must be resampled.

ML algorithms' inherent nature often performs better for the majority class and disregards the minority class; therefore, the imbalance of spam and non-spam classes dramatically affects classification performance since non-spam classes intrinsically outnumber the spam classes. Furthermore, eventually, it enhances the performance of spam detection on imbalanced Twitter real-time datasets with a range of imbalance degrees. Here we emphasize how to enhance the classification performance and reduce the number of features using the data mining approaches. The main contributions of this paper include four aspects:

1. The current work examines the impact of applying three Bio-inspired optimization algorithms to extract the optimal feature subsets that help predict the spammers.
2. We build and compare multiple machine learning classifiers, namely BayesNet, C4.5, RandF, kNN, and MLP, to study the typical behavior of spammers based on more than 250,000 public tweets.
3. We carried out experiments training and validating the classification algorithms considering the 10-fold cross-validation and several evaluation metrics to analyze the usefulness of the suggested classification algorithm in predicting the spammers among real unbalanced datasets.
4. Identify the features that help improve the selected classifiers' performance.

The remainder of the article is as follows; Section 2 consists of the existing literature on using the machine learning model in spam prediction. Then Section 3 represents the feature selection techniques utilized in this article. Section 4 discusses the detail regarding executing the suggested classification algorithm and details about experiments and outcomes. Finally, conclusions are given in Section 5.

2. RELATED WORK

The growth of the Twitter social network and its ease of use have attracted the attention of millions of people and spammers. As more individuals engage in social networking platforms, spam has been regarded as a critical challenge. Therefore, most scholars have accomplished much in demoting and controlling spam on social platforms.

Murugan and Devi [13] proposed a hybrid technique using the Decision tree, Particle Swarm Optimization (PSO), and Genetic algorithm to collect and detect the spam in 600 million public tweets in real-time streaming. Notwithstanding the technique scored a high detection rate during experiments, the performance dropped when tested with real-time imbalanced datasets. In order to discover authentic and fake accounts, Adewole et al. [13] suggested a technique by utilizing a principal component analysis and tuned K-means algorithm to cluster spam accounts over a dataset comprised of 2 million tweets. The proposed approach results indicated that the Random Forest (RandF) classifier accomplished the best Accuracy of 96.30%, then the multilayer perceptron with 96%, and last, the Support Vector Machine (SVM), which achieved 95.60%.

In another study, Gupta et al. [15] presented a framework that combined user, text-based, and tweet text features to filter spam tweets in real-time. The proposed technique explores the “Spam Drift” issue, referred to as changing the features of spam tweets over time. Furthermore, because 79% of spam tweets hold suspicious URLs, scholars have developed the tool through URL crawl. The Neural Network classifier achieved an Accuracy of 91.65% compared to other classifiers. Chen et al. [16] suggested a system to improve spam detection using feature discretization. They began by collecting more than 600 million public tweets, then they have labeled 6.5 million spam tweets to extract 12 features. Later, they used six classifiers for the classification process: RandF, C4.5, Bayes Network, NB, kNN, and SVM. The Neural Network classifier achieved an Accuracy of 91.65% compared to other classifiers. Moreover, they studied the effect of feature discretization on continuous balanced and imbalanced datasets.

Later, Lin et al. [17] accomplished a comparative investigation of different machine learning algorithms for real-time Twitter spam detection on mixed sizes of Twitter datasets. They measure the classification performance using F1-score, Accuracy, TPR, and FPR, which showed that RandF and C5.0 surpassed other classifiers on balanced datasets. Yet, these two classifiers had an ordinary performance on imbalanced datasets. Also, Alom et al. [18] offered a framework based on deep learning approaches to detect spammer's tweets. They considered their suggested approach on two real-world datasets in terms of Accuracy, precision, recall, and F1 score. The authors designed two text-based classifiers and a combined classifier that leveraged tweet texts and users' metadata. The experiments achieved better results than other machine learning approaches in terms of Accuracy for the two datasets. In another research, Sun et al. [19] suggested a parallel technique using nine machine learning algorithms to offer a near-real-time spam detection model. They executed experiments under several scenarios to evaluate Twitter spam dataset in Accuracy, TPR, FPR, F1-score, Scalability, and Stability. The results showed that RandF and C5.0 outperformed the other classifiers, and RandF acts more stable than other algorithms.

3. PROPOSED METHOD

This research aims to design a data mining model to enhance classification accuracy by exploring the minimum number of features related to spammers’ behavior. The primary principle is investigating the impact of metaheuristic techniques on enhancing the classification performance over the real-time imbalanced dataset. The process starts with searching the feature space to reduce the feature space and prepare the conditions for the classification stage.

3.1. Metaheuristic Methods

3.1.1. Particle Swarm Optimization search

PSO is a stochastic-based optimization technique suggested by Kennedy and Eberhart [20]. In PSO, a potential nominee solution is presented as a finite-length string called a particle pi in the search space. To discover the shortest solution, the particle adjusts its searching direction according to its best previous experience (p_{best}) and the best experience of its companions' flying experience (gb_{est}). The particles use their local memory and knowledge from the swarm to discover the most promising solution.

