International Journal of Information Management Data Insights 1 (2021) 100006

Contents lists available at ScienceDirect

International Journal of Information Management Data Insights

journal homepage: www.elsevier.com/locate/jjimei

A comprehensive study of spam detection in e-mails using bio-inspired optimization techniques

Jai Batra[a], Rupali Jain[a], Vinay A. Tikkiwal[a,*], Amrita Chakraborty[b]

[a] Department of Electronics and Communication Engineering, Jaypee Institute of Information Technology, Noida 201304, India
[b] St. Mary's School, New Delhi 110029, India

ARTICLE INFO

Keywords:
Machine learning
Spam
k-Nearest Neighbours
Classification
Grey wolf optimization
Firefly optimization
Chicken swarm optimization
Grasshopper optimization
Whale optimization

ABSTRACT

Electronic mail is a medium of communication used frequently for conveying a variety of information. It has become an integral part of people's lives owing to its ease of access and capability to reach out to a large number of recepients without much hassle. This boon, however, has turned into a bane due to exploitation by marketers for publicizing their products, and scammers for fooling people into falling for their schemes. Such e-mails are usually termed as spam e-mails. The menace of spam e-mails requires attention because in addition to consuming resources like bandwidth and storage, they are time-consuming as their removal may require manual effort. This paper proposes a classification method based on k-Nearest Neighbours integrated with several bio-inspired optimization techniques to classify e-mails as spam or legitimate. The study first evaluates the performance of three distance metrics namely Euclidean, Manhattan, and Chebyshev, when utilized in k-Nearest Neighbours classification. Further, five bio-inspired algorithms namely, Grey wolf optimization, Firefly optimization, Chicken swarm optimization, Grasshopper optimization, and Whale optimization, have been explored and their performance is compared based on different evaluation measures like accuracy, precision, recall, speed of convergence to global optimum solution, *F*1-measure and computational time.

1. Introduction

An electronic mail (e-mail) presents a user with a reliable medium to communicate with small as well as large numbers of people by providing a cost-effective and user-friendly platform. E-mail has a large user base because it is a fast and free medium of communication available to everyone. The main problem associated with its usage is the excessive number of unsolicited e-mails that a user receives from sources that are either marketers or scammers. These messages are termed as spam messages and their existence has become a serious problem for users because they consume bandwidth, affect the server storage and are also a waste of time to tackle manually.

Spam e-mails containing unwanted and irrelevant product promotions sent by spammers have a tendency to frequently flood the inbox and create storage issues for other relevant and important e-mails. E-mails carrying information such as commercial offers, lotteries, bank-related offers, etc, are most likely to be sent by scammers who impart information that can fool a certain number of gullible users, and extort money from them. Moreover, such e-mails may contain viruses that severely affect the user's device. These activities by spammers and scammers pose major concerns because information conveyed by spam and non-spam emails are perceived differently by different users and even though a vast majority of users are able to identify spam e-mails and are aware of threats posed by them, a significant amount of users who receive spam e-mails are unaware of and respond to such e-mails becoming a viable source of income for the sender. This is why detection of spam e-mails has become crucial in this fast evolving and technology-dependent world.

In this study, different spam-detection methodologies are established and employed for classification between spam e-mails and regular (legitimate) emails. The classification method of k-Nearest Neighbours (k-NN) is used in this study because the frequency of occurrence of spam and legitimate e-mails are seen to be comparable in the data set employed. Bio-inspired optimization algorithms integrated with k-NN classification have been used to optimize the results of classification. These algorithms are called meta-heuristics, which means they rely upon the iterative improvement of either multiple solutions or a single solution.

The research questions to be answered in this study are given as follows:

- Since, k-NN requires distance metrics for computation of the class representatives of spam and legitimate classes, the first research question that needs to be answered in this study is to find out which of the commonly used distance metrics namely Euclidean, Manhat-

* Corresponding author.
E-mail address: vinay.anand@jiit.ac.in (V.A. Tikkiwal).

https://doi.org/10.1016/j.jjimei.2020.100006
Received 20 October 2020; Received in revised form 30 November 2020; Accepted 19 December 2020

tan, and Chebyshev, has the best performance in terms of common evaluation measures.
- The second research question is to find out the best model among Grey wolf optimization (GWO) algorithm, Firefly optimization algorithm (FOA), Chicken swarm optimization (CSO), Grasshopper optimization algorithm (GOA) and Whale optimization algorithm (WOA) in terms of speed of convergence to globally optimal solution as well as other evaluation measures.

The rest of this paper is organized as follows. Section 2 gives a comprehensive review of the related work. Section 3 presents the proposed methodologies followed by Section 4 which gives the description of data set used and the experimental results obtained. Section 5 presents the discussion of results, Section 6 gives the contribution of this study to literature, and implications to practice are given by Section 7. Section 8 highlights the conclusions drawn, and future scope of this study is given by Section 9.

2. Related work

Spam messages sent by marketers to promote and advertise their products are considered as a nuisance by most people whose limited server storage is filled-up by these unwanted e-mails (Raad et al., 2010). The amount of time it takes to remove all the spam e-mails hinders the productivity of individuals on a daily basis. Some innocent users may also become prey to attacks like e-mail spoofing (Hu and Wang, 2018) and phishing schemes (Banu and Banu, 2013; Halaseh and Alqatawna, 2016), that include mass-forwarded e-mails by scammers who try to steal money and bank account details from users. Spam e-mails are also used by attackers and hackers for the distribution of viruses and other such dangerous software hidden behind attractive links and exciting offers (Karim et al., 2019).

The problem of spam e-mails, therefore, needs to be addressed promptly and effective measures need to be taken to curb the problem. Attempts have been made in reducing spam e-mails acquired on a daily basis by e-mail users. These range from creation of advanced spam-filtering tools to introduction of anti-spamming laws in the United States which would prevent spammers from sending unwanted messages. Several approaches have been proposed to detect and filter spam e-mails whose overview can be found in Wang and Cloete (2005). To reduce the number of spam messages, IP address filtering (Bajaj et al., 2017) has also been introduced, which is a heuristic approach that depends on the source's IP address in an incoming e-mail to determine its legitimacy as a non-spam e-mail. Malicious URLs detection systems are proposed which remove spam content and malicious URLs in email (Ranganayakulu and C., 2013; Rathod and Pattewar, 2015). A complete spam detection system named Cosdes has also been created which possesses an efficient near-duplicate matching scheme and a progressive update scheme (Tseng et al., 2011). The Blacklist method is another approach, which rejects the acceptance of an e-mail with an address that can be found on the list of sources that cannot be trusted. Spam detection methods that employ text categorization using hybrid algorithms like Neuro-SVM (Dhawan and Simran, 2018) and NLP pre-processing methods (Etaiwi and Naymat, 2017) have also been developed. Cryptography has been explored which is a tool that requires a digital signature on an e-mail as proof of authorization, otherwise, it is discarded by the filter. Despite all these efforts, sooner or later, spammers succeed in finding ways to evade these approaches. Thus, there is a need for more reliable and sophisticated approaches that are capable of minimizing, if not eliminating, all spam e-mail. For this purpose, various machine learning approaches have been adopted in which usually a group of spam and legitimate e-mail messages are used for training a learning algorithm so that future incoming e-mail can be automatically categorized. Examples include Artificial Neural Network (Özgür et al., 2004), Deep Neural Network (Barushka and Hájek, 2018), Granular Support Vector Machine – Boundary Alignment algorithm (Tang et al., 2006), Naive Bayesian algorithm (Deshpande et al., 2007), Random Weight Network (Faris et al., 2019), and a quadratic-neuron-based neural tree (QUANT) (Su et al., 2010) to name a few. An excessive number of features while detecting spam e-mails may negatively affect the performance of a learning classifier, therefore methods have also been proposed for feature selection (Dutta et al., 2018) and extraction (Diale et al., 2019; Inuwa-Dutse et al., 2018) like Common Vector Approach (Gunal et al., 2006) and Local Concentration (Zhu and Tan, 2010).

In this study, this categorization between spam and legitimate e-mails will be done with the help of bio-inspired algorithms. Some research has already been conducted in this domain using algorithms like Antlion Optimization algorithm (Naem et al., 2018), Krill Herd Optimization algorithm (Faris et al., 2015), Octopods (Sadek et al., 2019), and Modified K-Means integrated Levy flight Firefly Algorithm with chaotic maps (Aswani et al., 2018). This study seeks to understand the utility of other bio-inspired techniques in solving the spam detection problem.

3. Methods used

Bio-inspired computing (BIC) includes algorithms inspired by various biological phenomenons. These algorithms are based on the behavior and natural tendencies of animals, birds, insects, and other unique characteristics of nature. To develop a model from such biological and naturally occurring phenomenons, there is a requirement of adoption of a proper depiction of problem, assessment of the solution's quality based on a fitness function, and definition of operators used to develop a new set of solutions. BIC algorithms require several algorithm-dependent parameters because each problem is unique and they also require a large number of iterations to optimize the objective function, which can prove to be disadvantageous. These algorithms, however, have two significant advantages. The first is an excellent information-sharing mechanism that helps the algorithm to converge faster, and the second is the lower probability of getting stuck in the locally optimum solution. Other advantages of using BIC includes detection of patterns that were previously unknown and lower reliance on mathematical modeling or exhaustive training (Sekh et al., 2020).

To solve the problem of classification using meta-heuristics, several BIC algorithms have been used in this paper. The first algorithm is GWO, which mimics the hunting nature of grey wolves in nature. Another optimization algorithm is CSO, which is inspired by the behavior and lifestyle of a chicken swarm which consists of roosters, hens and chicks. The third algorithm is FOA and it works based on the flashing behavior of fireflies as a measure of their attractiveness. GOA is an algorithm that mathematically models and mimics the behavior of grasshopper swarms in nature. A relatively new optimization algorithm is also employed in this study which is called WOA and it revolves around simulating the search for prey, encircling prey, and bubble-net foraging behavior of humpback whales.

The classification technique of k-NN and its amalgamation with proposed BIC techniques is discussed in this section along with a detailed explanation of the approaches. The corresponding framework of proposed model in this study is illustrated in Fig. 1.

3.1. k-Nearest Neighbours classification

This study has employed k-NN for classification of data into spam e-mail and non-spam (or legitimate) e-mail based on a set of attributes and supervised data set. The data set is an n-dimensional array of $m \times n$ and it is split into two subsets termed as training data D_{train} of $p \times n$, and testing data D_{test} of $q \times n$. The supervised training data is initially classified into two categories namely spam and legitimate e-mails based on their classification values stored as class in B_{train}.

