RESEARCH ARTICLE

Modified Genetic Algorithm for Feature Selection and Hyper Parameter Optimization: Case of XGBoost in Spam Prediction

NAZEEH GHATASHEH[1], **ISMAIL ALTAHARWA**[2], **AND KHALED ALDEBEI**[1]
[1]Department of Information Technology, The University of Jordan, Aqaba 77110, Jordan
[2]Department of Computer Information Systems, The University of Jordan, Aqaba 77110, Jordan

Corresponding author: Nazeeh Ghatasheh (n.ghatasheh@ju.edu.jo)

ABSTRACT Recently, spam on online social networks has attracted attention in the research and business world. Twitter has become the preferred medium to spread spam content. Many research efforts attempted to encounter social networks spam. Twitter brought extra challenges represented by the feature space size, and imbalanced data distributions. Usually, the related research works focus on part of these main challenges or produce black-box models. In this paper, we propose a modified genetic algorithm for simultaneous dimensionality reduction and hyper parameter optimization over imbalanced datasets. The algorithm initialized an eXtreme Gradient Boosting classifier and reduced the features space of tweets dataset; to generate a spam prediction model. The model is validated using a 50 times repeated 10-fold stratified cross-validation, and analyzed using nonparametric statistical tests. The resulted prediction model attains on average 82.32% and 92.67% in terms of geometric mean and accuracy respectively, utilizing less than 10% of the total feature space. The empirical results show that the modified genetic algorithm outperforms Chi^2 and PCA feature selection methods. In addition, eXtreme Gradient Boosting outperforms many machine learning algorithms, including BERT-based deep learning model, in spam prediction. Furthermore, the proposed approach is applied to SMS spam modeling and compared to related works.

INDEX TERMS Genetic algorithm, business analytics, extreme gradient boosting, feature selection, hyper parameter optimization, spam prediction.

I. INTRODUCTION

Spam remains one of the long lasting security threats. E-mail spams represent a true challenge against mail service providers at the early stages of the Internet. Web spams exploit social engineering to lure a privileged user to login into a deceptive service. As Internet users developed awareness skills and became more competent to distinguish fake web content from truly legitimate one, attackers exploit the pervasiveness of social networks and corresponding media to launch the latest generation of spams, namely social spam. In addition to the opportunity to target a larger number of victims, social networks create an environment for ever-evolving avenues for spammers. It goes beyond traditional individual compromising activities such as

monetary frauds towards large-scale campaigns. Quite recently released Twitter dataset distinguished more than five ways of twitter spams, including, but not limited to, profanity, insulting, hate speech, malicious links, fraudulent reviews [1]. Similarly, recent research efforts considered similar spamming approaches against other online social networks and short message service (SMS) [2], [3]. It is not surprising that twitter reviews spam policy periodically [4].

As social spam campaigns emerged as a contemporary challenge against users, companies and even more governments, countermeasures evolved in a hand raising contest fashion. Earlier solutions were limited to the rule-based and regular expression matching. However, as spammers developed good experience to evade such detectors, information security practitioners considered content-based characteristics restrictively. Contemporary mature solutions utilize both content-based and account-based characteristics. In most

The associate editor coordinating the review of this manuscript and approving it for publication was Alberto Cano.

cases, the ultimate goal is to find the shortest list of characteristics or features that indicate spamming behavior [5], [6]. Some studies step further to identify spammers themselves [7], [8]. Machine learning techniques are leveraged in many ways to develop detection models. Earlier models utilized straightforward classification and categorization algorithms such as Support Vector MAchine (SVM), Naïve Bayes (NB), K-Nearest Neighbor (K-NN), and Decision Trees (DT) [3], [5], [6]. More advanced solutions explore opportunities of improvement as a result of utilizing deep learning (DL) techniques [2], [9].

Deep learning based solutions approved to outperform conventional machine learning based prediction models. Considering social spams, such behavior of deep learning models is justifiable as it performs well in identifying local patterns [9]. However, such performance comes at the expenses of model complexity. Additionally, Artificial Neural Networks (ANNs) models of deep learning are hard to interpret. Scalability and interpretability remain two contradictory desired characteristics of any social spam detector. In order to tackle this issue, we propose a novel dimension reduction solution. As parameter tuning is an unavoidable task regardless of the nature of the underlying prediction model, the proposed solution leverages a genetic algorithm to tune the parameters of the prediction model and select best descriptive features simultaneously. Such generated prediction models are still interpretable utilizing the final set of retained features. Further, proposed architecture allows developers to choose among a wide range of granularity depending on their targets and underlying computation capabilities.

A wide range of optimization techniques are proposed in literature. Alatas and Bingol categorized intelligent optimization techniques according to their scientific basis [10]. Further they compared their performance to the light-based intelligent optimizers [11]. Genetic algorithms (GA), biology based optimizer, is the most popular type of evolutionary algorithms (EA) for parameter optimization. It demonstrates noticeably outstanding performance for a wide range of problems. Genetic algorithms retain merits of both meta-heuristic search algorithms and stochastic optimization techniques [12]. This combination enables genetic algorithms to reach a global optima within a relatively fewer number of generations compared to other evolutionary algorithms.

One of the major issues in spam text research is the limited availability of labeled text datasets with high quality [13], [14]. For example the well known benchmark datasets are few, and many researchers use tools to collect domain specific datasets. As a result, many of the available text datasets have limited number of attributes, unverified class labeling, related to a specific language, imbalanced class distributions, or biased data. Furthermore, currently social media facilitate sharing multimedia content (e.g.,audio, video, text, images, etc) but incorporating such mix of content in model building seems to be one of the future challenges. Table 1 shows a summary of the related research works in Tweets modeling and points to some research gaps.

In order to evaluate the proposed solution, a real-world Twitter dataset is utilized, a quite large number of experiments incurred. Most experiments ended up with satisfactory performance due to the incorporation of the feature selection process. Some experiments provided outstanding results that outperform base-line solutions, even deep learning solutions. Incurred experiments reveal appropriateness of the proposed approach to handle social spam detection problem, providing a trade off between prediction performance and computation capabilities. Furthermore, the proposed approach is still applicable to a wide range of data mining problems. Below are the key contributions of this research:

1) Proposing a social spam content-based detection approach that considers wide variety of contemporary ways of spammers.
2) Developing a novel genetic algorithm to initialize a powerful classifier and feature selection.
3) Validating the proposed approach against publicly released twitter real-world dataset.

The rest of this paper is organized as follows, Section II investigates literature and related works. The proposed social spam detector and the corresponding genetic algorithm feature extraction approach are elaborated in Section III. Results are presented and discussed in Sections IV and V, respectively.Finally, Conclusion and future directions are drawn in Section VI.

II. RELATED WORKS
Recently, there has been a significant interest in detecting twitter spam. Compared to the traditional mail spam and web spam, twitter went beyond phishing, fraudulent, and scam. It creates new avenues for profanity, insulting, spreading hate speech, and bullying [1]–[3], [13], [24]. Researchers have investigated wide range of approaches to accommodate such divergence. Two streams of countermeasures have been proposed. First approach considers feature extraction. Second approach considers graph-based solution. Feature based solutions investigate content-based features, account-based features, or both of them. Graph based solutions investigate communication graphs of spam spreading focusing on identification of spammers.

In recent times, feature selection is one of the important key of research in machine learning, image retrieval, text mining, intrusion detection, etc. According to literature, different algorithms have been developed and employed for feature selection. For example, a greedy search based sequential forward selection (SFS) [25] and sequential backward selection (SBS) [26] have been applied for feature selection. However, these approaches suffer from a range of problems, such as stacking in local optima and high computational cost. In order to address these problems, new algorithms for feature selection have been proposed [27]–[29], such as Particle Swarm Optimization (PSO) [30], Ant Colony Optimization(ACO) [31], and Genetic Algorithm (GA) [32]. Furthermore, a novel filter feature selection method named the Proportional Rough Feature Selector (PRFS) has been

TABLE 1. Summary of related research works.

