







Training a Logistic Regression Machine Learning Model for Spam Email Detection Using the Teaching-Learning-Based-Optimization Algorithm

Savia Berrou¹, Khadija Al Kalbani¹, Milos Antonijevic² ,
Miodrag Zivkovic² , Nebojsa Bacanin² , and Bosko Nikolic³ 

¹ Modern College of Business and Science, Muscat, Oman
{20213778,20188465}@mcbs.edu.om

² Singidunum University, Danijelova 32, 11000 Belgrade, Serbia
{mantonijevic,mzivkovic,nbacanin}@singidunum.ac.rs

³ School of Electrical Engineering, Belgrade University,
Bulevar Kralja Aleksandra 73, 11000 Belgrade, Serbia
nbosko@etf.bg.ac.rs

Abstract. Spam and emails have always been intrinsically linked since the creation of the Advanced Research Projects Agency Network, otherwise known as (ARPANET). The latter witnessed, on May 3rd, 1978, the first known spam email to date. Today, spam emails negatively affect the users' productivity and private lives. A significant number of approaches emerged in the past two decades that deal with the spam detection problem, with limited success. Therefore, the current paper presents an intelligent and automated solution to spam email detection using a logistic regression model trained by a teaching-learning-based optimization algorithm. The proposed solution has been tested on two benchmark spam email datasets (CSDMC2010 and TurkishEmail), and evaluated against seven other contending cutting-edge metaheuristics utilized in the same experimental setup. The simulation outcomes without a doubt indicate the superior level of accuracy achieved by the proposed solution.

Keywords: Logistic regression · TLBO algorithm · Spam email detection

1 Introduction

The demand for intelligent and reliable spam email detection methods has exponentially increased in light of the booming phishing email threat. Indeed, current statistical observations imply that around 53.5 percent of the 236 billion emails exchanged daily consist of spam mail only [39]. Moreover, incurred financial damage due to these spam emails is estimated to be around a yearly 20.5 billion dollars [39].

© The Author(s) 2023

N. Bacanin and H. Shaker (Eds.): ICITB 2022, ACSR 104, pp. 306–327, 2023.

https://doi.org/10.2991/978-94-6463-110-4_22

As such, legislations and businesses have a keen interest in creating a cyber-secure environment geared against the threat of spam emails. Nevertheless, despite these aspirations, the innovations introduced by spammers are almost always ahead of the currently available anti-spam methods.

Consequently, the investment in the use of artificial intelligence and machine learning approaches during the development of effective preventive methods and countermeasures to the aforementioned problem is especially relevant since it has enabled the introduction of new, intelligent, and automated malicious spam email detection techniques, thus leading to the objective of the current paper.

Indeed, the latter looks to introduce another intelligent and automated approach to spam email detection via the training of a logistic regression machine learning model using the teaching-learning-based optimization algorithm.

To achieve its purpose, in addition to the ongoing introduction, the current paper presents multiple sections, spanning from the literature review to the conclusion, while including an overview of the logistic regression model and the teaching-learning-based optimization algorithm.

2 Literature Review

The literature review section compiles conceptual data necessary for the in-depth discussion of the topic introduced beforehand. The material used consists of previous research in the domain of intelligent spam email detection methods.

2.1 Spam Emails

Today, email is considered one of the most widely used communication tools for official and unofficial interactions. Nevertheless, the risks of attacks also increased since email becomes an essential medium to transfer sensitive information such as Personal Identifiable Information (PII) which includes individual names, addresses, phone numbers, and so on.

One of the most known risks to emails is Spam. Unfortunately, spam emails can cause direct damage either to a single machine or an entire network which will cost businesses and individuals millions of dollars annually. Eventually, this can lead to severe security risks, internationally and economically. Usually, commercial advertisements, money scams, email spoofing, anti-virus warning, and others are categorized as unwanted messages.

Spam can also be defined as uninvited communication intended to be delivered to an indiscriminate target, directly or indirectly, notwithstanding measures to prevent its delivery [29]. Spam email is used as bait hiding in various formats to harm or steal information from the targeted recipient. Over the years, spam emails became more sophisticated with more hidden payloads.

Spammers are often the ones responsible for sending and preparing spam emails. Spammers can be a person, or a group of people belonging to an organization, sending spam emails either directly or indirectly, where they include

payloads within the spam emails while making sure to update their strategy to bypass any new filtering technologies.

Spam emails can be characterized based on their purpose. Starting with unwanted spam emails, these types of spam emails have recipients that are usually uninterested to read or open them. Advertisement emails are a known type of unwanted spam.

The second type of spam email is called Indiscriminate. These spam emails are often received from someone anonymous, not included in the address book. The spammer generally targets many people where some are known, and others are random targets. However, it is more effective for spammers to send spam in huge bulk than targeted. An indiscriminate spammer can employ identity theft techniques where the name of the sender is known to the target. This technique is used to get the trusted sender's recipient to open and read an indiscriminate message [32].

The third type of spam email is Disingenuous. Usually, spam emails are regularly ignored by receivers and might be blocked by spam email filtration. Hence, disingenuous emails must disguise themselves in a way where the chance that their payload will be delivered and acted upon, is optimized [34]. The main approach to Disingenuous spam emails is to seem legitimate. For example, the recipient might receive an email about a password update or a new patch by Microsoft with an urgent need of downloading.

The final type of spam is spam email with payload. In this case, emails can be identified by the recipient or can be difficult to identify. This depends on the level of awareness of the recipient about the identification and the effect of spam emails. Nevertheless, different mechanisms can be employed to detect such spam emails. Additionally, this final type is designed to attract the recipient's attention and lead them to further interact with the spam email, eventually landing on the spammer's web page.

All in all, spam causes damages that have several common effects, summarized in the following: Direct effects can incur through spam usage in a directed channel to target specific people. The consequence of this type is incredibly broad due to the spammed email being used as a tool to send other spam emails. In this case, Network Resource Consumption can also be considered another consequence.

Today enhanced technology can easily identify and categorize suspicious emails as spam. This will cause the mailbox to be loaded with spam emails. Usually, this type of traffic consumes bandwidth and storage, increasing the risk of delivery delays and the high chance of losing messages through their way [27].

Another critical consequence to consider is human resource interaction. Filtering spam emails from the mailbox and sorting them usually require a huge amount of time and effort. This can be difficult if there is no tool to help with filtration especially if the received emails look legitimate, which requires a lot of time to differentiate between spam and ham email. Finally, the last consequence is lost emails: Receiving a bulk of spam emails along with legitimate emails might cause the recipient to ignore important emails by mistake.

2.2 Machine Learning

In the current ever-evolving technological era, machine learning plays an important role in the development of any new technology. The way that machine learning is enforced completely depends on certain factors such as the algorithm and testing samples used. The applications can range from resolving a network traffic issue to detecting malware, including spam detection. Today, new technology of spam filtration is bullied with machine learning concepts to help accelerate the process of spam detection. Many companies like Gmail, Yahoo, and Outlook are offering different tools and techniques to discern spam emails in a network [73].

Spam filtering can be implemented with other technologies to help identify malware and spam, either in the cloud, network devices (routers, switches), or end-user machines. Very famous known machine learning methods for spam detection are content-based filtering, rule-based filtering, or Bayesian filtering.