The BIC function shown in Algorithm 1 corresponds to the different BIC algorithms proposed in this paper. These BIC algorithms are used to compute multiple class representatives of the two categories,

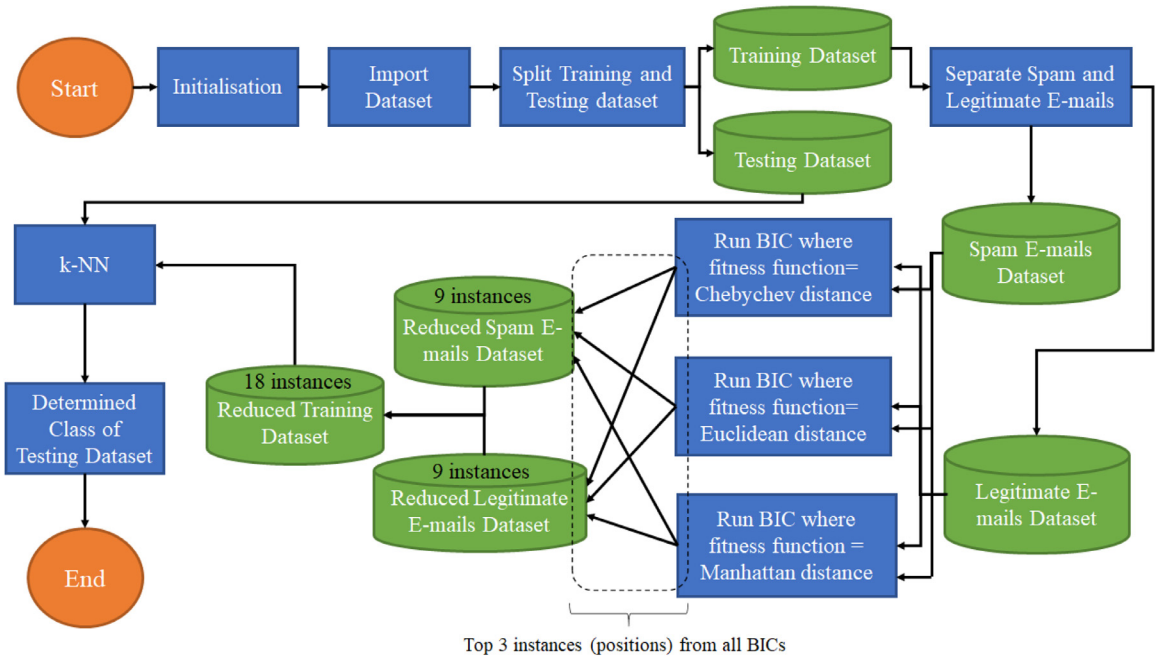

Fig. 1. Framework of the proposed model in this study.

Algorithm 1 MAIN

Input: D_{train}, D_{test}, B_{train} and B_{test}
Output: ζ
$r \leftarrow \{a_1, a_2, \ldots, a_p\} \in D_{train}$
$s \leftarrow \{b_1, b_2, \ldots, b_q\} \in D_{test}$
$v \leftarrow \{c_1, c_2, \ldots, c_p\} \in B_{train}$
$w \leftarrow \{d_1, d_2, \ldots, d_q\} \in B_{test}$
$Initialize : \phi_i \leftarrow \overrightarrow{X}_{rand} \mid i \in \{1, 2, 3, \ldots, 18\}$
$\zeta_i \leftarrow 0 \mid i \in \{1, 2, 3, \ldots, q\}$
for $i = 1$ to p **do**
  if $v_i = 1$ **then**
    $D_{spam_i} \leftarrow r_i$
  else
    $D_{legit_i} \leftarrow r_i$
  end if
end for
$\overline{x}_{spam} \leftarrow mean(D_{spam})$
$\overline{x}_{legit} \leftarrow mean(D_{legit})$
$\{\phi_1, \phi_2, \phi_3\} \leftarrow BIC(D_{spam}, \overline{x}_{spam}, C(a, b))$
$\{\phi_4, \phi_5, \phi_6\} \leftarrow BIC(D_{spam}, \overline{x}_{spam}, E(a, b))$
$\{\phi_7, \phi_8, \phi_9\} \leftarrow BIC(D_{spam}, \overline{x}_{spam}, M(a, b))$
$\{\phi_{10}, \phi_{11}, \phi_{12}\} \leftarrow BIC(D_{legit}, \overline{x}_{legit}, C(a, b))$
$\{\phi_{13}, \phi_{14}, \phi_{15}\} \leftarrow BIC(D_{legit}, \overline{x}_{legit}, E(a, b))$
$\{\phi_{16}, \phi_{17}, \phi_{18}\} \leftarrow BIC(D_{legit}, \overline{x}_{legit}, M(a, b))$
$\zeta \leftarrow kNN(\phi, s, k)$

using three distinct distance metrics that will be substituted in function $Dist(a, b)$. The first distance metric used in this study is Euclidean distance. Eq. (1) gives Euclidean distance E between any two points a and b as the length of line segment connecting them in n-dimensional space.

$$E(a, b) = \sqrt{\sum_{i=1}^{n} (a_i - b_i)^2} \tag{1}$$

The second distance metric used is the Manhattan distance in which the usual metric of Euclidean geometry is replaced by a new metric where the distance between two points is the sum of the absolute differences of their coordinates. Eq. (2) gives Manhattan distance M between any two points a and b in n-dimensional space as the sum of the lengths of the projections of the line segment between the points onto the coordinate axes.

$$M(a, b) = \sum_{i=1}^{n} |a_i - b_i| \tag{2}$$

The third distance metric used is the Chebyshev distance or L_∞ metric. It is a metric defined on a n-dimensional vector space where the distance between two data points is the maximum of their differences along any coordinate dimension. It is given by Eq. (3).

$$C(a, b) = \lim_{c \to \infty} \left(\sum_{i=1}^{n} |a_i - b_i|^c \right)^{\frac{1}{c}} \tag{3}$$

After establishment of class representatives, their set ϕ is treated as the new training data and used for finding the distance between class representatives and a testing data point.

The k-NN algorithm as shown in Algorithm 2 calculates the distance

Algorithm 2 k-NN

Input: ϕ, s and k
Output: ζ
for $i = 1$ to q **do**
  for $j = 1$ to 18 **do**
    $\psi_{neighbors} \leftarrow$ Compute $Dist(\phi_j, s_i)$
  end for
  $X \leftarrow sort_{ascending}(\psi_{neighbors})$
  if majority class among first k distances in X **then**
    $\zeta_i \leftarrow 1$
  else
    $\zeta_i \leftarrow 0$
  end if
end for

from each point of the testing data to all the points of the training data. A particular data point is grouped into the class, which more prevalent among its nearest neighbors based on proximity to the majority of its neighbors that are determined by "k". The value of "k" is presumed to be

a positive integer which is transferred as an input to the k-NN function. This computed class is stored in ζ_i for all instances of D_{test}.

These three metrics in collaboration with k-NN provide good results due to the efficiency of their usage and the many salient features of k-NN. Besides simplicity, k-NN is useful when the data set is large because it is easy to implement as little or no training time is required in addition to no mandatory prior knowledge of the data set. Some drawbacks are also associated with k-NN because it stores all of the training data and it is a challenging task to determine a suitable value of "k" that k-NN takes as input. This choice is important because a small value of "k" would lead to an increase in the variance of the model, and a large value would lead to an increase in the bias of the model. Thus, as the value of "k" increases, the error encountered during training the data increases with an increase in bias, but the error associated with testing the data is likely to decrease simultaneously due to decrease in variance. Hence, the value of "k" plays a crucial role in the variance-bias trade-off and has been chosen to be 5 in this study.

Despite its few limitations, it is the most sought-after classification method because these limitations can be handled by making some alterations to the standard algorithm. This approach has a plethora of applications in many domains, such as pattern recognition, recommendation systems, data mining, intrusion detection, semantic searching, and anomaly detection.

3.2. Hybrid GWO – k-NN approach

GWO is a meta-heuristic optimization algorithm that employs an approach inspired by the predatory nature of grey wolves in nature by mimicking their hunting mechanisms (Mirjalili et al., 2014). Grey wolves are considered apex hunters and the GWO algorithm imitates the structure of their leadership in a pack based on their hierarchical order and their hunting techniques. It considers four types of grey wolves named as alpha (α), beta (β), delta (δ) and omega (ω) who are recognized for understanding their leadership hierarchy.

The GWO algorithm makes use of a grey wolf population in which each wolf performs as a search agent, and it computes fitness for each of these search agents using a pre-defined fitness function. Over a number of iterations, it re-defines the wolf positions to hunt their prey. The search agent with the best resultant fitness value is termed as the α, while the second-best fitness value is termed as β and third-best fitness value is termed as the δ. The remaining wolves are termed as ω.

In nature, grey wolves encircle their prey during a hunt by updating their positions ($\vec{X}_g$) with reference to the position of the prey ($\vec{X}_o$). It can be mathematically modeled in the form of Eqs. (4)–(7) given below, where $\vec{P}$, $\vec{Q}$ and $\vec{R}$ are coefficients.

$$\vec{X}_g(t+1) = \vec{X}_o(t) - \vec{P}.\vec{R} \tag{4}$$

$$\vec{R} = |\vec{Q}.\vec{X}_o(t) - \vec{X}_g(t)| \tag{5}$$

$$\vec{P} = 2\vec{v}.\vec{y}_1 - \vec{v} \tag{6}$$

$$\vec{Q} = 2.\vec{y}_2 \tag{7}$$

The components $\vec{y}_1, \vec{y}_2$ have random values in the interval [0,1] and $\vec{v}$ is decreased linearly over time from 2 to 0 by following Eq. (8).

$$v = 2 - \left[\frac{2t}{MaxIteration}\right] \tag{8}$$

The β and δ, hunt their prey under the guidance of their α. Thus, the fittest three solutions generated are saved and used. The rest of the search agents and including the ω wolf, update their positions following the position of the α. Eqs. (9)–(11) mathematically model this process.

$$\vec{X}_g(t+1) = \frac{\vec{X}_1 + \vec{X}_2 + \vec{X}_3}{3} \tag{9}$$

$$\vec{X}_1 = \vec{X}_\alpha - \vec{P}_1.\vec{R}_\alpha, \vec{X}_2 = \vec{X}_\beta - \vec{P}_2.\vec{R}_\beta, \vec{X}_3 = \vec{X}_\delta - \vec{P}_3.\vec{R}_\delta \tag{10}$$

$$\vec{R}_\alpha = |\vec{Q}_1.\vec{X}_\alpha - \vec{X}_g|, \vec{R}_\beta = |\vec{Q}_2.\vec{X}_\beta - \vec{X}_g|, \vec{R}_\delta = |\vec{Q}_3.\vec{X}_\delta - \vec{X}_g| \tag{11}$$