Ref.	Work Done	Classifiers	Claims	Notes
[15]	Improved Genetic Algorithm for feature selection	Multinomial Naive Bayes	Outperforming Information Gain and CHI square for feature selection	Maximum reduction of features is more than 50%. Reports only total accuracy. No information regarding model validation
[16]	Mutual information method for feature selection, Predefined number of features is tested	Naïve Bayes, SVM, Regression Logistic, Decision Tree	Naive Bayes classification was the fastest. SVM Linear Classifier attained highest accuracy	Manual setup of selected features size. No significant improvement over ANNOVA F and CHI square for feature selection. Train and Test dataset splits. Report Accuracy, SD, Error %
[17]	Used ten filter-based feature selection techniques, ANOVA tests	k-nearest neighbors, Logistic Regression, C4.5 decision tree, Multilayer Perceptron, ANN	200 is the optimal number of features to select	Reports AUC. 5-fold cross-validation
[18]	CFS Subset Evaluation and Information Gain feature selection	SVM, K-Nearest Neighbor, Naive Bayes, AdaBoost, Bagging algorithms	Accuracy 92.96% by SVM	Focus on one domain, no basis for generalization.
[19]	Trend-line analysis of the relationships between number of features and accuracy	Naïve Bayes, Maximum Entropy, SVM, Extreme Learning Machine	The relationship between the number features and the accuracy can be determined and it is independent of the machine learning approach	Used balanced datastes. Validated using percentage split (75% training and 25% testing)
[20]	Selected 42 to 34,855 features that represent a public tweets dataset	Naïve Bayes, Maximum Entropy	86.3% Accuracy by "Sentiment-topic features". Selecting features beyond 500 will not improve the performance significantly	Report Accuracy only. No information about model validation
[21]	Evolutionary computation based feature selection. Subset of 500, 1000, and 1500 features examined	Feature Selection: Information Gain feature, Particle Swarm, Ant Colony, Cuckoo Search, Firefly Search. Classification: LibLinear, K-Nearest Neighbor, Naive Bayes	LibLinear attained best performance. 78.6 AUC using 500 features by Firefly Search	5-fold cross-validation. Arbitrary selected features number. Reports AUC only
[22]	Compare models and Feature selection	SVM, Naïve Bayes, ANN, K-Nearest Neighbor, Random Forest. Deep Learning: Recurrent NN, LSTM	Tweets Models: Accuracy 95.09 using LSTM. 5000 Features Accuracy 93.81	Ham is the positive class. No information about model validation. Arbitrary feature selection. Minimum number of features selected 5000
[23]	Assess transfer learning in predicting disaster tweets, SMS spam, News Groups, and IMDB binary sentiment classification	DistilBERT, BERT, H2O AutoML, and 7 others	Accuracy 0.84%, 0.98%, 0.98% in predicting disaster tweets, SMS spam, and IMDB respectively.	Reports Accuracy only. Percentage split validation (80% training and 20% validation).

proposed in [33]. The method addresses a high dimensional matrix in a short text classification problem. The method makes a regional distinction using a set of terms in order to differentiate documents that exactly belong to a class and documents that possibly belong to a class. In the work of [34], the authors have presented a comparative study of eight filter methods by employing mutual information using 33 datasets. Furthermore, in the work of [35], 12 feature selection methods are compared on text classification problem.

Genetic algorithm has been known to be a very efficient and useful approach for feature selection, as described in [36]–[40]. This is because of its ability of changing the functional configuration in order to improve the performance results. In [41], the authors have applied a Genetic Algorithm in order to reduce the number of features extracted from a Flavia image dataset. The authors of [42], [43] have proposed a hybrid Genetic Algorithm for feature selection based on machine learning techniques. They have investigated the performance of their algorithm using different datasets, such as *Wine* dataset and synthetic data sets. In [15], an approach for enhancing a classification performance of natural crisis-related Twitter messages has been proposed.

In this approach, a Genetic Algorithm has been utilized for feature selection. Another study has been proposed in [44]. The study employs a Genetic Algorithm for feature selection in order to increase a classification accuracy for breast cancer diagnosis.

Different feature selection approaches have been applied on many real-world applications, such as text categorization [45], image retrieval [46], intrusion detection [47]. Several feature selection approaches have also been applied on tweets classification. For example, the work of [16] has presented a method for sentiment analysis of airline tweets. It employs a mutual information method for the process of feature selection. Furthermore, the work of [15] has implemented an improved Genetic Algorithm for disaster preparedness and response in the Philippines. The algorithm aims to select the most important features from a large number of features for the classification process of disaster-related tweets. In [17], the authors have considered Chi-Squared, Mutual Information, Kolmogorov-Smirnov statistic, area under the Precision-Recall curve, and area under the Receiver Operating Characteristic curve for feature selection on a large high-dimensional dataset of collected tweets. Each tweet is

labeled to a positive sentiment or negative sentiment. The results demonstrated that employing these feature selection techniques on a sentiment classification process can have a great impact on the performance of a classifier. The [18] has applied classification techniques on tweets belonging to Renewable Energy. The Correlation based Feature Selection (CFS) Subset Evaluation and Information Gain feature selection have been used to reduce the number of used features.

The literature shows that the number of selected features used for tweets classification greatly affects the performance of the employed classifier. However, only a few works have discussed how and what an appropriate number of features should be selected to achieve the best classification performance [13], [19]. The approach of [17] has shown that using between 75 and 200 features enhances the tweets classification results over using the full feature set. In [20], the authors have investigated the using between 42 and 34,855 features to represent 1000 instances from the Stanford Twitter Corpus. They have found that using more than 500 features will not significantly improve the performance of a classifier. The work of [21] has studied the effect of the application of two-stage feature selection on the twitter sentiment analysis performance. A filter feature selection based on information gain has been used and 3 feature sets of 500, 1000, and 1500 features have been produced.

III. THE PROPOSED APPROACH

Computer-based Genetic Algorithm (GA) [48] is a search heuristic that was inspired from the natural evolution theory of Darwin. Since decades, GA has been actively used by researchers to address many challenges in different domains such as malware detection [49], energy optimization [50], cancer classification [51] and so on.

Recently, GA has been used as a search strategy for dimensionality reduction of a relatively large feature space [52]. Such an approach evades the limitations of the exhaustive search strategies. GA can be used to optimize the parameters of machine learning algorithms and reduce the dimensionality of the problem space. One approach in text modeling is to convert the input text into a set of features; such as TF/iDF modeling. Usually, the number of features is extremely large and so an overfitting probability is high. On the other hand, real-world classification datasets are usually an imbalanced distribution of class labels. Imbalanced datasets impose an additional challenge in avoiding classification bias and overfitting.

A. EXTREME GRADIENT BOOSTING

Chen and Guestrin in [53] introduced a powerful tree boosting algorithm, which is named eXtreme Gradient Boosting (XGBoost). The algorithm is claimed to be scalable; sparsity-aware; takes into consideration data compression and sharding; and cache-aware access. Figure 1 illustrates XGboost algorithm architecture. Each tree is trained on the residual error of the previous tree which improves the performance
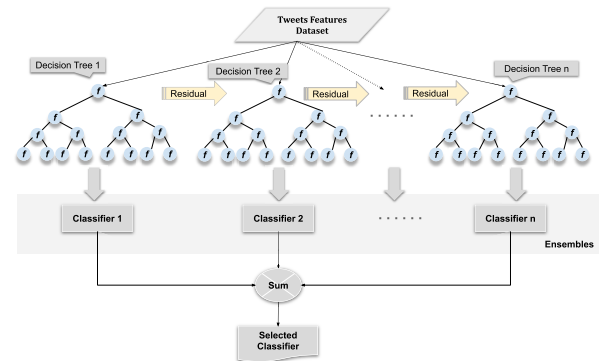

FIGURE 1. eXtreme gradient boosting (XGBoost).

of the constructed tree. The sum of each tree's predictions constructs the final prediction.

The characteristics of XGBoost enable it to outperform other machine learning algorithms and require less system resources. Theoretical and empirical proofs support these claims in [53]–[57]. XGboost in [54] produced the best performing models over 11 machine learning algorithms in text-based spam classification. However, tree-based algorithms in general tend to perform well in relatively small number of features compared to artificial neural networks. Therefore, this research aims at leveraging the benefits of XGBoost algorithm by reducing the number of text features in the prediction model building process.

The major challenge in building XGBoost-based models is proper parameter tuning [53]–[56]. This research aims at proposing a novel GA variation that optimizes the parameters of a classifier (i.e., eXtreme Gradient Boosting), and to reduce the features space simultaneously.

B. DATASET DESCRIPTION

The main tweets dataset used in this research was introduced in [22]. It has 5096 tweets. About 17% of tweets are labeled as "Spam" and the rest as "Ham". Tweets are labeled in a manual fashion by considering and examining each one separately [22]. If a tweet content is considered as unacceptable by the community or harmful, then the tweet is labeled as spam. Otherwise, it is labeled as ham (i.e.,, normal tweet). Figure 2 summarizes the number of the text characters in all instances. The length of the stored characters of each tweet may exceed the number of original tweet length because some special characters and emoticons are stored as a set of representative Unicode characters.