Furthermore, before machine learning technology, spam email detection was achieved using Knowledge Engineering, a method that depends on a manual that is used as a reference to identify Spam from Ham emails. The major drawback of this method is the time and effort consumption. Hence, with the use of artificial intelligence (AI), current machine learning is a more sophisticated tool than the knowledge engineering method which makes possible the capacity to learn how to automatically differentiate between Spam (unsolicited) and ham (legitimate) emails, and later be able to apply this learned information to newly received emails.

In feature selection, data mining is usually used by machine learning for detection mechanisms to discover new or existing patterns (or features) from a data set which is then used for classification [43]. Today, Gmail can block more than 99.9 percent of suspected spam emails, phishing, and malware from reaching the recipients' inboxes. This is all due to machine learning detecting spam emails with high accuracy, which improved remarkably over the past years. Ever-changing spam tactics can now be easily detected. This is because machine learning was used to identify patterns in large data sets constituting a huge success.

Machine learning processes usually can be divided into six to seven steps. The variation of the steps involved depends on the model or algorithm used to classify data sets. Data collection is often the first step and is considered important since the second step is highly dependent on it. That is because selecting the features is based on the data set collected. Starting with features like numeric, integer, and so on, feature engineering allows for the creation of valuable input by identifying a feature to help provide an accurate result.

On the other hand, redundancy can sometimes affect the way output will be received. That is why the second step is important since it makes sure that there is no redundancy which is done by cleaning the data set to have an accurate result. Removing unwanted data, missing values, rows, columns, duplicate values, erroneous data type conversion, and more is the method through which cleaning the data is done. Eventually, the cleaned data sets will be divided into two groups: training sets and testing sets. The purpose of training sets is to train the

model. A testing set is used to check the accuracy of the model, or algorithm, after training.

Selecting the model is the third step, which completely depends on the specific objective. Spam recognition, image recognition, sound recognition, and the like can be the objective the model is trying to achieve. Additionally, there are two types of models: supervised and unsupervised. The supervised model involves modeling the trained data to output a data label, whereas the unsupervised model simulates the trained data set without data labeling.

In step number four, the model is trained. Training the model plays a major part since it is crucial to accomplish accurate predictions as output. In this step, the patterns are identified by applying training sets to the machine learning model. The trained model with time will develop its ability to produce more accurate predictions.

Evaluating the model is the fifth step. In this step, the trained model is evaluated once the training is complete. This is done by implementing the testing data into the machine learning model. In this step, the speed and performance of the machine learning model will be determined once the model is trained.

In step number six, the model parameter is tuned for better results. In this step, the performance of the machine learning model can be adjusted by tuning the parameters. Parameters are usually known as the variables related to a specific model.

Making predictions is commonly the last step in machine learning. In this step, the testing data sets are applied to the trained model to find more accurate results.

Nowadays, python programming is mostly used by data analysts and scientists to implement machine learning approaches. Deciding what kind of computing resources is needed for training the models is an important decision to be made by data scientists. Training a model can be achieved by using local computers. However, sometimes the necessary capacity to complete the process of training a sizable data set may not be accomplished by the local computers' processing. In this case, cloud processing units may be used to transition the workload to the cloud environment, where data scientists can have access to a broader selection of computing resources including Graphics Processing Units (GPUs).

2.3 Spam Email Detection Methods

Spam email detection methods using machine learning have been at the forefront of the current research trend. Email-providing entities, such as Gmail, Outlook, and many more companies of the like, have come to grasp effective techniques to diffuse most email spam threats through the usage of spam filtering machine learning methods. These methods are acclaimed for allowing the possibility of systematic bulk analysis, progressive learning, recognition, and discovery of phishing messages and spam mail identifiers. The previously listed processes are achieved through the high adaptability of machine learning algorithms and their ability to

use the learned material to independently generate new rules while still adhering to the pre-established ones.

The following categorization of spam email detection methods comes as a result of the former commentary on machine learning interests:

- **Content-Based Spam Email Detection Method [28]:** Employing the content-based spam email detection method allows for the creation of automated detection rules. During the process of analyzing email content, this method offers the ability to identify the distribution, occurrence, or absence of specific phrases and words. The collected language is then employed in the generation of new spam detection rules for any ensuing email analysis. Furthermore, the Support Vector Machine, the Neural Networks, the K-Nearest Neighbors Algorithm, and the Naive Bayesian Classification are all recognized as examples of machine-learning techniques that can utilize content-based spam detection.
- **Adaptive Spam Email Detection Method [28]:** Through the adaptive spam email detection method, the detection and filtering process triggers the classification of spam emails into various groups, each under a common emblematic text. The content of a new email is then compared to the one of each group, allowing for the computation of a similitude percentage. The latter is subsequently employed to conclude whether the analyzed email is affiliated with any of these specific groups.
- **Previous Likeness-Based Spam Email Detection Method [28]:** According to the previous likeness-based spam email detection method, memory-based, or instance-based machine learning techniques are employed to enable the usage of resemblance analysis in the interest of facilitating email categorization according to training emails accumulated in the system. Such a spam email detection method can be utilized by machine learning approaches akin to the K-Nearest Neighbors Algorithm.
- **Heuristic or Rule-Based Spam Email Detection Method [28]:** The heuristic or rule-based spam email detection method enables the bulk evaluation of a substantial number of patterns according to previously generated rules, also known as heuristics. Furthermore, the confirmation of an email as spam is determined through the usage of a scoring system. The latter allows for the attribution of a score to each analyzed email following two rules: An increased score is imputed if any number of patterns correspond to the email's content, whereas a decreased score is assigned if no matching patterns are found. A score surpassing a particular threshold leads to the automatic filtering of the email as spam. Therefore, this approach necessitates a continuous pattern revision to circumvent the threat of obsolescence. As an example, the heuristic or rule-based spam detection method is employed by Apache's Spam Assassin.
- **Case Base Spam Email Detection Method [28]:** The case base or sample base spam email detection method allows email classification via the extraction of non-spam and spam email data that is converted into sets, trained, and tested by machine learning algorithms, to then be utilized in the analysis of future email content.

2.4 Metaheuristics Optimization

Metaheuristics optimization, including swarm intelligence methods, has recently gained popularity within the scientific circles, due to a large number of successful applications and successes in handling NP-hard problems. A significant number of metaheuristics methods exist today, explained by the no free lunch theorem [57], stating that an universal solution to all optimization problems is not existing. Consequently, the researchers have implemented a large number of algorithms, and employed them on a wide range of real-world problems from different domains, such as medical diagnostics [15,21,25,37,47], wireless sensor networks [5,10,13,53,63,72], stock price forecasting [17], intrusion detection and other security applications [2,35,45,51,61,62,66,71] and plant classification problem [18].

Metaheuristics algorithms have been also employed to tune the cloud, edge and fog computing [3,6,16,24,52,65], feature selection [9,20,23,36,41,54,67], dropout regularization [12], a wide spectrum of COVID-19 challenges [26,64,68–70], artificial neural networks optimization [4,7,8,11,14,19,50], text clustering [22,55] and cryptocurrencies predictions [46].

3 Logistic Regression Model

Nowadays, modern researchers and scientists are widely depending on the logistic regression model for data analysis. The logistic regression model is implemented as a built-in feature for products produced by many software development companies. Variable categorization is commonly used by the logistic regression model. For example, since it uses binary as the categorization criteria, it will predict whether the received email is spam (1) or ham (0) since it can take one of the two binary values as output. This is known as binary classification. Multinomial is another way of classification where models with data sets can have more than two possible outputs.