When integrated with k-NN, the proposed model in Algorithm 3, is

Algorithm 3 Grey Wolf Optimization Algorithm

Input: D, $\overline{x}$ and $F(a,b)$
Output: ϕ
$Initialize: \alpha_{fit} \leftarrow \infty, \beta_{fit} \leftarrow \infty, \delta_{fit} \leftarrow \infty, \alpha_{pos} \leftarrow \overrightarrow{X}_{rand}, \beta_{pos} \leftarrow \overrightarrow{X}_{rand}, \delta_{pos} \leftarrow \overrightarrow{X}_{rand}$
$r_i \leftarrow \{a_1, a_2, \ldots, a_p\} \in D \mid i \in \{1,2,3,\ldots,p\}$
for $t = 1$ to $MaxIteration$ **do**
  for $i = 1$ to p **do**
    Compute $F(r_i, \overline{x})$
    $f \leftarrow F(r_i, \overline{x})$
    if $f < \alpha_{fit}$ **then**
      $\alpha_{fit} \leftarrow f$
      $\alpha_{pos} \leftarrow r_i$
    else if $f < \beta_{fit}$ **then**
      $\beta_{fit} \leftarrow f$
      $\beta_{pos} \leftarrow r_i$
    else if $f < \delta_{fit}$ **then**
      $\delta_{fit} \leftarrow f$
      $\delta_{pos} \leftarrow r_i$
    end if
    $Y_a \leftarrow \alpha_{pos}$
    $Y_b \leftarrow \beta_{pos}$
    $Y_c \leftarrow \delta_{pos}$
  end for
  Update v using (8)
  for $i = 1$ to p **do**
    Compute P and Q using (6) and (7) respectively
    Update $\vec{X}_g$ using (9)
    $r_i \leftarrow \vec{X}_g$
  end for
end for
$\phi \leftarrow \{Y_a, Y_b, Y_c\}$

used for the identification of class representatives of both categories. Even though the standard GWO algorithm has disadvantages like a low solving accuracy/precision, inferior local searching ability and slow convergence rate (Singh, 2018), it is used in conjunction with k-NN in this study because of advantages like less number of control parameters and a strong global optimization ability (Luo and Liu, 2019; Saremi et al., 2015). Literature shows applications of GWO in the fields of outlier detection (Aswani et al., 2016), three dimensional path planning for UAVs (Dewangan et al., 2019), and feature selection (Emary et al., 2014; Sharma et al., 2019), to name a few.

3.3. Hybrid CSO – k-NN approach

CSO is a BIC optimization technique that takes inspiration from the hierarchy of leadership in a chicken swarm (Meng et al., 2014). Its parameters are based on the foraging behaviors of different kinds of chicken in the swarm. In each swarm, there is a head rooster, a majority of hens including some mother-hens, and few chicks associated with each mother-hen. This designation is established by computing the fitness values of the chicken. Different kinds of chicken behave according to the following pre-defined rules.

- Several groups exist inside a single chicken swarm, and each group or flock consists of a head rooster that dominates other chickens, some hens that follow the rooster, and a few chicks that move around their mother.

- Whether a particular chicken is a rooster, hen, or chick is known as its identity and it is established by calculation of fitness values of all chickens. The fitness value is also used to assign the flock to each chicken.
- The chickens that obtain the best fitness values are assigned the identity of roosters, and each of these roosters become head rooster of separate flocks. The chickens that obtain the worst fitness values are assigned identity of chicks. The remaining chickens are identified as hens. Since the chicks follow their mothers, the mother-child relationship is randomly established between some hens and the chicks.
- The conditions that remains unchanged, and are only re-defined after several (S) time step include the status of identity essential for establishing hierarchy in a swarm, dominance of head roosters, a roosters relationship with other chickens in its flock and mother-child relationship of chicks and mother-hens.
- Hens forage for food by following the head rooster, who is seen as the leader of the flock. The hens are majority kind of chickens in any flock, and a small fraction of the hens are randomly assigned the identity of mother-hens. A mother-hen can have several chicks scavenging for food around her.
- An important assumption says that any kind of chicken can randomly snatch good food from others while foraging regardless of the fact that it has been already discovered by some other chicken.

In chicken swam, the number of roosters is less than the number of hens because each rooster is assigned the role of leader of the group, and several hens are grouped under one rooster. Since only hens can lay eggs that hatch into chicks, they play a more important role in reaching the best solution, and their number should be kept larger. Therefore, the number of roosters y_r is taken as 15% of the total population, and the number of hens y_h is taken as 70% of the total population. The rest of the population is assigned to the chicks y_c. The mother-hen's population y_{mh} comprises of only some hens; therefore, their number is less as compared to hens and taken as 20% of y_h.

The movement of roosters, hens, and chicks while foraging is determined by different rules. Roosters show dominant behavior as they are the leaders of their flock and get an extensive search space for food. Eqs. (12) and (13) compute their movement.

Where, $X_{i,j}^{t+1}$ is the new position of a particular rooster at index i, $X_{i,j}^{t}$ is the position of that rooster at time t, $Randn(0,\sigma^2)$ represents a Gaussian distribution having value of mean as 0 and that of standard deviation as σ^2, the smallest constant value in a computer is used as ε to avoid zero-division-error, k is a randomly selected roosters index picked from the top fittest solutions of roosters group, fit_i and fit_k are the corresponding fitness values of these chosen roosters.

$$X_{i,j}^{t+1} = X_{i,j}^{t} * (1 + Randn(0,\sigma^2)) \tag{12}$$

$$\sigma^2 = \begin{cases} 1, & \text{if } fit_i \le fit_k, \\ exp\left(\frac{(fit_k - fit_i)}{|fit_i| + \varepsilon|}\right), & \text{otherwise.} \end{cases} \tag{13}$$

Hens follow the head rooster in their flock to search for food. Eqs. (14)–(16) mathematically model their movement.

$$X_{i,j}^{t+1} = X_{i,j}^{t} + U(X_{r_1,j}^{t} - X_{i,j}^{t})Z + V(X_{r_2,j}^{t} - x_{i,j}^{t})Z \tag{14}$$

$$U = exp\left(\frac{(fit_1 - fit_{r_1})}{abs(fit_i) + \varepsilon}\right) \tag{15}$$

$$V = exp(fit_{r_2} - fit_i) \tag{16}$$

Where, Z is a random number uniformly distributed in the interval [0,1], $r_1 \in [1, N]$ is the index of a rooster which is the i^{th} hen's leader, and $r_2 \in [1, N]$ is an index of a rooster or hen, chosen randomly, from the swarm.

The chicks forage for food by moving around their mother in a flock. This movement is given by the Eq. (17).

$$X_{i,j}^{t+1} = X_{i,j} + F(X_{m,j}^{t} - X_{i,j}^{t}) \tag{17}$$

Where, $X_{m,j}^{t}$ is the position of the mother of a chick at index i at time t such that $m \in [1, N]$. The value of F is selected randomly for every chick from the range [0,2], and it expresses the speed with which a chick follows its mother.

Since spam-detection is a problem of minimization in terms of distance-based classification, the fittest chickens are those who have minimum fitness values because the fitness functions are the distance metrics. The class representatives are computed for all three metrics by following the rules of this algorithm as shown in Algorithm 4.

Algorithm 4 Chicken Swarm Optimization Algorithm

Input: D, $\overline{x}$ and $F(a,b)$
Output: ϕ
$Initialize\ Y_{a_{fit}} \leftarrow \infty, Y_{b_{fit}} \leftarrow \infty, Y_{c_{fit}} \leftarrow \infty, Y_a \leftarrow \overline{X}_{rand}, Y_b \leftarrow \overline{X}_{rand}, Y_c \leftarrow \overline{X}_{rand}$
$Initialize\ Y_{hen_i} \leftarrow \overline{X}_{rand} \mid i \in \{1,2,3,\ldots,y_h\},\ Y_{rooster_i} \leftarrow \overline{X}_{rand} \mid i \in \{1,2,3,\ldots,y_r\},\ Y_{mhen_i} \leftarrow \overline{X}_{rand} \mid i \in \{1,2,3,\ldots,y_{mh}\}, Y_{chick_i} \leftarrow \overline{X}_{rand} \mid i \in \{1,2,3,\ldots,y_c\}$
$r_i \leftarrow \{a_1, a_2, \ldots a_p\} \in D$
for $t = 1$ to $MaxGeneration$ **do**
  if $(t\%S) == 0$ **then**
    for $i = 1$ to p **do**
      $\psi_i \leftarrow F(r_i, \overline{x})$
    end for
    $R, \psi_{sorted} \leftarrow sort_{ascending}(r, \psi)$
    for $i = 1$ to p **do**
      if $i <= y_r$ **then**
        $Y_{rooster} \leftarrow R_i$
      else if $i <= y_r + y_h$ **then**
        $Y_{hen} \leftarrow R_i$
      else
        $Y_{chick} \leftarrow R_i$
      end if
    end for
    for $i = 1$ to y_{mh} **do**
      $l \leftarrow rand(j) \mid j \in \{1,2,3,\ldots,y_h\}$
      $Y_{mhen_i} \leftarrow Y_{hen_l}$
    end for
    Randomly assign $Y_{rooster_i}$ to $Y_{hen_j} \mid i \in \{1,2,3,\ldots,y_r\}, j \in \{1,2,3,\ldots,y_h\}$
    Randomly assign Y_{mhen_i} to $Y_{chick_j} \mid i \in \{1,2,3,\ldots,y_{mh}\}, j \in \{1,2,3,\ldots,y_c\}$
  end if
  for $i = 1$ to y_r **do**
    $X_{i,j}^{t} \leftarrow Y_{rooster_i}$
    Compute $X_{i,j}^{t+1}$ using (12)
    $X_{new} \leftarrow X_{i,j}^{t+1}$
    $f \leftarrow F(X_{new}, \overline{x})$
    if $f < F(X_{i,j}^{t}, \overline{x})$ **then**
      $Y_{rooster_i} \leftarrow X_{new}$
      if $f < Y_{a_{fit}}$ **then**
        $Y_{a_{fit}} \leftarrow f$
        $Y_a \leftarrow X_{new}$
      else if $f < Y_{a_{fit}}$ **then**
        $Y_{b_{fit}} \leftarrow f$
        $Y_b \leftarrow X_{new}$
      else if $f < Y_{a_{fit}}$ **then**
        $Y_{c_{fit}} \leftarrow f$
        $Y_c \leftarrow r_i$
      end if
    end if
  end for
  for $i = 1$ to y_h **do**
    $X_{i,j}^{t} \leftarrow Y_{hen_i}$
    Compute $X_{i,j}^{t+1}$ using (14)
    $X_{new} \leftarrow X_{i,j}^{t+1}$
    if $F(X_{new}, \overline{x}) < F(X_{i,j}^{t}, \overline{x})$ **then**
      $Y_{hen_i} \leftarrow X_{new}$
    end if
  end for
  for $i = 1$ to y_c **do**
    $X_{i,j}^{t} \leftarrow Y_{chick_i}$
    Compute $X_{i,j}^{t+1}$ using (17)
    $X_{new} \leftarrow X_{i,j}^{t+1}$
    if $F(X_{new}, \overline{x}) < F(X_{i,j}^{t}, \overline{x})$ **then**
      $Y_{chick_i} \leftarrow x_{new}$
    end if
  end for
end for
$\phi \leftarrow \{Y_a, Y_b, Y_c\}$

The value of time step S is chosen with utmost care. This choice is important because if S is assigned a large value, the algorithm will take longer to converge to a global optimum solution. On the other hand, if S is assigned a small value, the algorithm is likely to get stuck in the local optimum solution. It is observed that the larger is the value of S, the smaller its search space becomes, and this phenomena help in locating the global optimum solution. However, if the chosen value is too large,

the accuracy of the solution is likely to be compromised. In this paper, the value of S is chosen after analysis over multiple simulations.