It is apparent that the average character length, which represents each tweet, is about 100 characters in both classes, and there is no significant variance difference in the distribution of both classes as well. Therefore, length analysis adds to the challenges in building a robust classification model.

C. THE MODIFIED GA

The modified GA aims at directing the stochastic selection aspect towards a fine subset of features. At the same time, to find the best possible classification algorithm parameters.
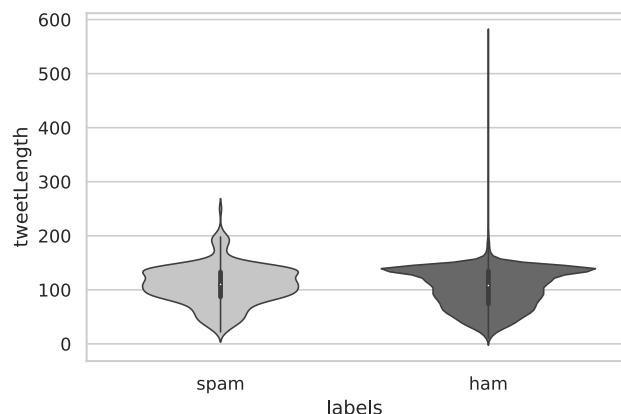
FIGURE 2. The character length distribution of the text (emoticons are stored as a series of Unicode characters).

Therefore, it is to find the best combination of features and parameters simultaneously. Usually, GA is used to either initialize the classification algorithm parameters or in feature selection. The proposed modifications would leverage the capabilities of GA in defining the optimal combination of features subset and parameters. Moreover, particular modifications of some methods limit the absolute randomness of GA phases. For example, ensuring no duplicate genes in each chromosome. The Modified GA and its phases are presented in the following subsections:

1) GA MAIN CODE

Many recent research studies in different domains [58]–[61] illustrated the power of GA in optimizing the parameters of XGBoost to achieve better prediction performance. Algorithm 1 represents the initial configurations of the modified GA that is used to optimize the parameters of XGboost and select the most appropriate features subset. Initially, a number of GA parameters will set the maximum percentage of features to be selected, the parents' crossover ratio in each population, the maximum number of generations, and the number of classifier parameters to be optimized. The result will be a chromosome having an optimized set of XGboost parameters and the selected features subset. The chromosome structure is illustrated in Figure 5. The input dataset is split into 70% training and 30% testing partitions for the GA-based XGBoost model building and validation. Table 4 describes the GA parameters.

2) INITIAL POPULATION

Creating the initial population of the GA is challenging as it is not an easy task to select a representative subset of the whole population. Neither in selecting the initial set of classifier parameters nor the subset of the feature set. Redundancy of gene values is also one of the issues to consider at this phase. To limit the absolute randomness of the GA, XGBoost boosting parameters are generated using a uniform random number generator within a recommended value range (Table 5 lists the value ranges of XGBoost parameters).

Algorithm 1 Genetic Algorithm

1: Set percentage of features to be selected
2: Set crossover ratio
3: Set number of parents in initial population
4: Set number of generations
5: Number of parents to select = crossover ration * number of parents
6: Define classifier parameters to be optimized
7: generateTrainTestData(Features Dataset) ▷ 30:70% data split
8: Generate initial population(`Number of parents`)
9: fitness function(initial generation)
10: **for** `Number of generations` **do**
11:     train(current population)
12:     Select parents (Number of parents to select) ▷ return those having highest fitness value
13:     children = crossover(selected parents)
14:     mutate (children)
15: **end for**
16: Best Chromosome = highest fitness chromosome
17: **procedure** train(current population)
18:     **for** `c = 1 to Number of chromosomes` **do**
19:         chromosome = population[c]
20:         *parameters* ← *chromosome*[1 − 7]
21:         *features* ← *chromosome*[8 − *end*]
22:         predictions = classify(parameters, traindata[features])
23:         fitness function(actual labels, predictions)
24:     **end for**
25:     **return** *fitness*
26: **end procedure**

The features subset, which is part of the chromosome, is created by a custom procedure that randomly selects a subset of the whole feature set; i.e., subset of the whole TF-iDF vector. The procedure ensures creating a chromosome with no duplicate features and selecting from the full features vector. Actually, the list of selected features is the set of features indices in the TF-iDF vector;

Initializing the initial population will result in forming the parents chromosomes according to the preset parents size in Algorithm 1.

3) FITNESS FUNCTION

The bias imposed by imbalanced class distributions generally favors the majority class; which in most cases does not represent the class of interest. Therefore, positive class based metrics will dramatically mislead the selection of the best model relying on the objective function. The Geometric Mean (GMean) on the contrary considers both the positive and negative class as an objective function [62]–[64]. The GA and the validation of the selected models in this research utilize the GMean as an objective function. In addition, it is used as the main metric in comparing the performance of different

---

Algorithm 2 Initializing Initial Population

1: **procedure** Generate initial population(*numberofparents*)
2:    **for** p = 1 to number of parents **do**
3:       learningRate[p] = rand.uniform(0.01, 1)
4:       nEstimators[p] = randrange(10, 1500, step = 25)
5:       maxDepth[p] = randrange(1, 10, step= 1)
6:       minChildWeight[p] = rand.uniform(0.01, 10.0)
7:       gammaValue[p] = rand.uniform(0.01, 10.0)
8:       subSample[p] = rand.uniform(0.01, 1.0)
9:       colSampleByTree[p] = rand.uniform(0.01, 1.0)
10:      features[p] = select text features   ▷ select text features ensures no duplicate genes are present in each chromosome
11:    **end for**
12:    concatenate parameters and features into chromosomes
13:    population = all generated chromosomes
14:    **return** *population*
15: **end procedure**

Algorithm 3 Simplified Fitness Function

1: **procedure** fitness function(*y_true, y_pred*)   ▷ y_true is actual class labels, y_pred predicted labels
2:    TP = Count TP(*y_true, y_pred*)
3:    TN = Count TN(*y_true, y_pred*)
4:    FP = Count FP(*y_true, y_pred*)
5:    FN = Count FN(*y_true, y_pred*)
6:    $TPR = \frac{TP}{TP+FN}$
7:    $TNR = \frac{TN}{FP+TN}$
8:    $fitness = \sqrt{TPR \times TNR}$
9:    **return** *fitness*
10: **end procedure**

Algorithm 4 Crossover

1: **procedure** crossover(Selected parents)   ▷ Uniform crossover
2:    **for** number of children to generate **do**
3:       uniform crossover of selected parents parameters
4:       uniform crossover of selected parents features
5:       new child = parents parameters and features after crossover
6:    **end for**
7:    **return** *newChildren*
       ▷ ensures no duplicate genes are present in each child chromosome
       ▷ ensures uniform crossover at almost half the size of each parent
       ▷ first 7 genes are classifier parameters and the rest are the text features
8: **end procedure**

Algorithm 5 Mutation

1: **procedure** mutate(children)
2:    **for** number of children **do**
3:       randomly select parameter gene index
4:       add random value to existing gene value
5:       Ensure not exceeding parameter value range
6:       randomly select feature gene index
7:       mutate by replacing it with a value from the full features set
8:       ensure adding new feature to the current chromosome
9:    **end for**
10:    **return** *MutatedChildren*
11: **end procedure**

---

classification models. The Algorithm 3 and Equation 8 illustrate the GMean calculation. It is the square root of the True Positive Rate (TPR or Recall) multiplied by the True Negative Rate (TNR or Specificity). The TPR is a positive class based metric and TNR is a negative class metric; deriving a TPR and TNR based metric equals a metric that represents the accuracy of both classes (i.e., Spam and Ham).

4) SELECTION AND CROSSOVER

Crossover is an essential phase in GA to generate a new number of children from the parents. A child's genes will be a combination of two parent chromosomes, so the children are expected to have better genes than the parents do. To achieve this, a uniform crossover is performed using almost half of each parent genes. The crossover phase ensures generating children where each has no duplicate genes. Each chromosome will undergo two crossovers; one for XGBoost parameters and the other for the selected features set. Algorithm 4 will select the best parents to crossover according to the

GMean objective function and generate a number of new children for the next generation.

5) MUTATION

Because of the stochastic nature of GA, some genes may be overseen in the initial population or in the generated children. To increase the chance of fair inclusion of missed geneses the mutation tries to include new genes in the children. One parameter gene and one feature gene are selected randomly in each child chromosome and replaced with a new value. Specifically for the selected features, the mutated gene value will be selected from the full features vector set such that it is not one of the parents' genes. Mutation is illustrated by Algorithm 5.