Each particle is moving around the n-dimensional search space S with an objective function $f : S \subseteq \Re^n \rightarrow \Re$. Each particle $x_{i,t}$ has a position, a fitness function $f(x_{i,t})$, and ''flies'' through the problem space with a velocity $v_{i,t}$. A new position $z_1 \in S$ is called better than $z_2 \in S$ iff $f(z_1) < f(z_2)$.

A subset of all particles is assigned as its neighborhood to each particle. The best previous experience of all neighbors of particle i is called g_{best}. The best search space position particle i has visited until iteration t is its previous experience p_{best}. Each particle additionally keeps a fraction of its old velocity. The particle updates its velocity and position with the following equations [21]:

$$v_{pd}^{new} = \omega * v_{pd}^{old} + C_1 * rand_1() * (pbest_{pd} - x_{pd}^{old}) + C_2 * rand_2() * (gbest_{d_d} - x_{pd}^{old}) \quad 1$$

$$x_{pd}^{new} = x_{pd}^{old} + v_{pd}^{new} \quad 2$$

3.1.2. Cuckoo search

Cuckoo search is a population-based metaheuristic algorithm proposed by Xin-She Yang and Suash Deb [22]. Several studies demonstrated that the Cuckoo search algorithm is computationally efficient and uncomplicated to implement [23]. The Cuckoo search algorithm is inspired by the brood parasitism of some cuckoo species. Some species use the nests of other host birds to lay their eggs that look like the pattern and color of the native eggs to reduce the probability of discovering them and rely on these birds for accommodating their eggs. Sometimes, some host birds discover and throw the alien eggs away or abandon their nests and build a new one in another place. The goal is to employ new and potentially better solutions (cuckoos) to replace a not-so-good solution in the nests [22]. According to the cuckoo search algorithm, each egg in a nest depicts a solution, and a cuckoo egg represents a new solution. Therefore, the Cuckoo search algorithm is more efficient in exploring the search space as it will ensure the algorithm will not fall into a local optimum.

3.1.3. Bat search

The Bat algorithm is a metaheuristic Swarm Intelligence algorithm proposed by Yang [24]. The bats' capabilities encouraged bats to search for their prey and determine different types of insects and obstacles even in total darkness. Bats emit loud sound pulses to detect the target and avoid obstacles. In order to transpose this manner into an intelligent algorithm, the author declares three assumptions. First, all bats will use echolocation to determine their prey. Secondly, all bats fly randomly, according to their internal encoded frequency (freq), velocity (v), and position in space (x). The update of these three variables at each iteration of the algorithm is as follows:

$$freq_i = freq_{min} + (freq_{max} - freq_{min}) \cdot \beta \quad 3$$

$$v_i^t = v_i^{t-1} + (x_i^{t-1} - x\_best_j) \cdot freq_i \quad 4$$

$$x_i^t = x_i^{t-1} + v_i^t \quad 5$$

Where $\beta \in [0,1]$ is a random vector drawn from a uniform distribution, moreover, as the bat attains a position closer to its target then, it will decrease its loudness (Ai) and increase its rate of pulse emission (ri) as follows:

$$A_i^{t+1} = \alpha \cdot A_i^t \quad (6)$$

$$r_i^{t+1} = r_i^0 \cdot [1 - e^{-\gamma \cdot t}] \quad (7)$$

where α $(0 < \alpha < 1)$ and γ $(\gamma > 0)$ are constants. Finally, the authors assume that the loudness will vary from a significant value to a minimum one.

3.2. Dataset and feature statistics

Most Spammers embed URLs into tweets to refer victims effortlessly to superficial websites to fulfill their illegal goals such as malicious phishing, advertisement, scams, and virus distribution [25]. Due to the security and the privacy policy of Twitter, most researchers designed and developed the Twitter Streaming API to collect tweets. Accordingly, in this article, several experiments have been performed on datasets supplied by [16, 17], which are publicly available on the Internet for scientific research. This article suggests a comparative study between three Bio-inspired optimization algorithms to improve the spam detection rate on Twitter. So scholars in [10] built a Twitter API to extract more than 600 million public streams that contain URLs. After collecting tweets, they utilized Trend Micro's Web Reputation Service, which identified 6.5 million tweets containing malicious URLs. Next, they obtained the features that can help in detecting spam.

We select statistic features of user-profile-based and tweet content-based features to ensure detection precision and lower training duration, as illustrated in Table 1. The Account-level features comprise the account's age, number of followers of the account, number of followings, number of favorites this user has received, number of lists in which the user is an associate, and number of tweets this user has shared. The Account-level based features describe the behaviors of the user account. For example, in the feature "no_following," we think that spammers have more followings than ordinary users in many cases. The second category, content-based features, analyses the content of the tweet. It includes the number of times this tweet has been retweeted, the number of favorites this tweet received, the number of hashtags in this tweet, the number of times this tweet is mentioned, the number of URLs that are included in this tweet, the number of characters and the number of digits in this tweet. The content-based features are associated with analyzing the content of the tweet. For instance, a spammer may add more hashtags to a tweet to attract users to browse and click the malicious URL.