CSO has been chosen for the purpose of spam-detection due to inspiration drawn from its recent extensive applications in fields like constrained optimization (Wang et al., 2019), parameter estimation (Chen et al., 2016), data clustering (Irsalinda et al., 2017), and feature selection (Hafez et al., 2015). Literature also indicates that a number of modified versions of CSO have been invented to solve problems like the tendency of algorithm to get trapped in the local minima for longer than considered fruitful (Deb et al., 2020). CSO has been seen to outperform a number of other BIC algorithms because it inherits major advantages of many algorithms and has a self-adaptive quality that enables it to solve optimization problems (Meng et al., 2014).

3.4. Hybrid FOA – k-NN approach

FOA is a BIC technique which is meta-heuristic and is utilized in optimization problems to find approximate results. It takes inspiration from the flashing nature of fireflies (Johari et al., 2013). The primary purpose of a firefly's bio-luminescent nature is to attract other fireflies, and the flashing of their tails behaves like a signal system to accomplish this task. Fireflies move based on the brightness of their neighbors, where a brighter firefly is deemed to be more attractive. Therefore, any firefly's movement depends on its own attractiveness and the attractiveness of those around it.

A firefly's light attractive coefficient η is given by the following equation, where d is the distance between two fireflies, γ is a varying coefficient such that $\gamma \to 0$ and η_0 is the light attractiveness of a firefly at $d = 0$.

$$\eta = \eta_0 e^{(-\gamma d^2)} \tag{18}$$

To develop a model from flashing nature of fireflies, three assumptions have been invented for FOA:

- Each firefly will be assumed to feel attracted to all other fireflies because all fireflies are considered uni-sexual.
- The attractiveness of any firefly is considered to be proportional to the level of their brightness, thus, for any two fireflies, the lesser bright firefly will move in the direction of the brighter one due to attraction; however, it should be kept in mind that the intensity of brightness tends to decrease as the mutual distance increases.
- A firefly will move to random positions if there are no fireflies brighter than a given firefly.

The amount of movement of any firefly a to a comparatively more brighter i.e., more attractive firefly b is calculated using-

$$X_{ab} = X_b + \eta_0 e^{(-\gamma_{ab}{}^2)}(X_b - X_a) + \lambda \varepsilon_a \tag{19}$$

The equations define the movement based on three parts. The first part denotes the current location of firefly b, the second part is reflective of movement due to the attraction between firefly a and b, and the third part is added for randomization using the vector of random variables ε_a, which can be drawn from different distributions. In the third part, λ is called the scaling parameter, and it controls the step size. The scaling parameter either decreases exponentially or is kept constant.

This proposed approach as illustrated in Algorithm 5, is beneficial as it optimizes the results and eventually reduces the cost complexity of the k-NN algorithm. Even though the standard FOA faces issues with premature convergence and suffers due to fixed parameters, it is good at exploration and diversification, which makes it advantageous for this study. This algorithm is known to have high applicability (Khan et al., 2016), and amalgamation of standard FOA with other algorithms has been utilized to find solutions in various fields, such as numerical optimization (Pan et al., 2019) and global optimization (Zhang et al., 2016).

Algorithm 5 Firefly Optimization Algorithm

Input: D, $\overline{x}$ and $F(a, b)$
Output: ϕ
$Initialize\ Y_{a_{fit}} \leftarrow \infty$, $Y_{b_{fit}} \leftarrow \infty$, $Y_{c_{fit}} \leftarrow \infty$, $Y_a \leftarrow \overrightarrow{X}_{rand}$, $Y_b \leftarrow \overrightarrow{X}_{rand}$, $Y_c \leftarrow \overrightarrow{X}_{rand}$
$r_i \leftarrow \{a_1, a_2, \ldots, a_p\} \in D \mid i \in \{1, 2, 3, \ldots, p\}$
for $t = 1$ to $MaxIteration$ **do**
  for $i = 1$ to p **do**
    for $j = 1$ to p **do**
      $f_1 \leftarrow F(r_i, \overline{x})$
      $f_2 \leftarrow F(r_j, \overline{x})$
      if $f_2 > f1$ **then**
        Compute X_{ab} using (19)
        $X_{new} \leftarrow X_{ab}$
        $f \leftarrow F(X_{new}, \overline{x})$
        if $f < f_1$ **then**
          $r_i \leftarrow X_{new}$
          if $f < Y_{a_{fit}}$ **then**
            $Y_{a_{fit}} \leftarrow f$
            $Y_a \leftarrow X_{new}$
          else if $f < Y_{a_{fit}}$ **then**
            $Y_{b_{fit}} \leftarrow f$
            $Y_b \leftarrow X_{new}$
          else if $f < Y_{a_{fit}}$ **then**
            $Y_{c_{fit}} \leftarrow f$
            $Y_c \leftarrow r_i$
          end if
        end if
      end if
    end for
  end for
end for
$\phi \leftarrow \{Y_a, Y_b, Y_c\}$

3.5. Hybrid GOA – k-NN approach

Grasshoppers are pests that affect crop production and ruin agriculture. These insects form one continental-scale swarms that are considered one of the largest among all creatures.

In a grasshopper swarm, the swarming characteristic is found in both nymph and adulthood. In the larval phase, grasshopper movement is slow and in small steps whereas in adulthood, long-range and abrupt movement is the essential characteristic of the swarm. It is seen that nymph grasshoppers, in millions, hop and roll like cylinders for moving and they eat almost all vegetation in their path. On becoming adult, they create a swarm in the air and migrate over large distances.

GOA mathematically models and mimics the behavior of grasshopper swarms in nature for solving optimization problems (Saremi et al., 2017). The following mathematical Eq. (20) simulates the swarming behavior of grasshoppers.

$$X_i = H_i + W_i + G_i \tag{20}$$

Where X_i is the position of the i^{th} grasshopper, G_i is the gravity force on it, W_i shows the wind advection, and H_i is the social interaction. These are modeled by the following equations.

$$G_i = -g\widehat{e_g} \tag{21}$$

$$W_i = w\widehat{e_w} \tag{22}$$

$$H_i = \sum_{\substack{j=1 \\ j \neq i}}^{N} h(d_{ij})\widehat{d_{ij}} \tag{23}$$

Where g is the gravitational constant and $\widehat{e_g}$ shows a unity vector towards the center of earth in Eq. (21). The w in Eq. (22) is a constant drift and $\widehat{e_w}$ is a unity vector in the direction of wind. In Eq. (23), d_{ij} is the Euclidean distance between the i^{th} and the j^{th} grasshopper given by Eq. (1), and $\widehat{d_{ij}}$ is a unit vector modeled by Eq. (22). The N denotes the number of grasshoppers.

$$\widehat{d_{ij}} = \frac{x_j - x_i}{d_{ij}} \tag{24}$$

The h is a function that defines the strength of social forces and it is given by Eq. (23), where I indicates the intensity of attraction and r is the attractive length scale.

$$h(d_{ij}) = I e^{\frac{-d_{ij}}{r}} - e^{-d_{ij}} \tag{25}$$

The above equations design an algorithm which is not equipped completely to solve optimization problems. Stochastic algorithms are required to perform exploration and exploitation to determine an accurate approximation of the global optimum. This can be achieved by the incorporation of new parameters as given by Eq. (26).

$$X_i^n = c\left(\sum_{\substack{j=1 \\ j\neq i}}^{N} c\left(\frac{u_n + l_n}{2}\right) h(d_{ij})\widehat{d_{ij}}\right) + \widehat{T_n} \tag{26}$$

This equation shows that the next position of a grasshopper is based on its current position, the position of all other grasshoppers, and the position of the target. Here, u_n is the upper bound and l_n is the lower bound in the n^{th} dimension. The $\widehat{T_n}$ is the value of the n^{th} dimension in the target which is regarded as the best solution found so far and c is a decreasing coefficient to shrink the comfort, repulsion, and attraction zone. To balance exploration and exploitation, c needs to be decreased in proportion to the number of iteration so as to promote exploitation as the iteration count increases. The value of c is calculated using Eq. (27).

$$c = c_{\max} - s\left(\frac{c_{\max} - c_{\min}}{L}\right) \tag{27}$$

Where c_{max} and c_{min} are the maximum and minimum value respectively, s denotes the current iteration, and L denotes the maximum number of iterations.

The proposed model given in Algorithm 6, indicates the importance of GOA in the computation of class representatives because it shows promising results when it comes to escaping from local optima and balancing the exploration and exploitation trends (Mirjalili et al., 2018). Despite having the drawback of slow convergence speed in its standard form, it finds applications in the fields of feature selection (Mafarja et al., 2019), data clustering (Lukasik et al., 2017), and multi-objective optimization to name a few (Mirjalili et al., 2018), usually with modifications.

Algorithm 6 Grasshopper Optimization Algorithm

Input: D, $\overline{x}$ and $F(a, b)$
Output: ϕ
$Initialize\ Y_{a_{fit}} \leftarrow \infty, Y_{b_{fit}} \leftarrow \infty, Y_{c_{fit}} \leftarrow \infty, Y_a \leftarrow \overrightarrow{X}_{rand}, Y_b \leftarrow \overrightarrow{X}_{rand}, Y_c \leftarrow \overrightarrow{X}_{rand}$
$r_i \leftarrow \{a_1, a_2, \ldots, a_p\} \in D \mid i \in \{1, 2, 3, \ldots, p\}$
for $t = 1$ to $MaxIteration$ **do**
  for $i = 1$ to p **do**
    Compute $F(r_i, \overline{x})$
    $f \leftarrow F(r_i, \overline{x})$
    if $f < Y_{a_{fit}}$ **then**
      $Y_{a_{fit}} \leftarrow f$
      $Y_a \leftarrow X_{new}$
    else if $f < Y_{a_{fit}}$ **then**
      $Y_{b_{fit}} \leftarrow f$
      $Y_b \leftarrow X_{new}$
    else if $f < Y_{a_{fit}}$ **then**
      $Y_{c_{fit}} \leftarrow f$
      $Y_c \leftarrow r_i$
    end if
  end for
  $\widehat{T_n} \leftarrow Y_a$
  Update c using (27)
  for $i = 1$ to p **do**
    Compute $\widehat{d_{ij}}$ and $h(\widehat{d_{ij}})$ using (24) and (25) respectively
    Update X_i using (26)
    $r_i \leftarrow \vec{X}_i$
  end for
end for
$\phi \leftarrow \{Y_a, Y_b, Y_c\}$

3.6. Hybrid WOA – k-NN approach

WOA is a new meta-heuristic algorithm as illustrated in Algorithm 7, that mimics the hunting behavior of humpback whales (Mirjalili and Lewis, 2016). The humpback whales hunt small fishes close to the surface by encircling them and swimming around them while shrinking the circle and creating bubbles in their path.