The modified GA will run for a preset number of generations aiming at the maximization of the GMean value of the generated models. The spam and ham text features are the TF-iDF vectors generated by the pre-processing step presented in Section III-D1. The best chromosome that will be selected in the last GA generation will contain the best XGBoost parameters and the accompanying selected spam
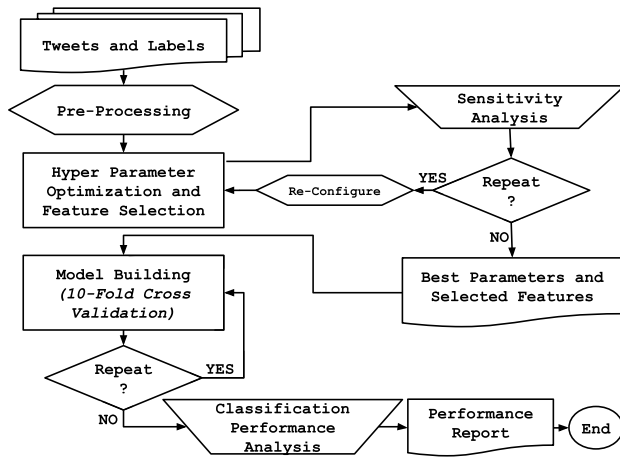
FIGURE 3. The proposed research methodology.

features subset. This chromosome will be used consequently to initialize an XGBoost algorithm to generate a spam prediction model in the Model Building phase illustrated in Figure 3 and Figure 4.

D. PROPOSED METHODOLOGY

The proposed methodology is divided into five main phases: (1) Dataset pre-processing, (2) Hyper parameter optimization and feature selection, (3) Sensitivity analysis, (4) Model building and validation, and (5) Classification performance analysis. Figure 3 is an abstract view of the proposed research methodology. Figure 4 is a more detailed view of the methodology, and the parameter values of each step are listed in Section IV.

1) DATASET PRE-PROCESSING

Each data instance contains raw tweet text and a label (i.e., "Ham" or "Spam"). Each tweet text and its label is pre-processed to be cleaned and converted into features through a number of steps: (1) Tokenize the tweet and remove extra space and special characters, (2) Stem each tokenized word using "Porter Stemmer" [65]. This will reduce the tokenized word to its root, stem, or base. (3) Each stem is given a weight using a vectorizer; depending on the term frequency–inverse document frequency (TF-iDF) [66], [67]. Therefore, each tweet is converted into a representative TF-iDF vector (i.e., a set of features), and (4) The class labels are encoded into 0's, i.e., "Ham" class, and 1's, i.e., "Spam" class, to satisfy the requirements of the classification algorithm.

2) HYPER PARAMETER OPTIMIZATION AND FEATURE SELECTION

In this step, a modified GA tunes the parameters of the classification algorithm such that it improves the prediction rates. It is divided into two main parts: (a) GA feature selection and (b) GA hyper parameter optimization. Each chromosome in this step is designed to hold two types of genes such that

genes at the beginning are the parameters to be optimized and the rest of genes are the selected features. Figure 5 shows the detailed structure of the chromosomes, the chromosomes after GA crossover, and an illustration of the genes after the mutation process.

The initial population consists of parent chromosomes holding randomly selected parameters within a recommended and pre-defined range based on literature [53], [68], [69]), and randomly selected unique features within each chromosome. It is the responsibility of the initialization algorithm to ensure choosing features without having any duplicates in each chromosome.

3) SENSITIVITY ANALYSIS

The main aim of the sensitivity analysis step is to find the best possible combination of XGBoost parameters and subset of feature space [70]. GA performs several hyper parameter optimizations and feature selections in order to examine the behavior of the classification algorithm. Consequently, the results of different optimizations and feature selections under different configurations lead to understanding the behavior of the algorithm.

Several GA configurations are examined in this step by mainly specifying: (1) the initial population size, crossover percentage, and number of generations; and (2) the desired percentage of features to retain (i.e., the number of feature genes in the chromosomes). The effect of the configurations on the objective function is examined to determine the candidate classifier parameters and the subset of feature space.

4) MODEL BUILDING AND VALIDATION

The optimized classifier parameters and selected features subset, which maximized the objective function, are used to build a robust classification model. 10-Fold stratified cross-validation (10 CV) is used to avoid the bias in model building process. This model building process is repeated 50 times (50 × 10 CV) to assess the classifier stability.

IV. EXPERIMENTS SETUP

Google^TM Colaboratory (a.k.a. Colab, https://colab.research.google.com/) environment is one of Google^TM Research products. Colab offers a browser-based machine learning projects' development environment that supports Python^TM https://www.python.org/ code run over different modern processing architectures. The experiments of this research are implemented as Python^TM projects and conducted over CPU-based Colab environment.

The major configurations of the proposed approach steps, which are described in Section III-D, are listed as follows:

A. DATASET PRE-PROCESSING

The used TF-iDF vectorizer parameters in Tweets text pre-processing are listed in Table 2. The maximum number of possible features is extracted according to the pre-processing step.
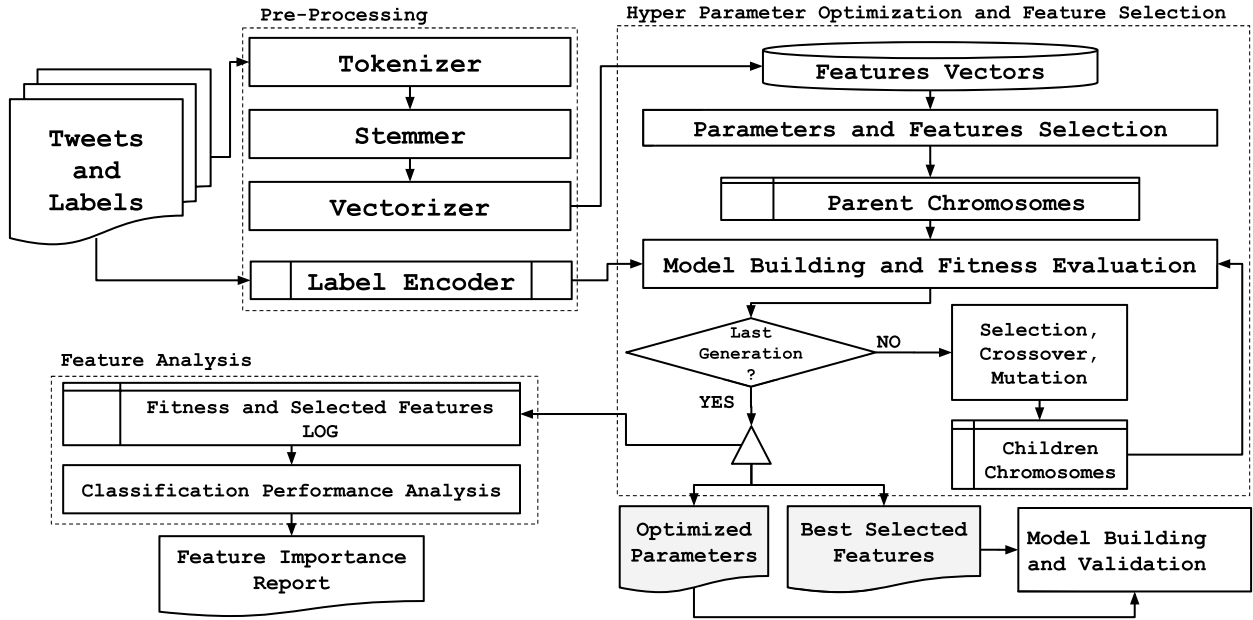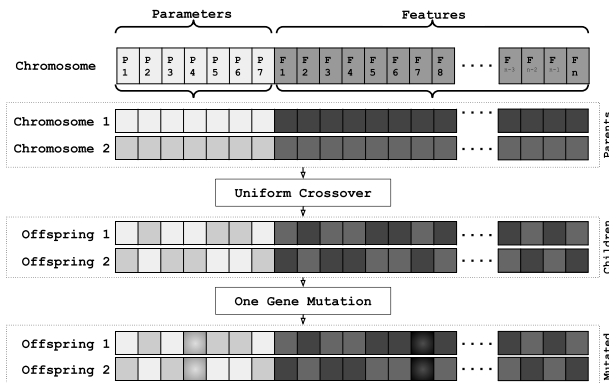
FIGURE 4. The detailed research methodology.

FIGURE 5. The chromosomes structure, crossover, and one gene mutation.

TABLE 2. TF-iDF vectorizer parameters.