Table 1. Database Features and their categories

Feature Category	Feature Name	Description
User- profile - level	account_age	Age of the account
	no_follower	Number of followers
	no_following	Number of followees
	no_favorites	Number of times the account has been favorited
	no_lists	Number of times the user has been listed
	no_tweets	Number of user-posted tweets
Content-level	no_retweets	Number of time the tweet was re-tweeted
	no_hashtag	Number of hashtags that appears in the tweet

	no_mention	Number of times this tweet is mentioned
	no_urls	Number of URLs contained in the tweet
	no_char	Number of characters in the tweet
	no_digits	Number of digits in the tweet

As described above, experiments will be performed on three collections of a dataset illustrated in Table 2. Where tweets in 5K-Random and 95K-Random were randomly composed and were entirely separate, dataset 95K-Continuous had 11 tweets collected continuously. The non-spam are more often than spam tweets; datasets 2 and 3 are more alike to real-world scenarios as the ratio of spam to non-spam in dataset one is 1:1, while the spam ratio in datasets two and three is 1:19.

Table 2. The three datasets with spam to non-spam ratios

Dataset	Sampling method	No. of spam tweets	No. of non-spam tweets
DB1	5K-Random	5000	5000
DB2	95K-Random	5000	95000
DB3	95K-Continuous	5000	9500

4. Experiments and Results Discussion

During the construction of the experiments, we have applied 10-fold cross-validation to avoid overfitting the learned models to estimate our method's generalized error. For binary classification tasks datasets, Accuracy may be enough. Nevertheless, Accuracy, Precision, and Recall alone may be confusing for highly imbalanced datasets. Therefore, we considered more appropriate performance measures to compare different classifiers to handle imbalanced datasets, such as Accuracy, Precision, Recall, Specificity, F1-score [25]. Their formulae are shown in Equations 8-12.

$$\text{Accuracy}=\frac{(\text{TP}+\text{TN})}{(\text{TP}+\text{FP}+\text{TN}+\text{FN})} \quad 8$$

$$\text{Precision}=\frac{\text{TP}}{\text{TP}+\text{FP}} \quad 9$$

$$\text{Recall}=\frac{\text{TP}}{\text{TP}+\text{FN}} \quad 10$$

$$\text{Specificity}=\frac{\text{TN}}{\text{TN}+\text{FP}} \quad 11$$

$$\text{F1-score}=\frac{2\times\text{Precision}\times\text{Recall}}{\text{Precision}+\text{Recall}} \quad 12$$

Where TP: true positives; TN: true negatives; FP: false positives; FN: false negatives.

We plan to run several simulation experiments to demonstrate the reliability of the suggested algorithm. In the simulation experiments, we have chosen to apply the 10-fold cross-validation method for the validation, and separately recorded classification Accuracy, Precision, Recall, Specificity, and F1-score of BayesNet, C4.5, RandF, kNN, and MLP classifiers.

First of all, we should explore the classification outcomes without feature selection to verify the performance of the proposed feature selection techniques. Table 3 briefs the performance before conducting the phase of feature selection. Firstly, regarding the 5k-random dataset, the RandF classifier outperformed the other classifiers by an Accuracy of 86.8%, Precision 85.8%, Recall 88.2%, Specificity 85.4%, and F1-score 87%. Considering the 95K-Random dataset, the RandF classifier outperformed the other classifiers by an Accuracy of 97.2%, Precision 97.3%, Recall 99.7%, and F1-score 98.5%. At the same time, using the 95K-Continuous dataset, the RandF classifier outperformed the other classifiers by scoring an Accuracy of 99.2%, Precision 99.3%, Recall 99.8%, Specificity 88.2%, and F1-score 99.6%. The results showed good performance in all metrics in RandF and C4.5 classifiers. The best performance is achieved when using the 95K-Continuous dataset.

Table 3. The evaluation results of classifiers on datasets 1, 2 and 3

5k-random					
Classifier	Accuracy	Precision	Recall	Specificity	F1-score
BayesNet	0.742	0.729	0.772	0.713	0.750
C4.5	0.830	0.824	0.837	0.821	0.830
RandF	**0.868**	**0.858**	**0.882**	**0.854**	**0.870**
kNN	0.741	0.753	0.717	0.765	0.734
MLP	0.728	0.781	0.635	0.822	0.700
95K-Random					
Classifier	Accuracy	Precision	Recall	Specificity	F1-score
BayesNet	0.939	0.963	0.973	0.284	0.968
C4.5	0.967	0.974	0.991	0.508	0.982
RandF	**0.972**	**0.973**	**0.997**	0.488	**0.985**
kNN	0.941	0.969	0.968	0.430	0.969
MLP	0.956	0.957	0.997	0.167	0.977
95K-Continuous					
Classifier	Accuracy	Precision	Recall	Specificity	F1-score
BayesNet	0.962	0.985	0.975	0.717	0.980
C4.5	0.990	0.992	0.996	0.865	0.994
RandF	**0.992**	**0.993**	**0.998**	**0.882**	**0.996**
kNN	0.962	0.981	0.978	0.655	0.980
MLP	0.968	0.979	0.987	0.602	0.983