This is called bubble-net feeding behavior and it represents the exploitation phase. Exploration phase is the second phase where the whales search randomly for a prey.

During the first phase, to hunt a prey, humpback whales first encircle the prey. This behavior can be modeled by the Eqs. (28)–(31).

$$\vec{X}_w(t+1) = \vec{X}_p - \vec{A}.\vec{C} \tag{28}$$

$$\vec{C} = |\vec{B}.\vec{X}_p - \vec{X}_w| \tag{29}$$

$$\vec{A} = 2\vec{\tau}.\vec{k}_1 - \vec{\tau} \tag{30}$$

$$\vec{B} = 2.\vec{k}_2 \tag{31}$$

Where t indicates the current iteration, $\overrightarrow{X_p}$ represents the best solution obtained so far, $\overrightarrow{X_w}$ is the position vector. The whales are search agents that update their positions according to the position of the best known solution which is regarded as the prey. The adjustment of the values of $\overrightarrow{A}$ and $\overrightarrow{B}$ vectors control the areas where a whale can be located in the search space of the prey. The components $\vec{k}_1, \vec{k}_2$ have random values in the interval [0,1] and $\vec{\tau}$ is decreased linearly over time from 2 to 0 by following Eq. (32).

$$\tau = 2 - \left[\frac{2t}{MaxIteration}\right] \tag{32}$$

The shrinking encircling behavior of bubble-net foraging is achieved by decreasing the value of τ using Eq. (32). The spiral Eq. (33), is used to create the position of the neighbor search agent.

$$\overrightarrow{X_w}(t+1) = \overrightarrow{C}.e^{yz}.cos(2\pi z) + \overrightarrow{X_p} \tag{33}$$

Where, $\overrightarrow{C}$ denotes the distance of the i^{th} whale and the prey, y is a constant that corresponds to the shape of the logarithmic spiral, and z is a random number in the interval $[-1, 1]$.

The humpback whales swim around the prey within a shrinking circle and along a spiral-shaped path simultaneously. To model the two mechanisms, a probability of 50% is assumed to choose between them during the optimization process and u is a random number chosen from the interval [0,1] for this process. This can be observed in Eq. (34).

$$\overrightarrow{X_w}(t+1) = \begin{cases} \vec{X}_p - \vec{A}.\vec{C} & \text{if } u < 0.5 \\ \overrightarrow{C}.e^{yz}.cos(2\pi z) + \overrightarrow{X_p} & \text{if } u \geq 0.5 \end{cases} \tag{34}$$

For exploitation, the humpback whales search randomly with respect to each others positions. Therefore, $\overrightarrow{A}$ is assigned random values greater than 1 or less than −1, to compel the search agents to move far away from a reference whale. Therefore, the position of a search agent in the

Algorithm 7 Whale Optimization Algorithm

Input: D, $\overline{x}$ and $F(a, b)$
Output: ϕ
$Initialize\ Y_{a_{fit}} \leftarrow \infty, Y_{b_{fit}} \leftarrow \infty, Y_{c_{fit}} \leftarrow \infty, Y_a \leftarrow \overrightarrow{X}_{rand}, Y_b \leftarrow \overrightarrow{X}_{rand}, Y_c \leftarrow \overrightarrow{X}_{rand}$
$r_i \leftarrow \{a_1, a_2, \ldots, a_p\} \in D \mid i \in \{1, 2, 3, \ldots, p\}$
for $t = 1$ to $MaxIteration$ **do**
  for $i = 1$ to p **do**
    Compute $F(r_i, \overline{x})$
    $f \leftarrow F(r_i, \overline{x})$
    $f \leftarrow F(r_i, \overline{x})$
    if $f < Y_{a_{fit}}$ **then**
      $Y_{a_{fit}} \leftarrow f$
      $Y_a \leftarrow x_{new}$
    else if $f < Y_{a_{fit}}$ **then**
      $Y_{b_{fit}} \leftarrow f$
      $Y_b \leftarrow x_{new}$
    else if $f < Y_{a_{fit}}$ **then**
      $Y_{c_{fit}} \leftarrow f$
      $Y_c \leftarrow r_i$
    end if
  end for
  Update τ using (32)
  for $i = 1$ to p **do**
    Compute A and C using (30) and (29) respectively
    if $u < 0.5$ **then**
      if $\mid A \mid >= 1$ **then**
        Update $\vec{X}_w$ using (35)
      else
        $\overrightarrow{X}_{rand} \leftarrow Y_a$
        Update $\vec{X}_w$ using (35)
      end if
    else
      Update $\vec{X}_w$ using (33)
    end if
    $r_i \leftarrow \vec{X}_w$
  end for
end for
$\phi \leftarrow \{Y_a, Y_b, Y_c\}$

exploration phase is updated according to a randomly chosen search agent $\vec{X}_{rand}$ instead of the best search agent found so far. This can be mathematically modeled using equations

$$\vec{X}_w(t+1) = \vec{X}_{rand} - \vec{A}.\vec{C} \tag{35}$$

$$\vec{C} = |\vec{B}.\vec{X}_{rand} - \vec{X}_w| \tag{36}$$

The proposed model, as described in Algorithm 7, shows the working of WOA as discussed in this paper. Literature shows that even though it is a relatively new algorithm, it is able to solve a wide range of optimization problems and outperform a number of the current algorithms (Mohammed et al., 2019). It is advantageous because it has a limited number of parameter (Nasiri and Khiyabani, 2018), very simple calculation, and yields a significant computational accuracy, even though its convergence speed is slow and it easily falls into the local optima (Guo et al., 2020).

In recent literature, it has been used for spam feature selection (Shuaib et al., 2019), community detection (Feng et al., 2020) and multimodal optimization (Li et al., 2019), to name a few.

4. Experimental results

This section consists of a performance analysis of the discussed BIC techniques when used with the three distance metrics, and compares them to determine the best algorithm suitable for classification of spam and legitimate e-mails. All proposed algorithms have been implemented using MATLAB 2018a and the computed results have been acquired using Dell Inspiron 5567 laptop with Intel(R) Core(TM) i3-6006U CPU @ 2.00 GHz Processor, 4.00 GB RAM, 64-bit Operating System, and HDD.

4.1. Description of data set

The data set used in this paper is the æSpambase Data Setg procured from the UCI Machine Learning Repository (Hopkins et al., 1999). This data set is utilized for classification of e-mails as spam and legitimate based on supervised learning in the proposed models of this study. It is a multi-variate data set that consists of 57 continuous type attributes measured over 4601 instances. The division and description of these attributes are given in Table 1. The last column i.e. the 58th attribute of the data set corresponds to the pre-defined class of an instance into spam if value is “1” and legitimate if value is “0”. Number of instances that are pre-classified as spam in this supervised data set is 1813 and those classified as legitimate is 2788.

Since many classification learning algorithms suffer issues with accuracy of prediction due to class-imbalance problem in data sets where the class distributions are largely imbalanced (Drummond and Holte, 2005), the data set used in this study was chosen to curb this problem to a certain degree. The class-imbalance problem generally occurs in data sets in which one class is very infrequent, say 1% of the data set, making it the minority class and the other class is the majority class (Ling and Sheng, 2010). Other factors that impact this problem are the complexity of the data set in terms of concept, the size of the training set, and the classifier used (Japkowicz and Stephen, 2002). However, the data set used in this study consists of 39.4% pre-classified spam e-mails and 60.6% pre-classified legitimate e-mails, making it lightly imbalanced, and therefore, limiting the affects of class imbalance problem in this study. Moreover, to validate the proposed models, the data set is split into two parts, 90% for training each model and 10% for testing each model. Thus, even though k-NN by itself may be affected by this problem, the models proposed in this study, which are used on a lightly imbalanced data set, are not affected to a noticeable degree.

The classification is performed using 10 fold cross-validation procedure, wherein the data set is divided into 10 approximately equal-sized partitions and classification is repeated 10 times. Each time, the classifier trains on 9 out of the 10 partitions, and a single partition is used as test set.

The final result for a proposed model is generated by averaging the results of the 10 evaluations, and recording the mean and variance values.

4.2. Measures used for evaluation

The standard methodology of representing classification results in the form of a confusion matrix, as shown in Table 6, is the primary source used to assess and compare the proposed classification models. In a confusion matrix, true positive (TP) are the e-mails that are correctly classified as spam, whereas true negative (TN) are the e-mails that are correctly classified as legitimate. On the other hand, false positive (FP) are the e-mails that are classified as spam while they are legitimate and finally false negative (FN) are the mis-classified messages that are spam but are classified as legitimate. Table 7 illustrates the values of TP, TN, FP, and FN achieved for all the proposed models with the corresponding distance metric used.

To evaluate the proposed models, four evaluation criteria are adopted in all experiments to compute the values obtained from different BIC techniques and compare the models with each other. Since the classification problem in this study resembles a binary classification problem, flat performance measures are used (Costa et al., 2007). These measures are spam recall, spam precision, accuracy, and $F1$-measure.

Table 1
Description of data-set.

Attribute type	Number	Description
word_freq_WORD	48	This describes the percentage of words in the particular e-mail that match with "WORD", wherein a WORD is any string of alphanumeric characters bounded by non-alphanumeric characters
char_freq_CHAR	6	This describes the percentage of characters in the e-mail that match "CHAR", wherein a CHAR is any alphanumeric character
capital_run_length_average	1	This attribute holds the value of average length of uninterrupted sequences of capital letters
capital_run_length_longest	1	This consists of the length of longest uninterrupted sequence of capital letters
capital_run_length_total	1	This describes the sum of length of uninterrupted sequences of capital letters i.e. total number of capital letters in the e-mail

Table 2
Euclidean distance metric used in k-NN.

Algorithm	Accuracy (%)		Precision (%)		Recall (%)		*F*1-measure (%)		Time (s)
	Mean	Variance	Mean	Variance	Mean	Variance	Mean	Variance	Total
GWO	53.18	10.21	45.74	4.68	100	0	62.75	4.09	669
CSO	54.35	15.38	46.4	7.62	100	0	63.35	6.42	99
FOA	52.29	9.11	45.27	4.85	100	0	62.3	4.27	42,039
GOA	62.43	9.08	82.24	152.43	5.72	34.18	10.28	93.7	21,400
WOA	68.46	409.08	68.5	594.67	83.02	1349.4	65.25	589.7	98

Table 3
Manhattan distance metric used in k-NN.