Vectorizer Property	Value
Regular Expression Processing	r"[^A-Za-z0-9\-]\@*"
Stemmer	Porter
Stop Words	Not filtered
Max features to extract	Extract all features
Encoding	ISO-8859-1

TABLE 3. GA file-name/experiment code description.

Code	Description	Values
F	Percent of **F**eatures to select	1,5,10,20,30,40 %
P	No. **P**arents (First generation)	Rough tuning values
C	No. parents **C**rossover	Based on different crossover ratios *(the range for the first ten experiments: 10 to 100%)*
G	No. **G**enerations	100 for the first ten experiments, then 50 ,and for very large P values it was set to 5

TABLE 4. GA parameters description.

GA Parameter	Description
Perc2Sel	Percent of parents to select from the features set
CrossoverRatio	Percent of parents' crossover rate
NumFeat2Select	Derived value, depends on Perc2Sel to define the number of features to be selected
numberOfParentsMating	Derived value, depends on Crossover-Ratio to define the number of parents in crossover
numberOfXGBParameters	Number of classification algorithm parameters to be optimized (i.e., 7)
numberOfParameters	Chromosome size (i.e.,Total number of parameters to be optimized and max number of features to be selected which equals $numberOfXGBParameters + NumFeat2Select$)
numberOfGenerations	The number of generations that will be created

B. GA CONFIGURATION

Tables 3 and 4 show the GA parameters setup. The letters F, P, C, and G are used to summarize the description of each GA configuration (i.e., Metadata elements). F is the percent of features subset to be selected from the complete features set, P is the number of randomly selected parents in the first GA generation, C is the number of parents to crossover, and G is the number of GA generations. This standard file naming convention makes it easier to sort and interpret some effects of parameter tuning.

The best results of GA feature selection and hyper parameter optimization are selected based on fitness function

TABLE 5. XGBoost boosting parameters to be optimized by the modified GA.

XGB Parameter	Value Range	Description
learning_rate (eta)	0.01 - 1	Algorithm learning rate; lower the better but requires more iterations to find optimal solution.
n_estimators	10 - 1500	Maximum number of estimators
max_depth	1 - 10	Maximum tree depth, to control overfitting. (e.g.,high depth will bias the algorthim towards a specific sample)
min_child_weight	0.01 - 10.0	Minimum sum of observations in a child
gamma	0.01 - 10	Minimum reduction of loss when splitting
subsample	0.01 - 1.0	Random subset of observations for each tree
colsample_bytree	0.01 - 1.0	Subset of columns to be samples in the trees
seed	Fixed at 723	Used in parameter tuning and to have reproducible results

TABLE 6. Confusion Matrix.

		Predicted Tweet Class	
		Spam	Ham
Actual Tweet Class	Spam	TP	FN
	Ham	FP	TN

(i.e., GMean); which consist of the selected features and optimized XGBoost parameters, which are listed in Table 5).

C. CLASSIFICATION PERFORMANCE ANALYSIS

Analyzing the performance of the classifier to demonstrate its learning capability in model development is an essential part of the modeling phase assessment. Therefore, several metrics illustrate the performance of the developed model in detecting a potential spam tweet (i.e., classifying the tweets into Spam and Ham).

A visual summary of the classification results is represented by a confusion matrix [71]. Such that a two dimensional table aggregates the counts of the labeled tweets by the developed classification model into correct (True) and incorrect (False) labels. The aggregated counts are denoted specifically as True Positive (TP), False Positive (FP), True Negative (TN), and False Negative (FN). In this work, the positive class (i.e., class of interest) is the Spam tweet, and the negative class is the Ham tweet. Consequently, the four aggregated counts in the confusion matrix are interpreted as follows: TP count represents the actual Spam tweets that are classified correctly as Spam, TN count is the number of actual Ham tweets that are classified correctly as Ham, FP count is the number of the actual Ham tweets that are classified incorrectly as Spam, and the number of actual Spam tweets that are classified incorrectly as Ham represents the FN count. TN and TP represent the goodness of the classification model in correctly predicting the class label, while FP and FN show the level of the possible confusion a prediction model may have. Table 6 represents the confusion matrix that is used to derive a number of Spam classifier performance evaluation metrics.

Part of the derived evaluation metrics are:

1) True Positive Rate (TPR): the ratio of the correctly classified Spam tweets (i.e., tweets predicted as Spam and they are actually a Spam) [71]. It is alternatively named as recall or sensitivity.

$$TPR = \frac{TP}{TP + FN} \tag{1}$$

2) True Negative Rate (TNR): the ratio of the correctly classified Ham tweets (i.e., tweets predicted as Ham and they are actually Ham) [71]. It is alternatively named as specificity.

$$TNR = \frac{TN}{FP + TN} \tag{2}$$

3) Positive Predictive Value (PPV). It is alternatively named as precision [71]:

$$PPV = \frac{TP}{TP + FP} \tag{3}$$

4) False Positive Rate (FPR): the probability of false alarm, Fall-out.

$$FPR = \frac{FP}{FP + TN} \tag{4}$$

5) Negative Predictive Value (NPV) [71]:

$$NPV = \frac{TN}{TN + FN} \tag{5}$$

6) F-Score ($F1$) [71]:

$$F1 = 2 \times \frac{TPR \times PPV}{TPR + PPV} \tag{6}$$

7) Total Accuracy ($Accuracy$): It is traditionally derived from the confusion matrix and it represents the correctly classified instances count divided by the total number of instances. Alternatively, accuracy is also referred to as success rate (i.e.,, the ratio of correctly classified instances). Equation 7 illustrates accuracy metric.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{7}$$

There are many concerns in using the total accuracy as a performance metric, more particularly in imbalanced datasets [72]–[74]. Usually the negative class is dominant and more frequent in real life. Consequently, the model building phase would have a higher tendency towards modeling better the patterns of the negative class. Such tendency makes less prediction power of the positive class; which is usually the class of interest. Same issue arises when considering Spam

tweets. While TNR tends to rise up, TPR tends to decline. Therefore, further evaluation metrics are advised here.

The Geometric Mean (*GMean*) and Area Under the Curve (*AUC*) are used commonly in evaluating the classifiers of imbalanced class distribution. GMean and AUC take into consideration the minority class and seek the balance between the classes in illustrating the model accuracy (i.e., class independent metrics). The GMean is calculated according to Equation 8; i.e., the square root of the recall of the positive class multiplied by recall of the negative class. The calculation of the GMean ensures unbiased behavior of the metric either in objective function evaluation or in performance evaluation. A higher GMean value indicates better performance of the classifier [62]–[64].

$$GMean = \sqrt{TPR \times TNR} \qquad (8)$$

Receiver Operating Characteristic Curve (*ROC*) is a metric that takes different threshold values and confront them with the corresponding probabilities (i.e., *TPR* and *FPR*). The *AUC* is generated by calculating the area under the *ROC*. Therefore, there is a positive correlation between the value of *AUC* and the diagnostic ability of the classification model [75].

V. RESULTS AND DISCUSSION

It takes a considerable amount of computation time to have a relatively robust tweets spam prediction model using GA. The proposed methodology is quite complex and strives to find an optimal subset of tweet features and classifier parameters simultaneously. In tweets pre-processing step, the TF-iDF vectorizer generates the maximum number of possible features per tweet along with their TF-iDF value (i.e., the total of 14343 features extracted from all tweets).

Due to the time complexity of each GA search process, an initial relatively small subset of features (i.e.,, 1%) is examined in order to study the performance behavior of the classifier and select the most appropriate algorithm configurations (i.e., sensitivity analysis). The performance analysis of the first 10 experiments relies on 1% of feature space, 10 initial population parents, and 100 generations. Next, the generations are fixed at 50 and the crossover ratio at 60% of parents. Finally, the effect of different feature subset size is examined (i.e., F = 1, 5, 10, 20, 30, and 40).

A number of selected features and parameters are used to 10 × 50 cross-validate prediction models. The performance metrics (GMean in particular) indicate promising capabilities of the modified GA in finding a subset of the feature space and optimizing the parameters of the classifier accordingly. The outcomes of this research are compared to related work in terms of dimensionality reduction.