The next step is to investigate the potential of using feature selection techniques. The feature selection step aims to simultaneously recognize the essential features with the highest relevancy for target classes and minimum redundancy with other features in the dataset. At the end of this step, a subset of features is chosen for the next round. The selected parameter selection and features by the Bat, Cuckoo, and PSO techniques are listed in Tables 4 and 5. The table shows the parameters selected for the BAT, Cuckoo, PSO and algorithms. Worth mentioning that the number of features has significantly decreased compared with the initial dataset. In this phase, we decreased the size of 5k-random, for example, features from 12 to only 9-2 features.

Table 4. Parameters selection for PSO, BAT and Cuckoo algorithms

Algorithm	**Parameter settings**
Bat	The initial population = 20 $A(0) = 0.9$, $r(0) = 0.9$, $\gamma = \sigma = 0.9$ Maximum number of iterations = 40
Cuckoo	The initial population = 20 $\beta = 1.5$ Maximum number of iterations = 40.
PSO	The initial population = 20 c1 and c2 = [0-2] Maximum number of iterations = 40

Table 5. The optimal features by the Bat, Cuckoo, and PSO techniques

5k-random		
Technique	**No. of Features**	**Selected Features**
Bat Search	9	no_follower, no_userfavourites, no_lists, no_tweets, no_retweets, no_hashtag, no_urls, no_char, no_digits
Cuckoo Search	2	no_tweets, no_digits
PSO Search	8	no_follower, no_userfavourites, no_lists, no_tweets, no_retweets, no_hashtag, no_char, no_digits
95k-random		
Technique	**No. of Features**	**Selected Features**
Bat Search	6	no_lists, no_tweets, no_hashtag, no_urls, no_char, no_digits
Cuckoo Search	3	no_userfavourites, no_urls, no_digits
PSO	8	no_userfavourites, no_lists, no_tweets, no_retweets, no_hashtag, no_urls, no_char, no_digits
95k-continuous		
Technique	**No. of Features**	**Selected Features**
Bat Search	5	no_follower, no_following, no_tweets, no_hashtag, no_usermention
Cuckoo Search	5	account_age, no_userfavourites, no_tweets, no_hashtag, no_usermention
PSO	4	no_follower, no_tweets, no_hashtag, no_usermention

Figures 1-3 and Tables 6-8 show the relationship between Bat, Cuckoo, and PSO search algorithms using the 5k-random, 95k-continuous, and 95k-random datasets. Results showed that Bat, Cuckoo, and PSO techniques share the "no_tweets" and "no_digits" features using the 5k-random dataset. At the same time, Bat and Cuckoo algorithms share six features. Results using the 95k-continuous dataset indicated that Bat, Cuckoo, and PSO techniques share the "no_tweets," "no_usermention," and "no_hashtag" features. At the same time, Bat and PSO algorithms share the "no_follower" feature. Regarding using the 95k-random dataset, results indicated that Bat, Cuckoo, and PSO techniques share the "no_urls" and "no_digits" features. The Bat and PSO techniques share the "no_tweets," n"o_hashtag," "no_char, " and "no_lists" features.

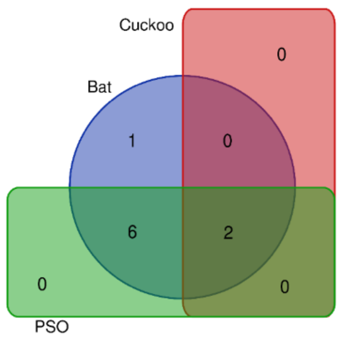

Figure 1. Relationship between Bat, Cuckoo, and PSO search algorithms using the 5k-random dataset

Table 6. Relationship between Bat, Cuckoo, and PSO search algorithms using the 5k-random dataset

Technique	**Total**	**Features**
Bat, Cuckoo, PSO	2	no_tweets , no_digits
Bat, PSO	6	no_retweets , no_userfavourites , no_hashtag, no_char , no_follower , no_lists
Bat	1	no_urls

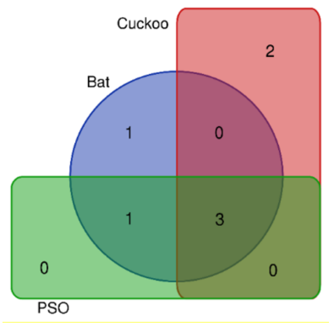

Figure 2. Relationship between Bat, Cuckoo, and PSO search algorithms using the 95k-continuous dataset