The logistic regression model is a useful analysis tool, especially for attack detection since it is considered as a classifier rather than a pure regression. The mathematics of logistic regression depends on the “odds” of an event, which is the probability of the event occurring divided by the probability of it not happening. Like linear regression, logistic regression predicts the value of input data (e.g., whether an email is a ham or spam). In contrast to logistic regression, linear regression will predict less accurate results when compared to logistic regression [1].

A relationship between several features and variables is produced by the logistic regression model for input. Like linear regression, logistic regression is suitable when the variable being predicted is a probability on a binary range from 0 to 1. Nevertheless, if linear regression with a binary range from 0 to 1 is used, it will not work properly since the independent variable values are constrained by 0 and 1, where movement beyond the dependent values provided in the sample data set could produce impossible results (below 0 or above 1). A probability curve on a binary scale must therefore be sigmoid-shaped (s-shaped)

and mathematically constrained between 0 and 1, which the logistic regression model provides [40].

In the logistic regression model, the curve of its probability is compared to the perfectly straight line in linear regression. To find the distance in the curve, the dataset used in the logistic regression model must be transformed. The distance can be achieved by testing different parameters and variables of data sets by converting the probabilities to odds, to create a logistic function from the odds to maximize the likelihood estimation (MLE) to help find accurate results.

Binary Logistic Regression, Multinomial Logistic Regression, and Ordinal Logistic Regression are the three types of logistic regression. In the case of an email, only two possible outputs such as spam or ham can be identified by Binary Logistic Regression as a result. Whereas more than two non-ordered results such as predicting what car a person prefers (Van, SUV, or Minivan) can be identified by Multinomial Logistic Regression. Finally, there are more than two possible results that are ordered such as rating best school programs from 1 to 10 in Ordinal Logistic Regression.

4 Teaching-Learning-Based-Optimization Algorithm

An optimization algorithm is a process through which, given specific parameters and a specific system, an optimum value is identified. This optimum value, referring to either a minimum or a maximum, fulfills all the required conditions of the optimization problem in question.

In the case of the Teaching-Learning-Based-Optimization (TLBO) Algorithm, the objective is to use the meta-heuristic, population-based, optimization procedure to simulate a typical learning environment. Indeed, while mimicking classroom circumstances, the TLBO algorithm chooses, from a class of students, also known as learners, the highest learner as the teacher. The learners gradually improve their knowledge thanks to the teacher's input [30]. Therefore, teachers and learners constitute two essential elements of the aforementioned algorithm.

Conversely, a class of learners can be defined as P , standing for Population, and following $P_G = [X_{1,G}, X_{2,G}, \dots, X_{N_p,G}]$ where G refers to the generation in question, $X_{i,G}$ the vector of the individual number i , and N_p , the population size [58]. Knowing the dimension of subjects as D , $X_{i,G} = [x_{1i,G}, x_{2i,G}, \dots, x_{Di,G}]^T$ defines each vector $X_{i,G}$ ($i = 1, 2, \dots, N_p$) as an individual of generation G [58].

At initialization, the Eq. (1):

$$P_{i,j}^0 = P_j^{min} + rand * (P_j^{max} - P_j^{min}) \quad (1)$$

randomly initializes a population as P^0 [42]. In this case, the maximum value is represented by P_j^{max} while the minimum value is P_j^{min} , in addition to a (0, 1) ranged uniformly distributed random variable, known as *rand* [42].

These definitions lead to the observation that no algorithm-specific parameters are required in the TLBO algorithm, distinguishing the latter from most of its counterparts, while allowing for less complexity and more proper tuning accessibility. Furthermore, as previously underlined, the number of generations and the population size are the only common controlling parameters needed for this specific procedure.

Overall, the TLBO algorithm entails a Teacher Phase, and a Learner Phase, as two modes of learning. The following introduces a comprehensive discussion of these particular modes.

4.1 TLBO Algorithm: Teacher Phase

The teacher phase has the objective of upgrading the knowledge of learners to an amount equivalent to that of a teacher [48]. Nevertheless, the practice shows the role of a teacher as a knowledge imparter, who, instead of focusing on the performance of a single learner, looks to increase the average result of a population of learners. Furthermore, the knowledge-sharing process is facilitated by teacher-learner interactions.

Consequently, the teacher, designated as $X_{t,G}$ refers to the learner with the best fitness among a population of learners, in a specific generation G [58]. A teacher phase vector $V_{i,G} = [v_{1i,G}, v_{2i,G}, \dots, v_{Di,G}]^T$ is generated thanks to the following Eq. (2):

$$V_{i,G} = X_{i,G} + r_i(X_{t,G} - T_F M_G) \quad (2)$$

where a random value $r_i \in (0, 1)$, $i = 1, 2, \dots, N_p$, a mean vector of all the individuals in the population of generation G , M_G , and a learning weight, valued at either 1 or 2, $T_F = \text{round}[1 + \text{rand}(0, 1)\{2 - 1\}]$, are included [58]. This equation allows for a new solution to be generated according to the mean of the population and the previous best solution.

Following this process, $G + 1$ is introduced as the next iteration of the generation G , alongside new individuals $X_{i,G+1}$ ($i = 1, 2, \dots, N_p$) of the population P_{G+1} , selected via the following Eq. (3):

$$X_{i,G+1} = \begin{cases} X_{i,G}, & \text{if } f(X_{i,G}) \leq f(V_{i,G}), \\ V_{i,G}, & \text{otherwise} \end{cases} \quad (3)$$

where the fitness function is identified by $f(\cdot)$ [58]. As such, greedy selection refers to the process of the above-mentioned equation.

4.2 TLBO Algorithm: Learner Phase

During the learner phase, learners interact among themselves, as well as with the teacher, thus enhancing their knowledge [48]. Interactions among learners are made possible thanks to demonstrations, group deliberations, and further methods of the like. A more knowledgeable learner helps upgrade the knowledge of another learner when interacting with the latter. Such a result is achieved

thanks to each learner comparing their knowledge with the other learners of their class. The outcome of the comparison leads to additional, rectified, or unchanged knowledge in the grasp of the involved learners.

Thus, any specific learner is identified via a learner phase vector $U_{i,G} = [u_{1i,G}, u_{2i,G}, \dots, u_{Di,G}]^T$, according to the subsequent Eq. (4):

$$U_{i,G} = \begin{cases} X_{m,G} + r_m(X_{m,G} - X_{n,G}) & \text{if } f(X_{m,G}) < f(X_{n,G}), \\ X_{m,G} + r_m(X_{m,G} + X_{n,G}) & \text{otherwise} \end{cases} \quad (4)$$

where $m \neq n$, a random value $r_m \in (0, 1)$, and two individuals obtained from a random selection in the population of generation G , $X_{m,G}$ and $X_{n,G}$, are observed [58]. This equation allows for the generation of a new solution via the assistance of other solutions considered as partners.

Sequential to the previous procedure, $G+1$ is presented as the next iteration of the generation G , alongside new individuals $X_{i,G+1}$ ($i = 1, 2, \dots, N_p$) of the population P_{G+1} , selected via the following Eq. (5):

$$X_{i,G+1} = \begin{cases} X_{i,G} & \text{if } f(X_{i,G}) \leq f(U_{i,G}), \\ U_{i,G} & \text{otherwise} \end{cases} \quad (5)$$

where the fitness function is known as $f(\cdot)$ [58]. Akin to the teacher phase, this equation allows for greedy selection as well.

4.3 TLBO Algorithm: Procedure

The TLBO algorithm, seen in the following Algorithm 1, introduces a procedure with a beginning characterized by the initialization of the number of iterations, which refer to the number of generations, and the population size, thus demonstrating the algorithm's need for only 2 parameters.