Algorithm	Accuracy (%)		Precision (%)		Recall (%)		*F*1-measure (%)		Time (s)
	Mean	Variance	Mean	Variance	Mean	Variance	Mean	Variance	Total
GWO	100	0	100	0	100	0	100	0	710
CSO	100	0	100	0	100	0	100	0	118
FOA	100	0	100	0	100	0	100	0	41,458
GOA	100	0	100	0	100	0	100	0	20,629
WOA	100	0	100	0	100	0	100	0	102

Error rate has not been distinctly computed because it is simply complimentary to accuracy. These evaluation measures have been extensively used in literature for analysis of classification algorithms and their applications.

1. *Accuracy:* It measures the total number of e-mails correctly classified, and is given by Eq. (37).

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{37}$$

2. *Precision:* The precision with respect to spam e-mails, is the ratio of the number of e-mails that are correctly predicted to be in spam to the total number of e-mails predicted to be spam. It is computed using Eq. (38).

$$Precision = \frac{TP}{TP + FP} \tag{38}$$

3. *Recall:* The recall for spam e-mails, given by Eq. (39), is the ratio of the number of e-mails that are correctly predicted to be spam to the total number of e-mails actually belonging to the spam class.

$$Recall = \frac{TP}{TP + FN} \tag{39}$$

4. *F1-measure:* It is the harmonic mean of precision and recall. It is computed as shown in Eq. (40).

$$F1\ Measure = \frac{2.Precision.Recall}{Precision + Recall} \tag{40}$$

Among these, accuracy and *F*1 measure are considered more important, because accuracy measures the total number of e-mails correctly classified, and *F*1 is a combination of spam recall and spam precision.

Tables 2, 3, and 4, illustrate the resultant mean and variance values of different evaluation measures for the distance metrics Euclidean, Manhattan, and Chebyshev respectively. These values are computed for all discussed algorithms using 10-fold cross validation. Table 5 records the average value of each evaluation measure across the three distance metrics used in this study, for all the proposed models.

Table 6 describes the confusion matrix employed in this study and Table 7 illustrates the values of TN, TP, FN, and FP achieved for each algorithm across different distance metrics.

5. Discussion

In this era of communication and connectivity through the Internet, spamming using e-mails has become the most exploited medium by marketers and scammers. Spam e-mails are propagated by such entities for reasons ranging from marketing purposes to malicious activities. With increasing technological developments at pace with Web 2.0, a number of machine learning based models have been created and are being utilized to curb the problem of spamming. This study explores one such area of machine learning in the form of BIC techniques and their application in optimization of classification of e-mails into spam and legitimate. It also attempts to answer questions regarding the performance of different distance metrics generally used in classification and their implications in the spam detection field when used with k-NN classifier.

The first research question in this study inquires the performance of different distance metrics used by k-NN for establishment of class representatives. Through the results obtained by proposed models, it is observed that all algorithms show a 100% value for accuracy of classification, precision, recall and *F*1-measure when Manhattan Distance is used for computation of class representatives as shown in Table 3. The second best results for evaluation measures are obtained by use of Euclidean distance as recorded in Table 2. Chebyshev distance is observed to be the worst performing distance metric used in this study, regardless of optimization algorithm and across all evaluation measures except when it comes to computation of recall, as shown in Table 4, because its

Table 4
Chebyshev distance metric used in k-NN.

Algorithm	Accuracy (%)		Precision (%)		Recall (%)		*F*1-measure (%)		Time (s)
	Mean	Variance	Mean	Variance	Mean	Variance	Mean	Variance	Total
GWO	43.75	2.31	41.2	1.84	100	0	58.34	1.85	665
CSO	43.51	0.93	41.09	1.29	100	0	58.24	1.29	100
FOA	43.77	0.5	41.2	1.19	100	0	58.35	1.19	41,746
GOA	59.59	3.87	19.93	174.5	0.61	0.06	1.18	0.23	22,856
WOA	43.29	1.19	41	1.5	100	0	58.14	1.52	99

Table 5
Average value measures of evaluation.

Algorithm	Accuracy (%)	Precision (%)	Recall (%)	*F*1-measure (%)	Time (s)
GWO	65.64	62.31	100	73.7	681.33
CSO	65.95	62.5	100	73.86	105.67
FOA	65.35	62.16	100	73.55	41747.67
GOA	74	67.39	35.44	37.15	21628.33
WOA	70.58	69.83	94.34	74.46	99.67

Table 6
Confusion matrix.

		Predicted	
		Legitimate (LE)	Spam (SE)
TRUE	**Legitimate (LE)**	True Negative (TN)	False Positive (FP)
	Spam (SE)	False Negative (FN)	True Positive (TP)

ability to compute recall is 100%. The computation time for obtaining results becomes essential in such computationally intensive problems for the purpose of establishing functionally suitable results. As shown in Fig. 2, the computation time of the three distance metrics are comparable and depend on the algorithm being used for optimization.

The second research question introduced in this study seeks to identify the algorithm best suited for optimizing the classification of e-mails among GWO, CSO, FOA, GOA and WOA, when integrated with k-NN. In terms of computation time, as illustrated in Table 5, WOA takes the least amount of time to provide a solution, with an average time of 99.67 seconds, and FOA takes the most amount of time with an average of 41747.67 s. When it comes to accuracy, GOA shows best result among the proposed models, followed by WOA, which is also observed to have the highest value for precision and *F*1-measure. A 100% recall is observed when CSO, GWO or FOA is applied for optimization. These three algorithms also have comparable values for accuracy, precision and *F*1-measure, but they differ highly in computation time.

In terms of converging to a globally optimum solution, Fig. 3 shows the trends followed by different proposed models with respect to the

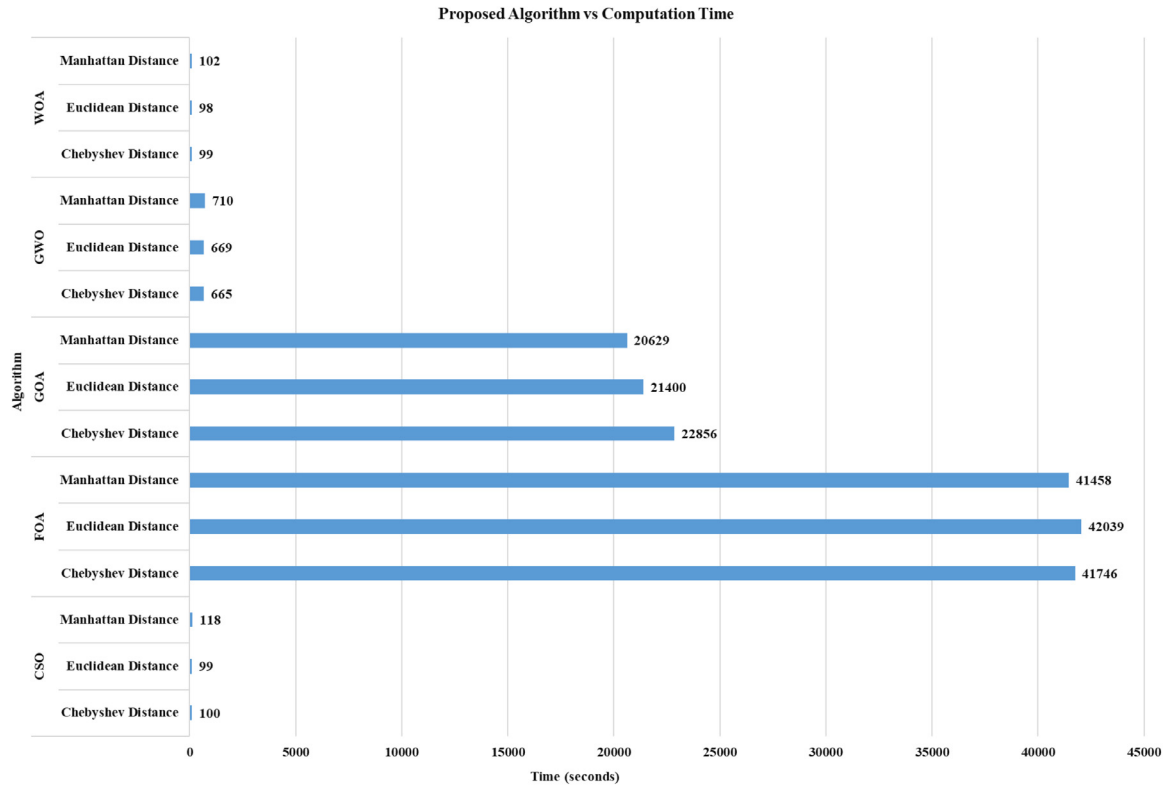

Fig. 2. Proposed models and their corresponding computation time.

Table 7
Values in the confusion matrix for proposed models.

Algorithm			Chebyshev		Euclidean		Manhattan	
			Predicted					
			LE	**SE**	**LE**	**SE**	**LE**	**SE**
GWO	**True**	**LE**	200	2589	634	2155	2789	0
		SE	0	1814	0	1814	0	1814
CSO		**LE**	189	2600	688	2101	2789	0
		SE	0	1814	0	1814	0	1814
FOA		**LE**	201	2588	593	2196	2789	0
		SE	0	1814	0	1814	0	1814
GOA		**LE**	2732	57	2772	17	2789	0
		SE	1803	11	1712	102	0	1814
WOA		**LE**	179	2610	1655	1134	2789	0
		SE	0	1814	318	1496	0	1814