Table 7. Relationship between Bat, Cuckoo, and PSO search algorithms using the 95k-continuous dataset

Technique	**Total**	**Features**
Bat, Cuckoo, PSO	3	no_tweets , no_hashtag , no_usermention
Bat, PSO	1	no_follower
Bat	1	no_following
Cuckoo	2	account_age , no_userfavourites

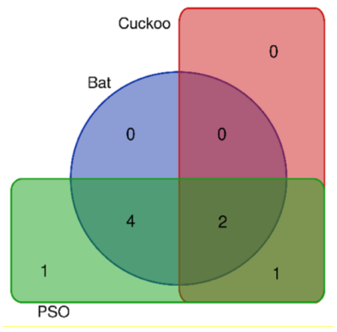

Figure 3. Relationship between Bat, Cuckoo, and PSO search algorithms using the 95k-random dataset

Table 8. Relationship between Bat, Cuckoo, and PSO search algorithms using the 95k-random dataset

Technique	**Total**	**Features**
Bat, Cuckoo, PSO	2	no_urls , no_digits
Bat, PSO	4	no_tweets , no_hashtag , no_char , no_lists
Cuckoo, PSO	1	no_userfavourites
PSO	1	no_retweets

Tables 9-11 reports the classification performance after performing feature selection over the three datasets. Firstly, Table 9 shows the classification results of the features selected by the Bat technique. The RandF classifier trained and tested over 95K-Continuous dataset outperformed the other classifiers achieving an excellent Accuracy, Precision, Recall, Specificity, and F1-score of 99%, 99.2%, 99.8%, 85%, and 99.5%, respectively, with five features.

As the feature selection methods reduced the features of datasets, it also roughly enhanced the overall performance accuracy, as Precision and Recall metrics. The classification results of the nine features selected by the Bat search technique over the 5k-random dataset detected that the classification Accuracy using the Bat algorithm varies between 94% and 96%, Precision varies between 95% and 97%, Recall varies between 96.8% and 99.9%, and F1-score varies between 96.8% and 98%. Next, the classification results of the 6 features selected by the Bat search technique over the 95K-Random dataset indicated that the classification Accuracy using Bat algorithm varies between 95% and 99%, Precision varies between 95% and 99%, Recall varies between 96% and 99%, and F1-score varies between 97% and 99%.

Table 9. The evaluation results of classifiers using the Bat algorithm

5k-random					
Classifier	Accuracy	Precision	Recall	Specificity	F1-score
BayesNet	0.730	0.779	0.711	0.754	0.730
C4.5	0.826	0.820	0.835	0.817	0.828
RandF	**0.826**	**0.820**	**0.835**	**0.817**	**0.828**
kNN	0.755	0.763	0.740	0.770	0.752
MLP	0.709	0.748	0.629	0.788	0.684
95K-Random					
Classifier	Accuracy	Precision	Recall	Specificity	F1-score
BayesNet	0.952	0.958	0.992	0.174	0.975
C4.5	**0.961**	0.963	0.997	0.269	**0.980**
RandF	0.958	0.966	0.990	0.341	0.978
kNN	0.940	**0.969**	0.968	**0.408**	0.968
MLP	0.953	0.954	**0.999**	0.086	0.976
95K-Continuous					
Classifier	Accuracy	Precision	Recall	Specificity	F1-score
BayesNet	0.968	0.982	0.984	0.659	0.983
C4.5	0.984	0.987	0.996	0.755	0.992
RandF	**0.990**	**0.992**	0.998	**0.850**	**0.995**
kNN	0.983	0.992	**0.990**	0.854	0.991
MLP	0.950	0.956	0.993	0.129	0.974

Secondly, Table 10 shows the classification results of the features selected by the Cuckoo technique. The RandF classifier trained and tested over 95K-Continuous dataset outperformed the other classifiers achieving an excellent Accuracy, Precision, Recall, Specificity, and F1-score of 98.9%, 99.1%, 99.7%, 82.7%, and 99.4%, respectively, with five features. As the feature selection methods reduced the features of datasets, it also enhanced the overall performance metrics, such as Precision and Recall metrics.

The classification results of the nine features selected by the Cuckoo search technique over the 5k-random dataset detected that the classification Accuracy using the Cuckoo algorithm varies between 62% and 68%, Precision varies between 63% and 70%, Recall varies between 53% and 68%, and F1-score varies between 60% and 69%. Next, the classification results of the three features selected by the Cuckoo search technique over the 95K-Random dataset indicated that the classification Accuracy varies between 94% and 95%, Precision was close to 95%, Recall varies between 98% and 100%, and F1-score was around 97%.