At the start of any given iteration, the best solution is found and set as the new teacher. Then, the algorithm proceeds to calculate the mean of the decision variables and generate a new solution and fitness.

The application of greedy selection leads to the identification of a new partner that interacts with the existing individual, thus triggering the generation of a new solution and fitness. The current iteration is concluded with an additional application of greedy selection to enable the memorization of the best solution, thus leading to the start of the next iteration.

Algorithm 1. The Teaching-Learning-Based-Optimization Algorithm [58]

```

Initialize population;
Set population bounds;
Generate a random initial population  $P_0$ ;
Set  $G = 0, F = 0$ ;
while  $G < G_{Max} \vee F < F_{Max}$  do
  for  $i = 1; i \leq N_p; i++$  do
    Select the teacher  $X_{t,G}$ ;
    Calculate the mean vector  $M_G$ ;
    Implement the teacher learning law according to Eq. (2);
    Check the bounds;
    Update the population according to Eq. (3);
  end for
   $G++, F = F + N_p$ ;
  for  $i = 1; i \leq N_p; i++$  do
    Select two random individuals  $X_{m,G}$  and  $X_{n,G}$ , where  $m \neq n$ ;
    Implement the learner learning law according to Eq. (4);
    Check the bounds;
    Update the population according to Eq. (5);
  end for
   $G++, F = F + N_p$ ;
end while

```

4.4 Improved TLBO Metaheuristics

A simple but effective alteration to the basic variant of TLBO was added with a goal to improve the converging capacity of the algorithm. Every individual in the populace is coupled with a *trial* parameter, that starts from the value 0, and it is increased after each round of execution. If the observed individual was not improved after k rounds elapsed, it is eliminated from the populace, and a novel, randomly produced individual will take its place. The value k is determined as $maxIter/N$, where N denotes number of solutions, and $maxIter$ represents number of iterations in a single run. Accordingly, in the experiments, k was fixed to 12 (500 iterations and 40 solutions). Altered algorithm has been simply given the name improved TLBO-ITLBO.

5 Experimental Setup and Comparative Analysis of the Results

5.1 Dataset Details and Basic Exploratory Data Analysis

This paper utilizes two public benchmark spam email datasets to evaluate the effectiveness of the proposed approach, namely the CSDMC2010 dataset (obtainable from <https://github.com/zrz1996/Spam-Email-Classifier-DataSet>) and TurkishEmail dataset, described in [31]. The count of distinct terms in both datasets shows that there exist 82 148 and 25 650 distinctive terms, respectively.

Table 1. Frequency statistical results of the observed datasets.

Property/dataset	CSDMC2010	TurkishEmail
len (in tokens)	1 574 504	289 598
len (distinctive tokens)	90 392	46 529
len (without stopwords)	47 385	30 808
len (tokens of the alphabetic characters strings)	47533	30861
len (stemming)	35 652	20 195
len (dictionary tfidf for vectorization)	35 617	20 138
Shape (dataframe)	(4 327, 35 617)	(826, 20 138)

Further conducted exploratory analysis shows that the imbalance ratio of the observed CSDMC2010 and TurkishEmail datasets are 2.14 and 1. The imbalance ratio is calculated by dividing the count of regular messages by the count of messages considered as spam. This analysis indicates that the CSDMC2010 is imbalanced, while the TurkishEmail dataset is balanced. In terms of sparsity, that is determined with the feature vector size of 1000, CSDMC2010 has sparsity of 90.48%, while the TurkishEmail obtained the result of 90.02%, concluding that both datasets can be considered as scarce. The basic frequency statistical analysis of the trained bundles of both datasets is presented in Table 1.

PorterStemmer was utilized for the CSDMC2010 data, as a well-known common python option for this kind of content. On the other hand, TurkishStemmer (obtainable from <https://kandi.openweaver.com/python/otuncelli/turkish-stemmer-python#Summary>) was used for TurkishEmail data. Both employed stemmers are popular and common choice for natural language processing use cases.

Further on, the fraction of borderline points test was utilized to assess the complexity. This particular test was suggested by Friedman to determine whether or not two multivariate instances belong to the same distribution. As two opposite classes within the training data are connected, certain amount of points percentage is associated, and then normalized. The final score near 0 suggest that the observed data are separable, while the outcome close to 1 indicates that data are not separable. This metric is susceptible to both imbalanced dataset and classes' separability.

5.2 Experimental Setup

The simulation environment that was used to conduct the LR tests is explained within this section, followed by the discussion of the experimental outcomes. Every solution of the ITLBO is consisting of the LR coefficients and intercept value, therefore every individual length D is obtained by $D = nf + 1$, where value nf denotes the overall amount of features (coefficient and intercept values). The experiments were conducted for both English (CSDMC2010) and TurkishEmail

datasets, with 500 features. The coefficients boundaries were determined empirically and set to $[-6, 5]$, that is different than the values used in cited paper [29] ($[-8, 8]$). The intercept boundaries were also obtained through experiments and trial and error technique, and determined values of $[0, 1]$ were used for calibration of both observed sets. This scenario assumes that every variable is continuous type.

With the goal of this paper being the train process of the LR model, the hyperparameters' values of the LR (for instance, regularization, C and so on) were set to defaults as defined in the Python scikit-learn package. Every metaheuristic approach included in this research was employed with 40 solutions in the population, while the maximal number of rounds was set to $maxIter = 500$ per run, with a total of 15 independent runs of each algorithm.

Fitness calculation of the every solution in the population (such as classification error metrics) is obtained during the training process. The best solution (that is determined to have the best fitness value over the training data) is then verified on the testing set, after all rounds in a run are completed, serving as the final outcome of the run. 10% of both datasets were used as the testing data, while 90% was used to train the LR model.

The efficacy of the proposed ITLBO method with respect to the converging speed and overall optimization capability was put into the comparisons with the results of seven other cutting-edge metaheuristic algorithms tested in the identical simulation setup. Other contending algorithms were namely the original implementation of TLBO ([49]), ABC ([38]), FA ([59]), BA ([60]), HHO ([33]), SNS ([56]) and SCA ([44]). The control parameter configuration for these rival methods was retrieved from the original publications, while the authors have independently implemented these algorithms for the sake of this research, and employed them to execute the LR training process. With a goal of easier tracking of the simulation outcomes, each algorithm was associated with LR prefix (such as LR-ITLBO, LR-FA, and so forth).

5.3 Obtained Experimental Results and Comparative Analysis

The aggregate results of the LR simulations performed over the English 500 dataset with all observed methods are given in Table 2. It can be observed that the LR-ITLBO method attained the overall second best result (after LR-SCA), however, the LR-ITLBO obtained the best median and mean metrics. Table 3 shows the detailed measurements of the best run of each algorithm. It can be noted that LR-ITLBO and LR-SCA attained almost identical accuracy (slightly in favor of LR-SCA), with also LR-ITLBO almost identical values for recall and precision.