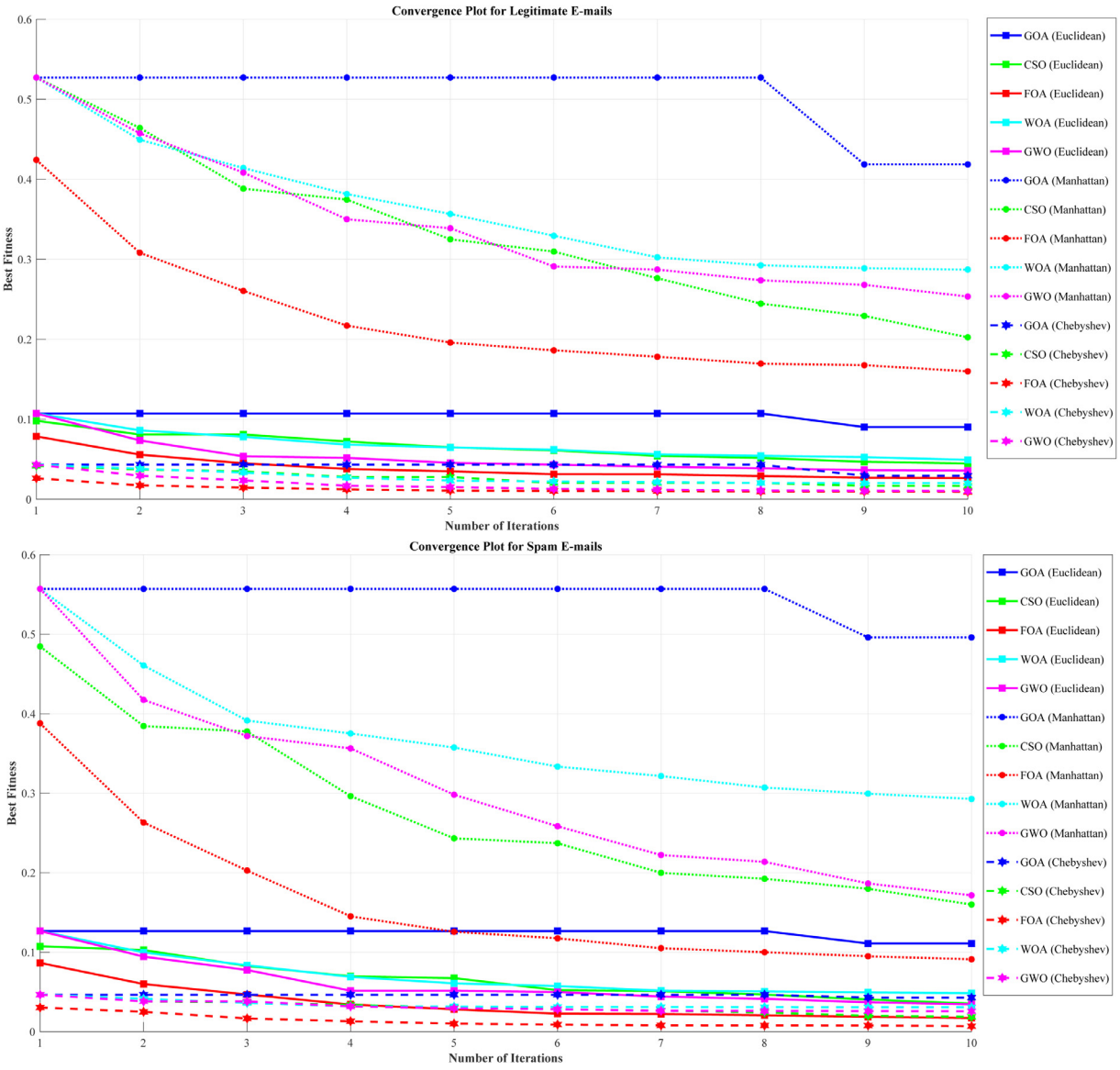

Fig. 3. Convergence plots of proposed algorithms.

distance metric used. It can be seen that FOA shows the quickest convergence to globally optimal solution as compared to other algorithms regardless of the distance metric used and has the same observation for both classes of e-mails. On the other hand, GOA has the slowest convergence plot for both observations discussed for FOA. The other algorithms namely, WOA, CSO, and GOA, have similar convergence plots but their trends are highly distinguishable when Manhattan distance is used and least distinguishable when Chebyshev distance is used.

6. Contribution to literature

A number of studies have been conducted for detection of spam e-mails in literature. BIC optimization algorithms, however, have not found much application in this domain. Similarly, even though a number of existing studies compare different classification algorithms to observe their behavior, very few compare the effect of the distance metrics used to carry out the classification.

This study contributes to literature by discussing the impact of different distance measures, when used for seeking solutions to classification problems such as the one discussed in this study. The performance of Euclidean, Manhattan, and Chebyshev distance metrics are discussed on the basis of various evaluation parameters which provides insights into their abilities as well as a ground for comparison when applied on the same problem.

Further, it presents observations into the performance of BIC optimization algorithms when applied for classification of spam e-mails to optimize results of k-NN. These meta-heuristic algorithms, namely GWO, CSO, FOA, GOA, and WOA, have not been used extensively for optimization of results in the field of applications revolving around spam detection.

7. Implications to practice

This study provides an understanding of results to be expected for each individual combination of the three distance metrics used in k-NN and the five BIC optimization algorithms applied. The performance of BIC algorithms have been differentiated based on various evaluation parameters when optimizing the solution of the same classification problem. Therefore, implications to practice are based on the requirement of future studies and their specific optimization problem while using the meta-heuristic models proposed in this paper.

8. Conclusion

Separation of spam e-mails from legitimate e-mails is a classification problem for which meta-heuristic solutions are proposed in this study in the form of BIC algorithms integrated with k-NN. The results obtained show 100% mean values for accuracy, precision, recall and *F*1-measure when Manhattan distance is used for classification in k-NN, which makes it the most suitable distance metric for classification purposes. The No Free Lunch (NFL) Theorem (Wolpert and Macready, 1997) proposes lack of suitability of a specific meta-heuristic approach for solving all the issues identified in different fields of optimization. This theorem is reinforced in the observations of this study because for the specific case of classification of e-mails into legitimate and spam, the study observes that the different models proposed have different results for different evaluation measures while optimizing the solution of same problem and on the same data set. GOA provides the highest average accuracy of 74%, FOA converges to global optimal solution the fastest, WOA provides the highest average value of precision (69.83%) and *F*1-measure (74.46%), and takes the least computation time (99.67 s), followed by CSO (105.67 s) and GWO (681.33 s), both of which have a recall of 100%.

Therefore, it can be concluded that a specific meta-heuristic approach cannot be considered best among all the proposed models and it is more likely to show better results on a particular evaluation parameter instead of all of them. However, based on the results obtained in this study, WOA can be considered a good model for the classification of e-mails because it shows promising results in majority of the evaluation parameters.

9. Future scope

This study evaluates the performance of different BIC techniques in optimizing the classification of e-mails as spam or legitimate using k-NN. This machine learning approach, when tested on a supervised data set, provides an understanding of trends followed by different distance metrics used in classification as well as trends followed by five different BIC techniques. Since, the meta-heuristic algorithms have been compared on various evaluation parameters when optimizing the solution of the same classification problem, their employment in future endeavors revolving around spam detection will be more streamlined and generate better results.

Since, single-algorithm spam-detection models have been explored in a number of literature, there is a lot of potential in research of hybrid and multi-algorithm systems. The effect on results obtained in this paper can also be explored with a change in the classification algorithm from k-NN to SVM, Naive Bayes, and other such commonly used classifiers. The modified versions of the BIC algorithms already invented in literature can also be employed to improve the results. Methodologically, other meta-heuristic approaches that provide faster convergence to globally optimal solution in a multidimensional problem domain, such as the one used in this study, may be explored for obtaining better results of different evaluation measures.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Supplementary material

Supplementary material associated with this article can be found, in the online version, at 10.1016/j.jjimei.2020.100006.

References

Aswani, R., Ghrera, S. P., & Chandra, S. (2016). A novel approach to outlier detection using modified grey wolf optimization and k-nearest neighbours algorithm. *Indian Journal of Science and Technology, 9*, 1–8.

Aswani, R., Kar, A. K., & Vigneswara Ilavarasan, P. (2018). Detection of spammers in twitter marketing: A hybrid approach using social media analytics and bio inspired computing. *Information Systems Frontiers, 20*(3), 515–530. 10.1007/s10796-017-9805-8.

Bajaj, S., Garg, N., & Singh, S. K. (2017). A novel user-based spam review detection. *Procedia Computer Science, 122*, 1009–1015. 10.1016/j.procs.2017.11.467. 5th International Conference on Information Technology and Quantitative Management, ITQM 2017

Banu, D. M. N., & Banu, S. (2013). A comprehensive study of phishing attacks. *International Journal of Computer Science and Information Technologies, 4*, 783–786.

Barushka, A., & Hájek, P. (2018). Spam filtering using integrated distribution-based balancing approach and regularized deep neural networks. *Applied Intelligence, 48*, 1–19. 10.1007/s10489-018-1161-y.

Chen, S., Yang, R., Yang, R., Yang, L., Yang, X., Xu, C., Xu, B., Zhang, H., Lu, Y., & Liu, W. (2016). A parameter estimation method for nonlinear systems based on improved boundary chicken swarm optimization. *Discrete Dynamics in Nature and Society*, 1–12. 10.1155/2016/3795961.

Costa, E., Lorena, A., Carvalho, A., & Freitas, A. (2007). A review of performance evaluation measures for hierarchical classifiers. AAAI Workshop – Technical Report, (pp. 1–6).

Deb, S., Gao, X., Tammi, K., Kalita, K., & Mahanta, P. (2020). Recent studies on chicken swarm optimization algorithm: A review (2014–2018). *Artificial Intelligence Review, 53*, 1–29. 10.1007/s10462-019-09718-3.

Deshpande, V. P., Erbacher, R. F., & Harrisr, C. (2007). An evaluation of Naïve Bayesian anti-spam filtering techniques. In *Proceedings of the 2007 IEEE SMC information assurance and security workshop* (pp. 333–340). 10.1109/IAW.2007.381951.

Dewangan, R., Shukla, A., & Godfrey, W. (2019). Three dimensional path planning using grey wolf optimizer for UAVs. *Applied Intelligence, 49*, 1–17. 10.1007/s10489-018-1384-y.

Dhawan, S., & Simran (2018). An enhanced mechanism of spam and category detection using neuro-SVM. *Procedia Computer Science, 132*, 429–436. 10.1016/j.procs.2018.05.156. International Conference on Computational Intelligence and Data Science

Diale, M., Celik, T., & Walt, C. V. D. (2019). Unsupervised feature learning for spam email filtering. *Computers & Electrical Engineering, 74*, 89–104. 10.1016/j.compeleceng.2019.01.004.

Drummond, C., & Holte, R. (2005). Severe class imbalance: Why better algorithms aren't the answer. In *Proceedings of the 16th European conference of machine learning: 3720* (pp. 539–546). 10.1007/11564096_52.

Dutta, S., Ghatak, S., Dey, R., Das, A., & Ghosh, S. (2018). Attribute selection for improving spam classification in online social networks: A rough set theory-based approach. *Social Network Analysis and Mining, 8*, 1–16. 10.1007/s13278-017-0484-8.

Emary, E., Zawbaa, H., Grosan, C., & Hassanien, A. E. (2014). Feature subset selection approach by gray-wolf optimization. *Advances in Intelligent Systems and Computing, 334*, 1–13. 10.1007/978-3-319-13572-4_1.

Etaiwi, W., & Naymat, G. (2017). The impact of applying different preprocessing steps on review spam detection. *Procedia Computer Science, 113*, 273–279. 10.1016/j.procs.2017.08.368.

Faris, H., Al-Zoubi, A. M., Heidari, A. A., Aljarah, I., Mafarja, M., Hassonah, M. A., & Fujita, H. (2019). An intelligent system for spam detection and identification of the most relevant features based on evolutionary random weight networks. *Information Fusion, 48*, 67–83. 10.1016/j.inffus.2018.08.002.