Table 10. The evaluation results of classifiers using the Cuckoo algorithm

5k-random					
Classifier	Accuracy	Precision	Recall	Specificity	F1-score
BayesNet	0.677	**0.701**	0.616	0.737	0.656
C4.5	**0.689**	0.692	**0.682**	0.696	**0.687**
RandF	0.633	0.637	0.616	0.649	0.626
kNN	0.627	0.636	0.593	0.661	0.614
MLP	0.645	0.689	0.530	**0.761**	0.599
95K-Random					
Classifier	Accuracy	Precision	Recall	Specificity	F1-score
BayesNet	**0.951**	0.952	0.999	0.037	**0.975**
C4.5	**0.951**	0.952	0.999	0.046	**0.975**
RandF	0.947	**0.955**	0.991	0.109	0.972
kNN	0.945	**0.955**	0.988	**0.118**	0.971
MLP	**0.951**	0.952	**1.000**	0.035	**0.975**
95K-Continuous					
Classifier	Accuracy	Precision	Recall	Specificity	F1-score
BayesNet	0.976	0.980	0.995	0.623	0.988
C4.5	0.984	0.987	0.995	0.758	0.991
RandF	**0.988**	**0.991**	**0.997**	**0.827**	**0.994**
kNN	0.977	0.991	0.986	0.823	0.988
MLP	0.956	0.971	0.983	0.435	0.977

Thirdly, Table 11 shows the classification results of the features selected by the PSO technique. The RandF classifier trained and tested over the 5k-random dataset outperformed the other classifiers achieving an excellent Accuracy, Precision, Recall, Specificity, and F1-score of 84.7%, 84.1%, 85.6%, 83.8%, and 84.8%, respectively, with eight features. As the feature selection methods reduced the features of datasets, it also enhanced the overall performance metrics, such as Precision and Recall metrics. The classification results of the nine features selected by the PSO search technique over the 95K-Random dataset detected that the classification Accuracy using the PSO algorithm varies between 93% and 97%, Precision varies between 95% and 97%, Recall varies between 96% and 99%, and F1-score varies between 96% and 98%. Next, the classification results of the four features selected by the Cuckoo search technique over the 95K-Continuous dataset indicated that the classification Accuracy using the PSO algorithm varies between 95% and 98%, Precision and Recall were close to 99%, and F1-score was around 99% using the RandF classifier.

Table 11. The evaluation results of classifiers using the PSO algorithm

5k-random					
Classifier	Accuracy	Precision	Recall	Specificity	F1-score
BayesNet	0.731	0.702	0.751	0.712	0.726
C4.5	0.821	0.817	0.827	0.815	0.822
RandF	**0.847**	**0.841**	**0.856**	**0.838**	**0.848**
kNN	0.750	0.759	0.732	0.767	0.745
MLP	0.715	0.782	0.598	0.833	0.677
95K-Random					
Classifier	Accuracy	Precision	Recall	Specificity	F1-score
BayesNet	0.939	0.963	0.973	0.284	0.968
C4.5	0.967	0.975	0.991	**0.508**	0.983
RandF	**0.972**	**0.974**	**0.997**	0.489	**0.985**

kNN	0.942	0.970	0.968	0.430	0.969
MLP	0.956	0.958	0.998	0.168	0.977
95K-Continuous					
Classifier	Accuracy	Precision	Recall	Specificity	F1-score
BayesNet	0.975	0.983	0.991	0.669	0.987
C4.5	0.982	0.985	0.996	0.721	0.991
RandF	**0.986**	**0.990**	**0.995**	0.813	**0.993**
kNN	0.979	0.990	0.988	**0.820**	0.989
MLP	0.950	0.956	0.993	0.129	0.974

Figures 4-6 and Tables 12-14 show the relationship between 5k-random, 95k-continuous, and 95k-random datasets using the Bat, Cuckoo, and PSO search algorithms datasets. Results showed that 5k-random 95k-continuous PSO 95k-random datasets share the "no_tweets" and "no_hashtag" features using the Bat algorithm. At the same time, 5k-random and 95k-continuous datasets share the "no_follower" feature. In contrast, 5k-random and 95k-random datasets shared "no_urls", "no_digits", "no_char", and "no_lists" features. The Cuckoo algorithm's results indicated that 5k-random and 95k-continuous techniques share the "no_tweets" feature. At the same time, 5k-random and 95k-random datasets share the "no_digits" feature, while 95k-continuous and 95k-random datasets share the "no_userfavourites" feature. 95k-continuous and 95k-random datasets have three and one distinct features, respectively. Regarding the PSO technique, results indicated that 5k-random, 95k-continuous, and 95k-random techniques share the "no_tweets" and "no_hashtag" features. The 5k-random and 95k-random techniques share the "no_retweets", "no_userfavourites", "no_digits", "no_char" and "no_lists" features. 95k-continuous and PSO 95k-random datasets have one distinct feature.