The aggregate results of the LR simulations performed over the Turkish 500 dataset with all eight contending methods are shown in Table 4. It can be observed that the LR-ITLBO method attained the overall fourth best result (after LR-ABC, LR-HHO and LR-SNS), however, the LR-ITLBO obtained excellent median and mean metrics, and the best result for the worst run, meaning that on average LR-ITLBO is capable of delivering very stable results, even in

Table 2. Overall metrics of all algorithms on English 500 dataset

Method	Best	Worst	Mean	Median	Std	Var
LR-ITLBO	3.00E-02	4.62E-02	3.73E-02	3.58E-02	5.24E-03	2.74E-05
LR-TLBO	3.00E-02	5.31E-02	3.81E-02	3.70E-02	7.51E-03	5.64E-05
LR-ABC	3.46E-02	4.39E-02	4.08E-02	4.16E-02	3.17E-03	1.01E-05
LR-FA	3.46E-02	4.62E-02	4.20E-02	4.27E-02	3.63E-03	1.32E-05
LR-BA	3.23E-02	4.85E-02	4.12E-02	4.39E-02	5.72E-03	3.27E-05
LR-HHO	3.70E-02	4.39E-02	4.08E-02	4.16E-02	2.88E-03	8.30E-06
LR-SNS	3.46E-02	4.85E-02	4.04E-02	3.81E-02	5.12E-03	2.62E-05
LR-SCA	2.77E-02	3.93E-02	3.73E-02	3.93E-02	4.30E-03	1.85E-05

Table 3. Detailed metrics of best runs of all algorithms on English 500 dataset

	LR-ITLBO	LR-TLBO	LR-ABC	LR-FA	LR-BA	LR-HHO	LR-SNS	LR-SCA
Accuracy (%)	97.1540	96.9977	96.5358	96.5358	96.7667	96.3048	96.5358	97.2286
Precision 0	0.976351	0.966887	0.966667	0.963576	0.966777	0.966555	0.966667	0.982935
Precision 1	0.956204	0.977099	0.962406	0.969466	0.969697	0.955224	0.962406	0.950000
M.Avg. Precision	0.970562	0.970142	0.965309	0.965453	0.967708	0.962944	0.965309	0.972438
Recall 0	0.979661	0.989831	0.983051	0.986441	0.986441	0.979661	0.983051	0.976271
Recall 1	0.949275	0.927536	0.927536	0.92029	0.927536	0.927536	0.927536	0.963768
M.Avg. Recall	0.969977	0.969977	0.965358	0.965358	0.967667	0.963048	0.965358	0.972286
F1-score 0	0.978003	0.978224	0.97479	0.974874	0.97651	0.973064	0.97479	0.979592
F1-score 1	0.952727	0.951673	0.944649	0.944238	0.948148	0.941176	0.944649	0.956835
M.Avg. F1-score	0.969948	0.969762	0.965184	0.96511	0.967471	0.962901	0.965184	0.972339

Table 4. Overall metrics of all algorithms on Turkish 500 dataset

Method	Best	Worst	Mean	Median	Std	Var
LR-ITLBO	2.41E-02	4.82E-02	3.61E-02	3.61E-02	9.84E-03	9.68E-05
LR-TLBO	2.41E-02	6.02E-02	4.22E-02	4.22E-02	1.15E-02	1.33E-04
LR-ABC	1.20E-02	6.02E-02	3.21E-02	3.01E-02	1.50E-02	2.26E-04
LR-FA	2.41E-02	7.23E-02	4.62E-02	4.82E-02	1.46E-02	2.14E-04
LR-BA	2.41E-02	4.82E-02	3.61E-02	3.61E-02	6.96E-03	4.84E-05
LR-HHO	1.20E-02	4.82E-02	2.81E-02	3.01E-02	1.33E-02	1.77E-04
LR-SNS	1.20E-02	4.82E-02	3.21E-02	3.61E-02	1.14E-02	1.29E-04
LR-SCA	2.41E-02	4.82E-02	3.41E-02	3.61E-02	8.28E-03	6.85E-05

the worst runs. Table 5 shows the detailed measurements of the best run of each algorithm. It can be noted that LR-ITLBO obtained very high level of accuracy of around 98.1%.

To provide visual representation of the LR-ITLBO capabilities against the contending algorithms, the convergence graphs and box plot graphics for the classification error rate and objective function are given in Fig. 1 for English dataset (500 features), and in Fig. 2 for TurkishEmail dataset (also 500 features), respectively.

Table 5. Detailed metrics of best runs of all algorithms on Turkish 500 dataset

	LR-ITLBO	LR-TLBO	LR-ABC	LR-FA	LR-BA	LR-HHO	LR-SNS	LR-SCA
Accuracy (%)	97.5904	97.5904	98.7952	97.5904	97.5904	98.7952	98.7952	97.5904
Precision 0	1	0.961538	1	0.961538	1	1	1	0.961538
Precision 1	0.942857	1	0.970588	1	0.942857	0.970588	0.970588	1
M.Avg. Precision	0.977281	0.97683	0.988306	0.97683	0.977281	0.988306	0.988306	0.97683
Recall 0	0.96	1	0.98	1	0.96	0.98	0.98	1
Recall 1	1	0.939394	1	0.939394	1	1	1	0.939394
M.Avg. Recall	0.975904	0.975904	0.987952	0.975904	0.975904	0.987952	0.987952	0.975904
F1-score 0	0.979592	0.980392	0.989899	0.980392	0.979592	0.989899	0.989899	0.980392
F1-score 1	0.970588	0.96875	0.985075	0.96875	0.970588	0.985075	0.985075	0.96875
M.Avg. F1-score	0.976012	0.975763	0.987981	0.975763	0.976012	0.987981	0.987981	0.975763

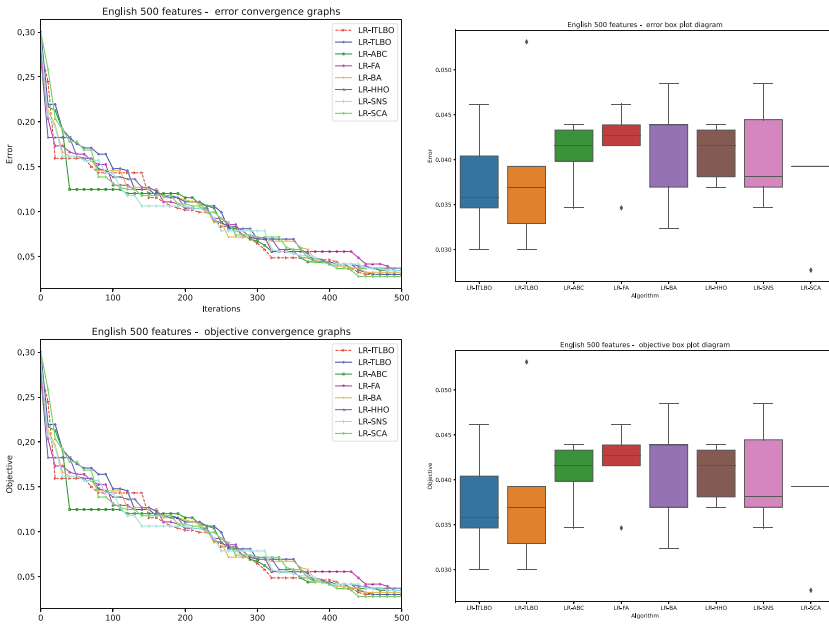


Fig. 1. Convergence diagrams and box plots for error rate and objective function of all contending algorithms on English 500 dataset