Faris, H., Aljarah, I., & Alqatawna, J. (2015). Optimizing feedforward neural networks using krill herd algorithm for e-mail spam detection. In *Proceedings of the 2015 IEEE Jordan conference on applied electrical engineering and computing technologies (AEECT)* (pp. 1–5). 10.1109/AEECT.2015.7360576.

Feng, Y., Chen, H., Li, T., & Luo, C. (2020). A novel community detection method based on whale optimization algorithm with evolutionary population. *Applied Intelligence*, 1–20. 10.1007/s10489-020-01659-7.

Gunal, S., Ergin, S., Gulmezoglu, M., & Gerek, O. (2006). On feature extraction for spam e-mail detection. *Lecture Notes in Computer Science, 4105*, 635–642. 10.1007/11848035_84.

Guo, W., Liu, T., Dai, F., & Xu, P. (2020). An improved whale optimization algorithm for feature selection. *Computers, Materials & Continua, 62*(1), 337–354. 10.32604/cmc.2020.06411.

Hafez, A. I., Zawbaa, H. M., Emary, E., Mahmoud, H. A., & Hassanien, A. E. (2015). An innovative approach for feature selection based on chicken swarm optimization. In *Proceedings of the 7th IEEE international conference of soft computing and pattern recognition* (pp. 19–24).

Halaseh, R. A., & Alqatawna, J. (2016). Analyzing cybercrimes strategies: The case of phishing attack. In *Proceedings of the 2016 cybersecurity and cyberforensics conference (CCC)* (pp. 82–88). 10.1109/CCC.2016.25.

Hopkins, M., Reeber, E., Forman, G., & Suermondt, J. (1999). *Spambase data set – UCI machine learning repository*. Hewlett-Packard Labs. 1501 Page Mill Rd., Palo Alto, CA 94304

Hu, H., & Wang, G. (2018). Revisiting email spoofing attacks. *CoRR*, 1–16. abs/1801.00853

Inuwa-Dutse, I., Liptrott, M., & Korkontzelos, I. (2018). Detection of spam-posting accounts on twitter. *Neurocomputing, 315*, 496–511. 10.1016/j.neucom.2018.07.044.

Irsalinda, N., Yanto, I. T. R., Chiroma, H., & Herawan, T. (2017). A framework of clustering based on chicken swarm optimization. *Advances in Intelligent Systems and Computing*, 1–8. 10.1007/978-3-319-51281-5_34.

Japkowicz, N., & Stephen, S. (2002). The class imbalance problem: A systematic study. *Intelligent Data Analysis, 6*, 429–449.

Johari, N., Zain, A., Mustaffa, N., & Udin, A. (2013). Firefly algorithm for optimization problem. *Applied Mechanics and Materials, 421*, 512–517. 10.4028/www.scientific.net/AMM.421.512.

Karim, A., Azam, S., Shanmugam, B., Kannoorpatti, K., & Alazab, M. (2019). A comprehensive survey for intelligent spam email detection. *IEEE Access, 7*, 168261–168295. 10.1109/ACCESS.2019.2954791.

Khan, W. A., Hamadneh, N. N., Tilahun, S. L., & Ngnotchouye, J. T. (2016). A review and comparative study of firefly algorithm and its modified versions. In *Optimization algorithms – Methods and applications* (pp. 281–313). Intechopen.

Li, H., Zou, P., Huang, Z., Zeng, C., & Liu, X. (2019). Multimodal optimization using whale optimization algorithm enhanced with local search and niching technique. *Mathematical Biosciences and Engineering, 17*, 1–27. 10.3934/mbe.2020001.

Ling, C. X., & Sheng, V. S. (2010). Class imbalance problem. *Encyclopedia of Machine Learning*, 171-171. 10.1007/978-0-387-30164-8_110.

Lukasik, S., Kowalski, P. A., Charytanowicz, M., & Kulczycki, P. (2017). Data clustering with grasshopper optimization algorithm. In *Proceedings of the 2017 federated conference on computer science and information systems (FEDCSIS)* (pp. 71–74). 10.15439/2017F340.

Luo, J., & Liu, Z. (2019). Novel grey wolf optimization based on modified differential evolution for numerical function optimization. *Applied Intelligence*, 1–19. 10.1007/s10489-019-01521-5.

Mafarja, M., Aljarah, I., Faris, H., Hammouri, A. I., Al-Zoubi, A. M., & Mirjalili, S. (2019). Binary grasshopper optimisation algorithm approaches for feature selection problems. *Expert Systems with Applications, 117*, 267–286. 10.1016/j.eswa.2018.09.015.

Meng, X.-B., Liu, Y., Gao, X., & Zhang, H. (2014). A new bio-inspired algorithm: Chicken swarm optimization. *International Conference in Swarm Intelligence*, 86–94. 10.1007/978-3-319-11857-4_10.

Mirjalili, S., & Lewis, A. (2016). The whale optimization algorithm. *Advances in Engineering Software, 95*, 51–67. 10.1016/j.advengsoft.2016.01.008.

Mirjalili, S., Mirjalili, S. M., & Lewis, A. (2014). Grey wolf optimizer. *Advances in Engineering Software, 69*, 46–61. 10.1016/j.advengsoft.2013.12.007.

Mirjalili, S. Z., Mirjalili, S., Saremi, S., Faris, H., & Aljarah, I. (2018). Grasshopper optimization algorithm for multi-objective optimization problems. *Applied Intelligence, 48*(4), 805–820. 10.1007/s10489-017-1019-8.

Mohammed, H. M., Umar, S. U., & Rashid, T. A. (2019). A systematic and meta-analysis survey of whale optimization algorithm. *Computational Intelligence and Neuroscience, 2019*, 1–25. 10.1155/2019/8718571.

Naem, A. A., Ghali, N. I., & Saleh, A. A. (2018). Antlion optimization and boosting classifier for spam email detection. *Future Computing and Informatics Journal, 3*(2), 436–442. 10.1016/j.fcij.2018.11.006.

Nasiri, J., & Khiyabani, F. M. (2018). A whale optimization algorithm (WOA) approach for clustering. *Cogent Mathematics & Statistics, 5*(1), 1–13. 10.1080/25742558.2018.1483565.

Özgür, L., Gungor, T., & Gurgen, F. (2004). Spam mail detection using artificial neural network and Bayesian filter. In *Proceedings of the international conference on intelligent data engineering and automated learning* (pp. 505–510). 10.1007/978-3-540-28651-6_74.

Pan, X., Xue, L., & Li, R. (2019). A new and efficient firefly algorithm for numerical optimization problems. *Neural Computing and Applications*, 1445–1453. 10.1007/s00521-018-3449-6.

Raad, M., Mohd Yasin, N., Alam, G., Bahaa, B., & Zaidan, A. (2010). Impact of spam advertisement through e-mail: A study to assess the influence of the anti-spam on the e-mail marketing. *African Journal of Business Management, 4*, 2362–2367.

Ranganayakulu, D., & C. , C. (2013). Detecting malicious URLs in e-mail – An implementation. *AASRI Procedia, 4*, 125–131. 10.1016/j.aasri.2013.10.020.

Rathod, S. B., & Pattewar, T. M. (2015). A comparative performance evaluation of content based spam and malicious URL detection in e-mail. In *Proceedings of the 2015 IEEE international conference on computer graphics, vision and information security (CGVIS)* (pp. 49–54). 10.1109/CGVIS.2015.7449891.

Sadek, M. M. A., Hamou, R., & Amine, A. (2019). A new bio inspired technique based on octopods for spam filtering. *Applied Intelligence*, 1–11. 10.1007/s10489-019-01463-y.

Saremi, S., Mirjalili, S., & Lewis, A. (2017). Grasshopper optimisation algorithm: Theory and application. *Advances in Engineering Software, 105*, 30–47. 10.1016/j.advengsoft.2017.01.004.

Saremi, S., Mirjalili, S. Z., & Mirjalili, S. (2015). Evolutionary population dynamics and grey wolf optimizer. *Neural Computing and Applications, 26*, 1257–1263. 10.1007/s00521-014-1806-7.

Sekh, A. A., Dogra, D. P., Kar, S., Roy, P. P., & Prasad, D. K. (2020). ELM-HTM guided bio-inspired unsupervised learning for anomalous trajectory classification. *Cognitive Systems Research, 63*, 30–41. 10.1016/j.cogsys.2020.04.003.

Sharma, P., Sundaram, S., Sharma, M., Sharma, A., & Gupta, D. (2019). Diagnosis of Parkinson's disease using modified grey wolf optimization. *Cognitive Systems Research, 54*, 100–115. 10.1016/j.cogsys.2018.12.002.

Shuaib, M., Abdulhamid, S. M., Adebayo, O. S., Osho, O., Idris, I., Alhassan, J. K., & Rana, N. (2019). Whale optimization algorithm-based email spam feature selection method using rotation forest algorithm for classification. *SN Applied Sciences, 1*, 1–17.

Singh, D. N. (2018). A modified variant of grey wolf optimizer. *Scientia Iranica*, 1450–1466. 10.24200/SCI.2018.50122.1523.

Su, M.-C., Lo, H.-H., & Hsu, F.-H. (2010). A neural tree and its application to spam e-mail detection. *Expert Systems with Applications, 37*(12), 7976–7985. 10.1016/j.eswa.2010.04.038.

Tang, Y., Krasser, S., Judge, P., & Zhang, Y. (2006). Fast and effective spam sender detection with granular SVM on highly imbalanced mail server behavior data. In *Proceedings of the 2006 international conference on collaborative computing: networking, applications and worksharing* (pp. 1–6). 10.1109/COLCOM.2006.361856.

Tseng, C.-Y., Sung, P.-C., & Chen, M.-S. (2011). Cosdes: A collaborative spam detection system with a novel e-mail abstraction scheme. *IEEE Transactions on Knowledge and Data Engineering, 23*(5), 669–682. 10.1109/TKDE.2010.147.

Wang, J., Cheng, Z., Ersoy, O. K., Zhang, M., Sun, K., & Bi, Y. (2019). Improvement and application of chicken swarm optimization for constrained optimization. *IEEE Access, 7*, 58053–58072.

Wang, X.-L., & Cloete, I. (2005). Learning to classify email: A survey. In *Proceedings of the fourth international conference on machine learning and cybernetics: 9* (pp. 5716–5719). 10.1109/ICMLC.2005.1527956.

Wolpert, D. H., & Macready, W. G. (1997). No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation, 1*(1), 67–82.

Zhang, L., Liu, L., Yang, X.-S., & Dai, Y. (2016). A novel hybrid firefly algorithm for global optimization. *PLoS One, 11*, 1–17. 10.1371/journal.pone.0163230.

Zhu, Y., & Tan, Y. (2010). Extracting discriminative information from e-mail for spam detection inspired by immune system. *IEEE Congress on Evolutionary Computation*, 1–7. 10.1109/CEC.2010.5586290.