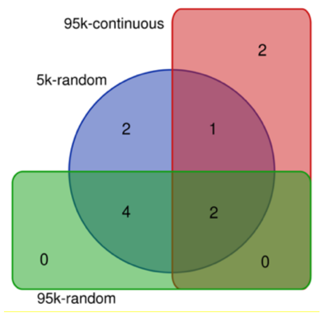

Figure 4. Relationship between 5k-random, 95k-continuous, and 95k-random datasets using the BAT technique

Table 12. Relationship between 5k-random, 95k-continuous, and 95k-random using the Bat technique

Dataset	**Total**	**Features**
5k-random, 95k-continuous, 95k-random	2	no_tweets, no_hashtag
5k-random, 95k-continuous	1	no_follower
5k-random, 95k-random	4	no_urls, no_digits, no_char, no_lists
5k-random	2	no_retweets, no_userfavourites
95k-continuous	2	no_usermention, no_following

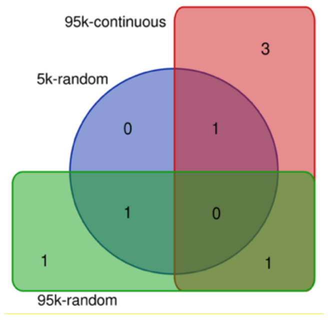

Figure 5. Relationship between 5k-random, 95k-continuous, and 95k-random datasets using the Cuckoo technique

Table 13. Relationship between 5k-random, 95k-continuous, and 95k-random using the Cuckoo technique

Dataset	**Total**	**Features**
5k-random, 95k-continuous	1	no_tweets
5k-random, 95k-random	1	no_digits
95k-continuous, 95k-random	1	no_userfavourites
95k-continuous	3	no_hashtag, account_age, no_usermention
95k-random	1	no_urls

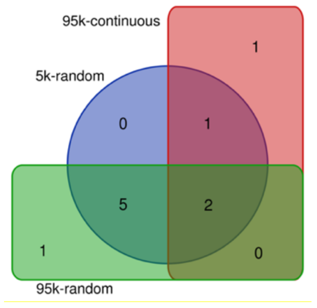

Figure 6. Relationship between 5k-random, 95k-continuous, and 95k-random datasets using the PSO technique

Table 14. Relationship between 5k-random, 95k-continuous, and 95k-random using the PSO technique

Dataset	**Total**	**Features**
5k-random, 95k-continuous, 95k-random	2	no_tweets, no_hashtag
5k-random, 95k-continuous	1	no_follower
5k-random, 95k-random	5	no_retweets, no_userfavouritesno_digits, no_char, no_lists
95k-continuous	1	no_usermention
95k-random	1	no_urls

The best classification performance for the 5k-random dataset was obtained using the Cuckoo algorithm using two features, 17% of the total number of features. While the best classification performance for the 95k-random dataset was obtained using the PSO algorithm using eight features, 67% of the total number of features. Moreover, the best classification performance for the 95K-Continuous dataset was obtained using the Bat algorithm using five features, 42% of the total number of features. The features which accomplished the best results are number of followers, number of followers, number of user-posted tweets, number of hashtags that appear in the tweet, and number of times this tweet is mentioned. The experiment results indicate that the RandF classifier has the best performance, followed by C4.5 and BayesNet. The outcomes demonstrate that all the features in the dataset are crucial for distinguishing spam accounts from legitimate ones. First, it was evident that the three classification algorithms perform pretty well and can be utilized to differentiate between spam and non-spam accounts. Second, although the proposed framework achieved an acceptable performance, a further improvement in accuracy should address model imbalance without compromising performance accuracy. Lastly, the performance of the selected classifiers based on class imbalance also revealed that Random Forest achieved the highest Accuracy, Precision, Recall, and F-score.

5. CONCLUSION

Twitter's user popularity, accessibility, and convenience have led to a rise in spam activity over the last few years. Spammers employ a variety of techniques in order to spread their spam messages. Although. The majority of the available spam detection tools on Twitter are often good at identifying spammers and blocking their accounts. Nevertheless, Spammers always establish new accounts and propagate the spam again. Here we examine the impact of applying three Bio-inspired optimization algorithms to extract the optimal feature subsets that help predict the spammers. Experimental results indicate that the suggested approach significantly enhances the spam detection rate with fewer features. The results showed good performance in all metrics in RandF and C4.5 classifiers. The best performance is achieved using the datasets collected continuously with the Bat algorithm. Also, the features which accomplished the best results are number of followers, number of followers, number of user-posted tweets, number of hashtags that appear in the tweet, and number of times this tweet is mentioned.

REFERENCES

[1] N. Ta, K. Li, Y. Yang, F. Jiao, Z. Tang, and G. Li, "Evaluating Public Anxiety for Topic-based Communities in Social Networks," IEEE Transactions on Knowledge and Data Engineering, 2020.

[2] C.-Y. Liu, M.-S. Chen, and C.-Y. Tseng, "Incrests: Towards realtime incremental short text summarization on comment streams from social network services," IEEE Transactions on Knowledge and Data Engineering, vol. 27, no. 11, pp. 2986-3000, 2015.