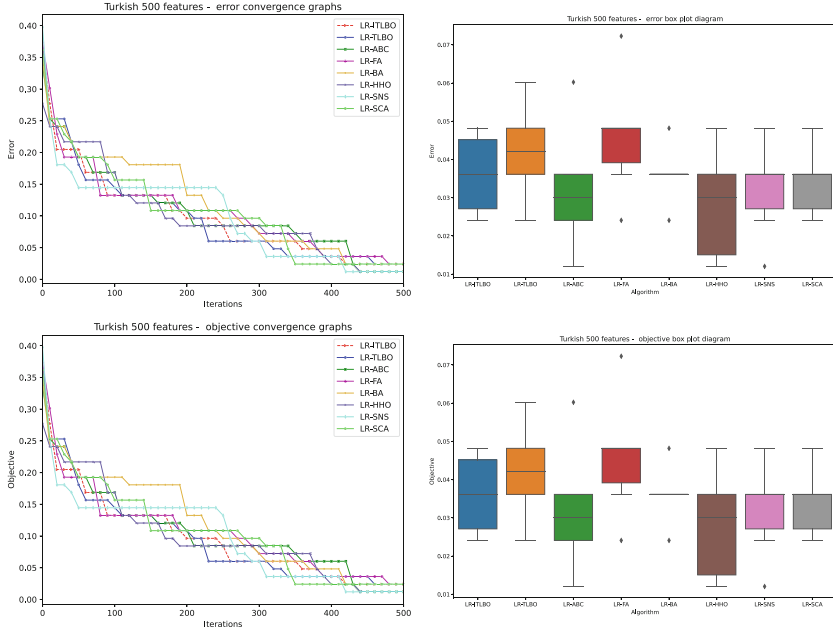


Fig. 2. Convergence diagrams and box plots for error rate and objective function of all contending algorithms on TurkishEmail 500 dataset

6 Conclusion

The research given in this paper proposes an enhanced variant of the TLBO metaheuristics algorithm, that targets the known deficiencies of the original implementation. The devised approach was named ITLBO, and it was employed in the machine learning simulation environment, to perform the task of LR training. The proposed model was named LR-ITLBO, and was verified against two benchmark spam email datasets - CSDMC2010 and TurkishEmail datasets.

To assess the performance of the LR-ITLBO, seven contending metaheuristics algorithms were employed in the same simulation conditions, and put into the comparative analysis. The experimental findings unarguably suggest that the proposed LR-ITLBO approach achieved very high level of accuracy on both English and Turkish datasets.

Before consideration of the practical implementation of the proposed approach in the real-world cybersecurity applications and other domains that include protecting the email services, additional experiments will be executed with more real-world email datasets to enhance the credibility of the approach even further.

References

1. Ahmed, N., Amin, R., Aldabbas, H., Koundal, D., Alouffi, B., Shah, T.: Machine learning techniques for spam detection in email and iot platforms: analysis and research challenges. *Security and Communication Networks* **2022** (2022)
2. AlHosni, N., Jovanovic, L., Antonijevec, M., Bukumira, M., Zivkovic, M., Strumberger, I., Mani, J.P., Bacanin, N.: The xgboost model for network intrusion detection boosted by enhanced sine cosine algorithm. In: *International Conference on Image Processing and Capsule Networks*. pp. 213–228. Springer (2022)
3. Antonijevec, M., Strumberger, I., Lazarevic, S., Bacanin, N., Mladenovic, D., Jovanovic, D.: Robust encrypted face recognition robot based on bit slicing and fourier transform for cloud environments. *Journal of Electronic Imaging* **31**(6), 061808 (2022)
4. Bacanin, N., Alhazmi, K., Zivkovic, M., Venkatachalam, K., Bezdán, T., Nebhen, J.: Training multi-layer perceptron with enhanced brain storm optimization metaheuristics. *Comput. Mater. Contin* **70**, 4199–4215 (2022)
5. Bacanin, N., Antonijevec, M., Bezdán, T., Zivkovic, M., Rashid, T.A.: Wireless sensor networks localization by improved whale optimization algorithm. In: *Proceedings of 2nd International Conference on Artificial Intelligence: Advances and Applications*. pp. 769–783. Springer (2022)
6. Bacanin, N., Antonijevec, M., Bezdán, T., Zivkovic, M., Venkatachalam, K., Malebary, S.: Energy efficient offloading mechanism using particle swarm optimization in 5g enabled edge nodes. *Cluster Computing*, pp. 1–12 (2022)
7. Bacanin, N., Bezdán, T., Venkatachalam, K., Zivkovic, M., Strumberger, I., Abouhawwash, M., Ahmed, A.B.: Artificial neural networks hidden unit and weight connection optimization by quasi-reflection-based learning artificial bee colony algorithm. *IEEE Access* **9**, 169135–169155 (2021)
8. Bacanin, N., Bezdán, T., Zivkovic, M., Chhabra, A.: Weight optimization in artificial neural network training by improved monarch butterfly algorithm. In: *Mobile Computing and Sustainable Informatics*, pp. 397–409. Springer (2022)
9. Bacanin, N., Petrovic, A., Zivkovic, M., Bezdán, T., Chhabra, A.: Enhanced salp swarm algorithm for feature selection. In: *International Conference on Intelligent and Fuzzy Systems*. pp. 483–491. Springer (2021)
10. Bacanin, N., Sarac, M., Budimirovic, N., Zivkovic, M., AlZubi, A.A., Bashir, A.K.: Smart wireless health care system using graph lstm pollution prediction and dragonfly node localization. *Sustainable Computing: Informatics and Systems* **35**, 100711 (2022)
11. Bacanin, N., Stoean, C., Zivkovic, M., Jovanovic, D., Antonijevec, M., Mladenovic, D.: Multi-swarm algorithm for extreme learning machine optimization. *Sensors* **22**(11), 4204 (2022)
12. Bacanin, N., Stoean, R., Zivkovic, M., Petrovic, A., Rashid, T.A., Bezdán, T.: Performance of a novel chaotic firefly algorithm with enhanced exploration for tackling global optimization problems: Application for dropout regularization. *Mathematics* **9**(21), 2705 (2021)
13. Bacanin, N., Tuba, E., Zivkovic, M., Strumberger, I., Tuba, M.: Whale optimization algorithm with exploratory move for wireless sensor networks localization. In: *International conference on hybrid intelligent systems*. pp. 328–338. Springer (2019)