It is worth noting that the performance metrics in this research are presented such that further comparison with existing or future research is possible. The following subsections show and discuss the findings in more detail.
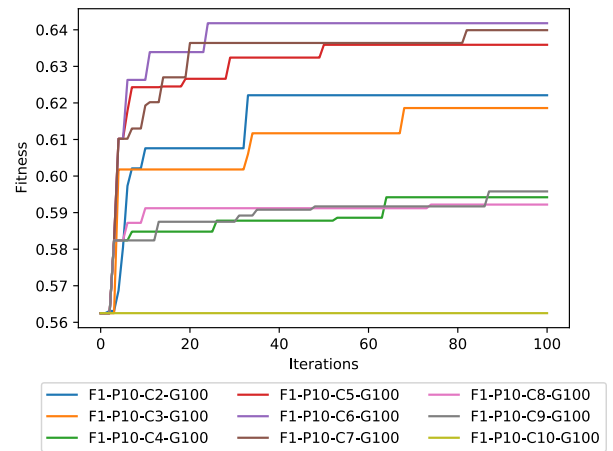

FIGURE 6. Sensitivity analysis: Number of iterations and crossover ratio by setting the parameters described in Table 3.

A. SENSITIVITY ANALYSIS

Sensitivity analysis results show the effect of GA crossover ratio on classification performance and its convergence. This is an important step to predict the behavior of the GA, and define the crossover ratio and number of GA iterations. Figure 6 represents the fitness curve of several crossover ratios ranging from 2% to 10%; while the remaining three parameters have been fixed at 1%of the total number of features, 10 parents, and 100 generations. Almost similar convergence pattern of the algorithm is present but a significant difference in fitness value groups the results into three main levels (i.e., Low, moderate, and high fitness value groups that correspond to different crossover ratios). Extreme values of crossover rate lead to very poor fitness values and lower prediction rate. Low values of crossover rate lead to relatively moderate fitness values. Moderate crossover values lead to relatively high fitness values, such that leveraging the prediction power of the spam classification models. It is apparent that 60% crossover rate is the most appropriate diversity factor to the next GA population. Experiments reveal that the number of iterations (i.e., number of GA populations) reaches a relatively local maximum early and slightly increases in consequent generations. Consequently, the number of appropriate generations is lowered to be 50 keeping in mind the effect on larger initial populations and number of features.

Several GA optimization and feature selection experiments (Table 7) aim at maximizing the fitness value. Some of the experiments cross over ratio (*C*) are fixed at 60%, 50 generations (*G*), and examine several percent of features to select (*F*). Furthermore, Table 8 lists some of the most frequently selected features in all the experiments and the presence of these top features in the best performing models.

B. MODEL VALIDATION

The selected features and the optimized parameters (i.e.,, the configurations that attained high fitness values) in some experiments are used to model spam prediction using

TABLE 7. Fitness value obtained by several GA feature selection and optimization experiments.

ExpID	Fitness	ExpID	Fitness	ExpID	Fitness
F1-P10-C1-G50	61.43%	F1-P250-C150-G50	75.19%	F1-P50-C5-G50	65.02%
F1-P10-C10-G100	56.25%	F1-P250-C5-G50	66.87%	F1-P500-C250-G50	81.13%
F1-P10-C2-G100	62.21%	F1-P300-C10-G50	71.96%	F1-P500-C300-G50	82.77%
F1-P10-C3-G100	61.86%	F1-P300-C100-G50	75.99%	F1-P500-C5-G50	75.96%
F1-P10-C4-G100	59.42%	F1-P300-C180-G50	77.92%	F1-P600-C360-G50	81.64%
F1-P10-C5-G100	63.59%	F1-P300-C20-G50	73.22%	F5-P300-C100-G50	77.65%
F1-P10-C6-G100	64.18%	F1-P300-C30-G50	74.68%	F10-P2870-C1722-G5	78.10%
F1-P10-C7-G100	63.99%	F1-P300-C40-G50	75.22%	F10-P300-C100-G50	80.33%
F1-P10-C8-G100	59.22%	F1-P300-C5-G50	76.64%	**F10-P400-C240-G50**	**84.85%**
F1-P10-C9-G100	59.58%	F1-P300-C50-G50	73.39%	F10-P5023-C3014-G5	80.22%
F1-P100-C5-G50	65.86%	F1-P350-C5-G50	73.86%	F20-P300-C100-G50	83.99%
F1-P100-C50-G50	73.28%	F1-P400-C240-G50	80.99%	F20-P5738-C3443-G5	80.69%
F1-P100-C60-G50	73.82%	F1-P400-C5-G50	71.01%	F30-P300-C100-G50	82.30%
F1-P150-C5-G50	71.69%	F1-P450-C5-G50	72.59%	F40-P300-C100-G50	83.88%
F1-P200-C120-G50	75.01%	F1-P50-C25-G50	68.31%	F40-P300-C100-G50	83.88%
F1-P200-C5-G50	74.47%	F1-P50-C30-G50	66.40%		

TABLE 8. Most frequent selected features and their presence in some experiments.

		ExpID: BestFitness:	F10-P400-C240-G50 84.85%	F20-P300-C100-G50 83.99%	F1-P400-C240-G50 80.99%	F1-P300-C180-G50 77.92%
FeatureText	FNo	Freq.				
co	2954	32	TRUE	TRUE	FALSE	TRUE
free	4885	30	TRUE	TRUE	TRUE	TRUE
enter	4264	28	TRUE	TRUE	TRUE	TRUE
shelv	11102	27	FALSE	FALSE	FALSE	TRUE
http	5920	26	TRUE	FALSE	TRUE	TRUE
porn	9685	26	TRUE	TRUE	TRUE	TRUE
election2juli	4177	25	FALSE	TRUE	FALSE	FALSE
foogo	4794	25	TRUE	FALSE	FALSE	FALSE
juju	6690	23	FALSE	TRUE	FALSE	FALSE
carr	2571	22	FALSE	TRUE	FALSE	FALSE
elimin	4190	22	FALSE	TRUE	FALSE	FALSE
lifetim	7304	22	TRUE	FALSE	FALSE	FALSE
mybodym	8300	22	FALSE	FALSE	FALSE	TRUE
overdose-revers	9194	22	FALSE	FALSE	FALSE	TRUE
rqanroxc47	10618	22	FALSE	TRUE	TRUE	TRUE
sabha	10713	22	FALSE	FALSE	FALSE	FALSE
wbdwyyxbgj	13620	22	FALSE	TRUE	FALSE	FALSE
xboxon	13976	22	FALSE	TRUE	FALSE	TRUE
clock	2910	21	FALSE	TRUE	FALSE	FALSE
continu	3131	21	FALSE	FALSE	FALSE	FALSE
darrellejon	3479	21	TRUE	FALSE	FALSE	FALSE
gtukhvetov4	5410	21	FALSE	FALSE	FALSE	FALSE
ibac	6021	21	FALSE	FALSE	FALSE	FALSE
luafkcop2r	7524	21	FALSE	FALSE	FALSE	FALSE
reachabl	10226	21	FALSE	FALSE	FALSE	FALSE
sportscent	11552	21	FALSE	FALSE	TRUE	FALSE
ude29	13014	21	FALSE	FALSE	FALSE	FALSE
z55nuvynrw	14229	21	FALSE	FALSE	FALSE	FALSE
zixqiz7fjq	14285	21	TRUE	TRUE	TRUE	TRUE

XGBoost. The model robustness is validated using 10-Fold cross-validation repeated 50 times. Table 9 summarizes the absolute difference between the GA fitness value and the validated XGBoost model performance.

The best optimization and feature selection experiment "F10-P400-C240-G50" attained a fitness value that equals 84.85% (i.e., Geometric Mean). The outcomes presented in Table 10 are used to build the spam prediction model that is validated using 10-fold stratified cross-validation and repeated 50 times.

XGBoost has been initialized with the optimized parameters in Table 10, and used the dataset with the selected features to build the spam classification model. The run is validated using 10-fold stratified cross-validation and repeated 50 times. Figure 7 is a box plot that summarizes the performance metrics, and Figure 8 depicts the performance metrics of each fold. The detailed performance metrics of the $10 \times 50CV$ are presented in Table 11.

There are two main observations worth mentioning here (a) the GA feature selection and parameter optimization

TABLE 9. Comparison between selected best fitness obtained by GA optimization and the results of 10-fold cross-validation repeated 50 times.

ExpID	Feat.	GA Fitness	GMean Avg. (SD)	Diff.
F1-P300-C180-G50	140	77.92	75.10 (0.033)	2.82
F1-P400-C240-G50	141	80.99	78.56 (0.031)	**2.43**
F20-P300-C100-G50	2638	83.99	80.60 (0.032)	3.39
F10-P400-C240-G50	1355	**84.85**	82.32 (**0.030**)	2.53

TABLE 10. Optimized XGBoost parameters and the number of selected features obtained by the experiment "F10-P400-C240-G50".