[3] S. W. Liew, N. F. M. Sani, M. T. Abdullah, R. Yaakob, and M. Y. Sharum, "An effective security alert mechanism for real-time phishing tweet detection on Twitter," Computers & Security, vol. 83, pp. 201-207, 2019.

[4] K. Thomas, C. Grier, D. Song, and V. Paxson, "Suspended accounts in retrospect: an analysis of twitter spam," in Proceedings of the 2011 ACM SIGCOMM conference on Internet measurement conference, 2011, pp. 243-258.

[5] S. Madisetty and M. S. Desarkar, "A Neural Network-Based Ensemble Approach for Spam Detection in Twitter," IEEE Transactions on Computational Social Systems, vol. 5, no. 4, pp. 973-984, 2018.

[6] F. Benevenuto, G. Magno, T. Rodrigues, and V. Almeida, "Detecting spammers on Twitter," in Proc. Collaboration, Electron. Messaging, Anti-Abuse Spam Conf. (CEAS), vol. 6, 2010, p. 12.

[7] X. Zhang, Z. Li, S. Zhu, and W. Liang, "Detecting spam and promoting campaigns in Twitter," ACM Trans. Web, vol. 10, no. 1, 2016, Art. no. 4.

[8] S. Sedhai and A. Sun, "Semi-supervised spam detection in Twitter stream," IEEE Trans. Comput. Social Syst., vol. 5, no. 1, pp. 169–175, Mar. 2017.

[9] B. Wang, A. Zubiaga, M. Liakata, and R. Procter, "Making the most of tweet-inherent features for social spam detection on Twitter," in Proc.Making Sense Microposts, 2015, pp. 10–16.

[10] C. Chen et al., "A performance evaluation of machine learning-based streaming spam tweets detection," IEEE Trans. Comput. Social Syst., vol. 2, no. 3, pp. 65–76, Sep. 2015.

[11] F. Benevenuto, G. Magno, T. Rodrigues, and V. Almeida, "Detecting spammers on twitter," in Collaboration, electronic messaging, anti-abuse and spam conference (CEAS), 2010, vol. 6, no. 2010, p. 12.

[12] G. Stringhini, C. Kruegel, and G. Vigna, "Detecting spammers on social networks," in Proceedings of the 26th annual computer security applications conference, 2010, pp. 1-9.

[13] N. S. Murugan and G. U. Devi, "Detecting streaming of Twitter spam using hybrid method," Wireless Personal Communications, vol. 103, no. 2, pp. 1353-1374, 2018.

[14] K. S. Adewole, T. Han, W. Wu, H. Song, and A. K. Sangaiah, "Twitter spam account detection based on clustering and classification methods," The Journal of Supercomputing, vol. 76, no. 7, pp. 4802-4837, 2020.

[15] H. Gupta, M. S. Jamal, S. Madisetty, and M. S. Desarkar, "A framework for real-time spam detection in Twitter," in 2018 10th International Conference on Communication Systems & Networks (COMSNETS), 2018: IEEE, pp. 380-383.

[16] C. Chen, J. Zhang, X. Chen, Y. Xiang, and W. Zhou, "6 million spam tweets: A large ground truth for timely Twitter spam detection," in 2015 IEEE international conference on communications (ICC), 2015: IEEE, pp. 7065-7070.

[17] G. Lin, N. Sun, S. Nepal, J. Zhang, Y. Xiang, and H. Hassan, "Statistical Twitter Spam Detection Demystified: Performance, Stability and Scalability," IEEE access, vol. 5, pp. 11142-11154, 2017.

[18] Z. Alom, B. Carminati, and E. Ferrari, "A deep learning model for Twitter spam detection," Online Social Networks and Media, vol. 18, p. 100079, 2020.

[19] N. Sun, G. Lin, J. Qiu, and P. Rimba, "Near real-time twitter spam detection with machine learning techniques," International Journal of Computers and Applications, pp. 1-11, 2020.

[20] Kennedy, James, and Russell Eberhart. "Particle swarm optimization." Proceedings of ICNN'95-international conference on neural networks. Vol. 4. IEEE, 1995.

[21] Bansal, Jagdish Chand. "Particle swarm optimization." Evolutionary and swarm intelligence algorithms. Springer, Cham, 2019. 11-23.

[22] Yang, Xin-She, and Suash Deb. "Cuckoo search via Lévy flights." 2009 World congress on nature & biologically inspired computing (NaBIC). Ieee, 2009.

[23] Jain, Shashank. "Birds: Particle Swarm and Cuckoo Search Optimization." Nature-Inspired Optimization Algorithms with Java. Apress, Berkeley, CA, 2022. 81-126.

[24] Yang, Xin-She. "Bat algorithm for multi-objective optimisation." International Journal of Bio-Inspired Computation 3.5 (2011): 267-274.

[25] Miao, Jiaju, and Wei Zhu. "Precision–recall curve (PRC) classification trees." Evolutionary intelligence (2021): 1-25.