14. Bacanin, N., Vukobrat, N., Zivkovic, M., Bezdan, T., Strumberger, I.: Improved harris hawks optimization adapted for artificial neural network training. In: International Conference on Intelligent and Fuzzy Systems. pp. 281–289. Springer (2021)
15. Bacanin, N., Zivkovic, M., Al-Turjman, F., Venkatachalam, K., Trojovský, P., Strumberger, I., Bezdan, T.: Hybridized sine cosine algorithm with convolutional neural networks dropout regularization application. *Scientific Reports* **12**(1), 1–20 (2022)
16. Bacanin, N., Zivkovic, M., Bezdan, T., Venkatachalam, K., Abouhawwash, M.: Modified firefly algorithm for workflow scheduling in cloud-edge environment. *Neural Computing and Applications* **34**(11), 9043–9068 (2022)
17. Bacanin, N., Zivkovic, M., Jovanovic, L., Ivanovic, M., Rashid, T.A.: Training a multilayer perception for modeling stock price index predictions using modified whale optimization algorithm. In: Computational Vision and Bio-Inspired Computing, pp. 415–430. Springer (2022)
18. Bacanin, N., Zivkovic, M., Sarac, M., Petrovic, A., Strumberger, I., Antonijevic, M., Petrovic, A., Venkatachalam, K.: A novel multiswarm firefly algorithm: An application for plant classification. In: International Conference on Intelligent and Fuzzy Systems. pp. 1007–1016. Springer (2022)
19. Basha, J., Bacanin, N., Vukobrat, N., Zivkovic, M., Venkatachalam, K., Hubálovský, S., Trojovský, P.: Chaotic harris hawks optimization with quasi-reflection-based learning: an application to enhance cnn design. *Sensors* **21**(19), 6654 (2021)
20. Bezdan, T., Cvetnic, D., Gajic, L., Zivkovic, M., Strumberger, I., Bacanin, N.: Feature selection by firefly algorithm with improved initialization strategy. In: 7th Conference on the Engineering of Computer Based Systems. pp. 1–8 (2021)
21. Bezdan, T., Milosevic, S., Venkatachalam, K., Zivkovic, M., Bacanin, N., Strumberger, I.: Optimizing convolutional neural network by hybridized elephant herding optimization algorithm for magnetic resonance image classification of glioma brain tumor grade. In: 2021 Zooming Innovation in Consumer Technologies Conference (ZINC). pp. 171–176. IEEE (2021)
22. Bezdan, T., Stoean, C., Naamany, A.A., Bacanin, N., Rashid, T.A., Zivkovic, M., Venkatachalam, K.: Hybrid fruit-fly optimization algorithm with k-means for text document clustering. *Mathematics* **9**(16), 1929 (2021)
23. Bezdan, T., Zivkovic, M., Bacanin, N., Chhabra, A., Suresh, M.: Feature selection by hybrid brain storm optimization algorithm for covid-19 classification. *Journal of Computational Biology* (2022)
24. Bezdan, T., Zivkovic, M., Bacanin, N., Strumberger, I., Tuba, E., Tuba, M.: Multi-objective task scheduling in cloud computing environment by hybridized bat algorithm. *Journal of Intelligent & Fuzzy Systems* **42**(1), 411–423 (2022)
25. Bezdan, T., Zivkovic, M., Tuba, E., Strumberger, I., Bacanin, N., Tuba, M.: Glioma brain tumor grade classification from mri using convolutional neural networks designed by modified fa. In: International conference on intelligent and fuzzy systems. pp. 955–963. Springer (2020)
26. Budimirovic, N., Prabhu, E., Antonijevic, M., Zivkovic, M., Bacanin, N., Strumberger, I., Venkatachalam, K.: Covid-19 severity prediction using enhanced whale with salp swarm feature classification. *Computers, Materials and Continua*, pp. 1685–1698 (2022)
27. Cormack, G.V., et al.: Email spam filtering: A systematic review. *Foundations and Trends® in Information Retrieval* **1**(4), 335–455 (2008)

28. Dada, E.G., Bassi, J.S., Chiroma, H., Adetunmbi, A.O., Ajibuwa, O.E., et al.: Machine learning for email spam filtering: review, approaches and open research problems. *Heliyon* **5**(6), e01802 (2019)
29. Dedetürk, B.K., Akay, B.: Spam filtering using a logistic regression model trained by an artificial bee colony algorithm. *Applied Soft Computing* **91**, 106229 (2020)
30. Ebraheem, M., Jyothsna, T.: Comparative performance evaluation of teaching learning based optimization against genetic algorithm on benchmark functions. In: 2015 IEEE Power, Communication and Information Technology Conference (PCITC). pp. 327–331. IEEE (2015)
31. Ergin, S., Sora Gunal, E., Yigit, H., Aydin, R.: Turkish anti-spam filtering using binary and probabilistic models. *Global Journal on Technology* **1** (2012)
32. Gibson, S., Issac, B., Zhang, L., Jacob, S.M.: Detecting spam email with machine learning optimized with bio-inspired metaheuristic algorithms. *IEEE Access* **8**, 187914–187932 (2020)
33. Heidari, A.A., Mirjalili, S., Faris, H., Aljarah, I., Mafarja, M., Chen, H.: Harris hawks optimization: Algorithm and applications. *Future generation computer systems* **97**, 849–872 (2019)
34. Jáñez-Martino, F., Alaiz-Rodríguez, R., González-Castro, V., Fidalgo, E., Alegre, E.: A review of spam email detection: analysis of spammer strategies and the dataset shift problem. *Artificial Intelligence Review*, pp. 1–29 (2022)
35. Jovanovic, D., Antonijevic, M., Stankovic, M., Zivkovic, M., Tanaskovic, M., Bacanin, N.: Tuning machine learning models using a group search firefly algorithm for credit card fraud detection. *Mathematics* **10**(13), 2272 (2022)
36. Jovanovic, D., Marjanovic, M., Antonijevic, M., Zivkovic, M., Budimirovic, N., Bacanin, N.: Feature selection by improved sand cat swarm optimizer for intrusion detection. In: 2022 International Conference on Artificial Intelligence in Everything (AIE). pp. 685–690. IEEE (2022)
37. Jovanovic, L., Zivkovic, M., Antonijevic, M., Jovanovic, D., Ivanovic, M., Jassim, H.S.: An emperor penguin optimizer application for medical diagnostics. In: 2022 IEEE Zooming Innovation in Consumer Technologies Conference (ZINC). pp. 191–196. IEEE (2022)
38. Karaboga, D.: Artificial bee colony algorithm. *scholarpedia* **5**(3), 6915 (2010)
39. Karim, A., Azam, S., Shanmugam, B., Kannoorpatti, K., Alazab, M.: A comprehensive survey for intelligent spam email detection. *IEEE Access* **7**, 168261–168295 (2019)
40. Kontsewaya, Y., Antonov, E., Artamonov, A.: Evaluating the effectiveness of machine learning methods for spam detection. *Procedia Computer Science* **190**, 479–486 (2021)
41. Latha, R., Saravana Balaji, B., Bacanin, N., Strumberger, I., Zivkovic, M., Kabiljo, M.: Feature selection using grey wolf optimization with random differential grouping. *Comput. Syst. Sci. Eng.* **43**(1), 317–332 (2022)
42. Maity, D., Ghosal, S., Banerjee, S., Chanda, C.K.: Bare bones teaching learning based optimization for combined economic emission load dispatch problem. In: 3rd International Conference on Electrical, Electronics, Engineering Trends, Communication, Optimization and Sciences (EEECOS 2016). pp. 1–6 (2016). <https://doi.org/10.1049/cp.2016.1554>
43. Mccord, M., Chuah, M.: Spam detection on twitter using traditional classifiers. In: international conference on Autonomic and trusted computing. pp. 175–186. Springer (2011)