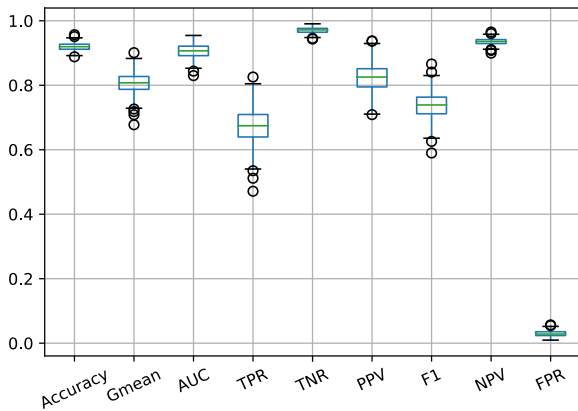| XGBoost Parameters	Optimized Value
learning_rate	0.47
n_estimators	93
max_depth	4
min_child_weight	0.08
gamma	0.42
subsample	0.94
colsample_bytree	0.84
Number of Selected Features	*1355*

FIGURE 7. Boxplot of the experiment "F10-P400-C240-G50" performance metrics.

effectiveness are noticeably acceptable, and (b) XGBoost classification model is barely affected by the algorithm randomness. The absolute difference between the GA fitness and the validated model is bound to an average of approximately 2.8, and the standard deviation of all the evaluation measures is less than 0.04 after 50 cross-validation runs.

C. STATISTICAL ANALYSIS

The effect of the stochastic nature of GA, and the imposed randomness of the classification algorithms on the experiments are described using the statistical tests. The whole process of GA-based parameter optimization and feature selection followed by a 10×50CV of XGBoost is repeated seven times. Runs denoted by the sequence "R01" to "R07" follow exactly the same steps as the run "F10-P400-C240-G50"; and to ensure non restricted random number generation the seed is not fixed to a certain value in any step.

TABLE 11. Results of experiment "F10-P400-C240-G50" repeated 50 times with 10-Fold cross-validation per each run. (Best fitness obtained by GA was GMean = 84.85%).

Metric	Min.	Avg.	Max.	SD
Accuracy	88.82	92.67	95.88	±0.010
GMean	72.71	82.32	91.26	±0.030
AUC	86.54	92.72	97.29	±0.018
TPR	54.02	69.68	84.88	±0.050
TNR	94.8	97.37	99.29	±0.008
PPV	70.27	84.54	95.59	±0.039
FPR	0.71	2.63	5.2	±0.008
F1	64.6	76.28	87.43	±0.037
NPV	91.19	94.02	96.97	±0.009

TABLE 12. Descriptive statistics of GMean values of different runs. (8 runs of "F10-P400-C240-G50" experiment, each is validated 50 × 10CV).

Run	Mean %	± SD	Min. %	25 %	50 %	75 %	Max %
R00	82.32	0.030	72.71	80.42	82.53	84.44	91.26
R01	81.48	0.031	67.83	79.58	81.47	83.50	92.10
R02	81.46	0.030	68.96	79.74	81.45	83.38	90.00
R03	81.57	0.028	72.02	79.66	81.65	83.47	90.19
R04	78.34	0.031	64.24	76.35	78.53	80.25	88.85
R05	81.10	0.030	70.98	79.43	81.21	83.02	89.89
R06	81.14	0.030	73.71	79.30	81.21	83.04	92.05
R07	82.46	0.029	72.35	80.64	82.26	84.35	90.40
Mean	**81.23**	**0.0300**	**70.35**	**79.39**	**81.29**	**83.18**	**90.59**

TABLE 13. p-Value of Wilcoxon statistical test between run pairs.

	R00	R01	R02	R03	R04	R05	R06
R01	0.0001						
R02	0.00002	0.76166					
R03	0.00013	0.56297	0.51467				
R04	0	0	0	0			
R05	0	0.05551	0.0756	0.03146	0		
R06	0	0.05079	0.05912	0.01797	0	0.98696	
R07	0.70065	0	0	0	0	0	0

The run "F10-P400-C240-G50" is the main baseline for the comparison and denoted by "R00". Table 12 lists the descriptive statistics of the GMean values of the runs. Only one of the runs "R04" showed a slight drift in the GMean value less than 2% below the average of all runs.

Further nonparametric statistical tests [25], [76]–[78] illustrate the level of similarity between the performance metrics of the runs. The tests analyse the GMean values of each run folds and the other runs. Namely, Wilcoxon Signed-Rank Test [25], [76], [77] between the run pairs and Kruskal statistical test [25], [76], [78] of run combinations.

The p-value of Wilcoxon statistical test between run pairs in Table 13 indicates a possible similarity between the runs "R00" and "R07"; considering $\alpha = 0.05$. The differences against the rest of the runs are possibly because the run "R00" is controlled in terms of a random number generation method. According to the Wilcoxon test, the runs "R01, R02, R03, R05, and R06" possibly have similar distributions, "R04" distribution is different from the others. However, the GMean value of the runs does not significantly differ as illustrated in Table 12.
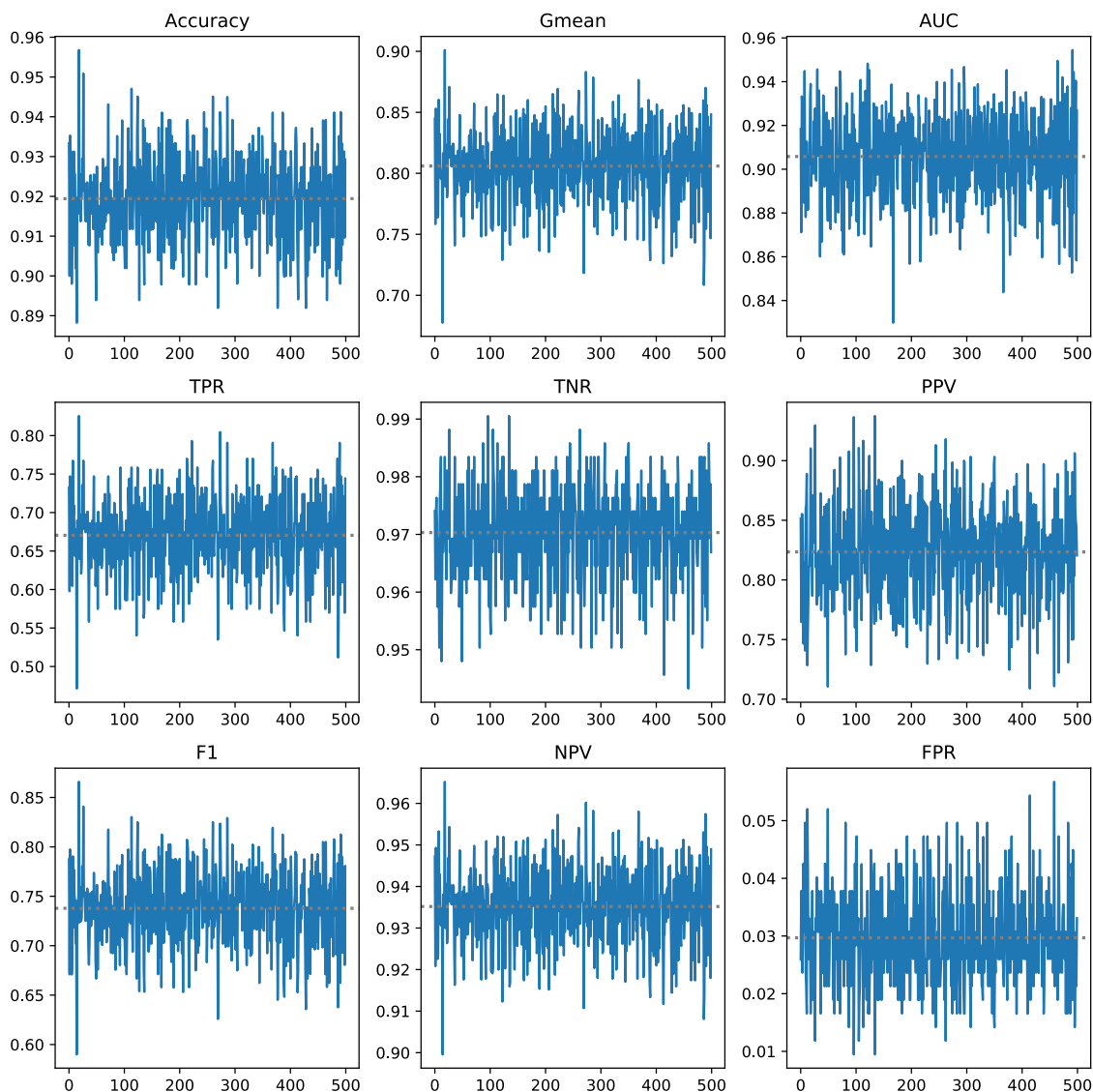
FIGURE 8. Performance metrics at each fold of the experiment "F10-P400-C240-G50".