44. Mirjalili, S.: Sca: A sine cosine algorithm for solving optimization problems. *Knowledge-Based Systems* **96**, 120–133 (2016). <https://doi.org/10.1016/j.knosys.2015.12.022>, <https://www.sciencedirect.com/science/article/pii/S0950705115005043>
45. Petrovic, A., Bacanin, N., Zivkovic, M., Marjanovic, M., Antonijevic, M., Strumberger, I.: The adaboost approach tuned by firefly metaheuristics for fraud detection. In: 2022 IEEE World Conference on Applied Intelligence and Computing (AIC). pp. 834–839. IEEE (2022)
46. Petrovic, A., Strumberger, I., Bezdan, T., Jassim, H.S., Nassor, S.S.: Cryptocurrency price prediction by using hybrid machine learning and beetle antennae search approach. In: 2021 29th Telecommunications Forum (TELFOR). pp. 1–4. IEEE (2021)
47. Prakash, S., Kumar, M.V., Ram, S.R., Zivkovic, M., Bacanin, N., Antonijevic, M.: Hybrid glfl enhancement and encoder animal migration classification for breast cancer detection. *Comput. Syst. Sci. Eng.* **41**(2), 735–749 (2022)
48. Raj, A., Venkaiah, C.: Optimal pmu placement by teaching-learning based optimization algorithm. In: 2015 39th National Systems Conference (NSC). pp. 1–6 (2015). <https://doi.org/10.1109/NATSYS.2015.7489080>
49. Rao, R.V., Savsani, V.J., Vakharia, D.: Teaching-learning-based optimization: a novel method for constrained mechanical design optimization problems. *Computer-aided design* **43**(3), 303–315 (2011)
50. Salb, M., Bacanin, N., Zivkovic, M., Antonijevic, M., Marjanovic, M., Strumberger, I.: Extreme learning machine tuning by original sine cosine algorithm. In: 2022 IEEE World Conference on Applied Intelligence and Computing (AIC). pp. 143–148. IEEE (2022)
51. Salb, M., Jovanovic, L., Zivkovic, M., Tuba, E., Elsadai, A., Bacanin, N.: Training logistic regression model by enhanced moth flame optimizer for spam email classification. In: *Computer Networks and Inventive Communication Technologies*, pp. 753–768. Springer (2023)
52. Srekanth, G., Ahmed, S.A.N., Sarac, M., Strumberger, I., Bacanin, N., Zivkovic, M.: Mobile fog computing by using sdn/nfv on 5g edge nodes. *Comput. Syst. Sci. Eng.* **41**(2), 751–765 (2022)
53. Strumberger, I., Bezdan, T., Ivanovic, M., Jovanovic, L.: Improving energy usage in wireless sensor networks by whale optimization algorithm. In: 2021 29th Telecommunications Forum (TELFOR). pp. 1–4. IEEE (2021)
54. Strumberger, I., Rakic, A., Stanojlovic, S., Arandjelovic, J., Bezdan, T., Zivkovic, M., Bacanin, N.: Feature selection by hybrid binary ant lion optimizer with covid-19 dataset. In: 2021 29th Telecommunications Forum (TELFOR). pp. 1–4. IEEE (2021)
55. Strumberger, I., Tuba, E., Bacanin, N., Zivkovic, M., Beko, M., Tuba, M.: Designing convolutional neural network architecture by the firefly algorithm. In: 2019 International Young Engineers Forum (YEF-ECE). pp. 59–65. IEEE (2019)
56. Talatahari, S., Bayzidi, H., Saraee, M.: Social network search for global optimization. *IEEE Access* **9**, 92815–92863 (2021)
57. Wolpert, D.H., Macready, W.G.: No free lunch theorems for optimization. *IEEE transactions on evolutionary computation* **1**(1), 67–82 (1997)
58. Xu, Y., Peng, Y., Su, X., Yang, Z., Ding, C., Yang, X.: Improving teaching-learning-based-optimization algorithm by a distance-fitness learning strategy. *Knowledge-Based Systems*, p. 108271 (2022)
59. Yang, X.S.: Firefly algorithms for multimodal optimization. In: *International symposium on stochastic algorithms*. pp. 169–178. Springer (2009)

60. Yang, X.S.: Bat algorithm for multi-objective optimisation. *International Journal of Bio-Inspired Computation* **3**(5), 267–274 (2011)
61. Zivkovic, M., Bacanin, N., Arandjelovic, J., Rakic, A., Strumberger, I., Venkatachalam, K., Joseph, P.M.: Novel harris hawks optimization and deep neural network approach for intrusion detection. In: *Proceedings of International Joint Conference on Advances in Computational Intelligence*. pp. 239–250. Springer (2022)
62. Zivkovic, M., Bacanin, N., Arandjelovic, J., Strumberger, I., Venkatachalam, K.: Firefly algorithm and deep neural network approach for intrusion detection. In: *Applications of Artificial Intelligence and Machine Learning*, pp. 1–12. Springer (2022)
63. Zivkovic, M., Bacanin, N., Tuba, E., Strumberger, I., Bezdan, T., Tuba, M.: Wireless sensor networks life time optimization based on the improved firefly algorithm. In: *2020 International Wireless Communications and Mobile Computing (IWCMC)*. pp. 1176–1181. IEEE (2020)
64. Zivkovic, M., Bacanin, N., Venkatachalam, K., Nayyar, A., Djordjevic, A., Strumberger, I., Al-Turjman, F.: Covid-19 cases prediction by using hybrid machine learning and beetle antennae search approach. *Sustainable Cities and Society* **66**, 102669 (2021)
65. Zivkovic, M., Bezdan, T., Strumberger, I., Bacanin, N., Venkatachalam, K.: Improved harris hawks optimization algorithm for workflow scheduling challenge in cloud–edge environment. In: *Computer Networks, Big Data and IoT*, pp. 87–102. Springer (2021)
66. Zivkovic, M., Jovanovic, L., Ivanovic, M., Bacanin, N., Strumberger, I., Joseph, P.M.: Xgboost hyperparameters tuning by fitness-dependent optimizer for network intrusion detection. In: *Communication and Intelligent Systems*, pp. 947–962. Springer (2022)
67. Zivkovic, M., Jovanovic, L., Ivanovic, M., Krdzic, A., Bacanin, N., Strumberger, I.: Feature selection using modified sine cosine algorithm with covid-19 dataset. In: *Evolutionary Computing and Mobile Sustainable Networks*, pp. 15–31. Springer (2022)
68. Zivkovic, M., Petrovic, A., Bacanin, N., Milosevic, S., Veljic, V., Vesic, A.: The covid-19 images classification by mobilenetv3 and enhanced sine cosine metaheuristics. In: *Mobile Computing and Sustainable Informatics*, pp. 937–950. Springer (2022)
69. Zivkovic, M., Petrovic, A., Venkatachalam, K., Strumberger, I., Jassim, H.S., Bacanin, N.: Novel chaotic best firefly algorithm: Covid-19 fake news detection application. In: *Advances in Swarm Intelligence*, pp. 285–305. Springer (2023)
70. Zivkovic, M., Stoean, C., Petrovic, A., Bacanin, N., Strumberger, I., Zivkovic, T.: A novel method for covid-19 pandemic information fake news detection based on the arithmetic optimization algorithm. In: *2021 23rd International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC)*. pp. 259–266. IEEE (2021)
71. Zivkovic, M., Tair, M., Venkatachalam, K., Bacanin, N., Hubálovský, Š., Trojovský, P.: Novel hybrid firefly algorithm: an application to enhance xgboost tuning for intrusion detection classification. *PeerJ Computer Science* **8**, e956 (2022)
72. Zivkovic, M., Zivkovic, T., Venkatachalam, K., Bacanin, N.: Enhanced dragonfly algorithm adapted for wireless sensor network lifetime optimization. In: *Data intelligence and cognitive informatics*, pp. 803–817. Springer (2021)
73. Zou, X., Hu, Y., Tian, Z., Shen, K.: Logistic regression model optimization and case analysis. In: *2019 IEEE 7th international conference on computer science and network technology (ICCSNT)*. pp. 135–139. IEEE (2019)

Open Access This chapter is licensed under the terms of the Creative Commons Attribution-NonCommercial 4.0 International License (<http://creativecommons.org/licenses/by-nc/4.0/>), which permits any noncommercial use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license and indicate if changes were made.

The images or other third party material in this chapter are included in the chapter's Creative Commons license, unless indicated otherwise in a credit line to the material. If material is not included in the chapter's Creative Commons license and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder.