Kruskal statistical test applies to three or more run combinations. Therefore, all run combinations are tested and the top p-values of the nonparametric Kruskal test are reported in Table 14. Assuming an $\alpha = 0.05$, the runs "R01, R02, and R05" are expected to have the highest similarity of GMean distributions, and "R01, R02, R03, R05, and R06" most probably have similar distributions.

D. COMPARISON ANALYSIS

1) COMPARISON WITH RELATED WORK

The overall performance and robustness of the proposed spam classification model outperforms the work in [22]. The authors in [22] claim a high performing LSTM (Long Short Term Memory) model, however the robustness of their approach is not well justified. The performance metrics are based on the "Ham" class as the positive class which is the majority class label. Swapping minority class with majority

TABLE 14. Kruskal statistical test of run combinations, showing only top combinations.

Run	p-Value
R01, R02, R05	0.14449
R01, R02, R03, R05	0.14072
R02, R05, R06	0.13411
R01, R05, R06	0.09977
R01, R02, R06	0.09401
R01, R02, R05, R06	0.09125
R01, R02, R03, R06	0.08424
R02, R03, R05	0.07825
R01, R03, R05	0.07208
R01, R02, R03, R05, R06	0.04973
R02, R03, R06	0.04358
R01, R03, R06	0.04145
R02, R03, R05, R06	0.03897
R01, R03, R05, R06	0.03371
R03, R05, R06	0.0312

class will lead to significantly higher performance metrics values; which is misleading when interpreting some metrics such as TPR. There is no evidence of using cross-validation

TABLE 15. Metrics re-calculation equations based on the selection of the positive class, for the sake of comparison. *In our research P denotes "Spam", and N denotes "Ham".*

Metric	Positive "Spam"	Positive "Ham"	Equivalence in our work
Recall	$\frac{TP}{TP+FN}$	$\frac{TN}{TN+FP}$	$1 - FPR$
Precision	$\frac{TP}{TP+FP}$	$\frac{TN}{TN+FN}$	NPV
F1	$2 \times \frac{TPR \times PPV}{TPR+PPV}$	$2 \times \frac{TNR \times NPV}{TNR+NPV}$	$2 \times \frac{TNR \times NPV}{TNR+NPV}$

TABLE 16. Comparison of the performance metrics in [22] with the best results of our experiments after re-calculating the performance metrics (i.e., considering "Ham" as the positive class).

Classifier	Precision	Recall	Accuracy	F1
KNN	91.61	91.96	91.96	91.38
NB	91.69	92.06	92.06	91.74
RF	93.25	93.43	93.43	93.04
ANN	91.80	91.18	91.18	91.41
SVM	92.91	93.14	93.14	92.97
SLSTM	95.54	98.37	95.09	96.84
Experiment F10-P400-C240-G50, GMean = 82.32 ± 0.030				
Avg+	94.02	97.37	92.67	95.67
Min+	91.19	94.8	88.82	
Max+	96.97	99.29	95.88	
SD	±0.009	±0.008	±0.010	

*$50 \times 10CV$
+Recalculated to consider "Ham" as positive class
according to the definitions in Table 15

TABLE 17. Effect of feature reduction in [22] compared to some experiments in our research.

Frequency-based feature reduction in [22]		
Features	Accuracy %	Notes
5000	93.81	
8000	93.91	
10,000	94.21	
14,000	95.09	
Some experiments in our research $(50 \times 10CV)$		
Features	Accuracy% ±SD (maximum)	Experiment ID
1355	92.67 ± 0.010 (95.88)	F10-P400-C240-G50
2638	91.94 ± 0.011 (95.68)	F20-P300-C100-G50
141	91.66 ± 0.010 (94.71)	F1-P400-C240-G50
140	90.61 ± 0.010 (93.53)	F1-P300-C180-G50
142	88.57 ± 0.011 (91.75)	F1-P100-C60-G50
144	84.95 ± 0.012 (88.24)	F1-P10-C6-G100

TABLE 18. p-Value of Wilcoxon statistical comparing GMean of "F10-P400-C240-G50" with Chi^2 feature selection method performance.

Experiment	p-Value
XGBoost & Chi^2	0.00000
MNNB & Chi^2	0.00000
KNN & Chi^2	0.00000
LR & Chi^2	0.00000
AdaBoost & Chi^2	0.00000
DT & Chi^2	0.51423

in assessing the model robustness. Features are reduced by the selection of most frequent words in the corpus, and the configuration of the basic classifiers used in performance comparison is not presented in the paper. For the sake of fair comparison with the results of our research, the performance metrics are re-calculated to consider "Ham" as the positive class label. Table 15 shows the equivalence equations used to re-calculate the results of our experiments to be comparable with the results in [22]. Nonetheless, Table 16 compares performance metrics of [22] in spam classification (having "Ham" as positive class, and "Spam" as negative class) and Table 17 confronts the effect of feature reduction.

The modified GA approach in this study outperforms the approach used in [22] in terms of feature selection, The number of features selected in the proposed approach (i.e., 1355 features) is much lower than the number of features selected in [22] (i.e.,, 5000 features). On the other hand, the maximum accuracy obtained using GA in our approach was 95.88% compared to 95.09% in [22].

2) COMPARISON WITH CHI^2 FEATURE SELECTION

Chi^2 [79], [80] statistical test has been used in text feature selection based on statistical significance of features. We selected the top "1355 features" using the Chi^2 method to compare the results with our best findings. Different machine learning algorithms are validated by $10 \times 50CV$ in building spam prediction models using "1355" selected features by Chi^2. The algorithms used without parameter tuning are XGBoost, Multinomial Naive Bayes (MNNB), K-Nearest Neighbors (KNN), Logistic Regression (LR), Adaptive Boosting (AdaBoost), and Decision Trees (DT). The p-value of Wilcoxon statistical test between the run "F10-P400-C240-G50" and Chi^2-based models are presented in Table 18, and the descriptive statistics of the performance metrics are shown in Table 19.

The p-value of the Wilcoxon test indicates different distributions of GMean value compared to the run "F10-P400-C240-G50". except for the DT model. The similar distributions are justified by the fact that XGBoost is an evolution of the decision tree algorithm; they share similar characteristics that could lead to similar behavior. However, "F10-P400-C240-G50" model outperforms the DT model as indicated by the majority of the performance metrics.

3) COMPARISON WITH PCA FEATURE SELECTION

Reducing the dimension of relatively large feature space while preserving most of the information is possible using Principal Component Analysis (PCA) [81]. The feature set is transformed into a number of principal components based on their covariance matrix, then a relatively small number of the principal components will be selected to represent the full feature set. We trained an XGBoost model using a different number of principal components (i.e., from 1 to 20) and validated the model using $50 \times 10CV$. An illustration

TABLE 19. Comparison between the run 'F10-P400-C240-G50' with Chi^2 feature selection method performance. (each is validated by 50 × 10CV).

| Metric	F10-P400-C240-G50	XGBoost & Chi^2	MNNB & Chi^2	KNN3 & Chi^2	LR & Chi^2	AdaBoost & Chi^2	DT & Chi^2
Accuracy	**92.67** (0.010)	92.06 (0.009)	88.77 (0.009)	87.17 (0.009)	91.34 (0.009)	92.46 (0.010)	91.38 (0.011)
GMean	**82.32** (0.030)	76.12 (0.032)	58.13 (0.044)	50.88 (0.049)	71.46 (0.037)	80.60 (0.030)	82.20 (0.028)
AUC	92.72 (0.018)	93.79 (0.016)	**95.61** (0.012)	77.13 (0.029)	95.39 (0.012)	92.82 (0.018)	83.31 (0.024)
TPR	69.68 (0.050)	58.71 (0.049)	33.99 (0.051)	26.23 (0.050)	51.46 (0.054)	66.55 (0.049)	**70.77** (0.047)
TNR	97.37 (0.008)	98.88 (0.005)	**99.96** (0.001)	99.63 (0.003)	99.50 (0.003)	97.76 (0.007)	95.59 (0.010)
PPV	84.54 (0.039)	91.58 (0.035)	**99.49** (0.012)	93.61 (0.051)	95.51 (0.028)	86.00 (0.041)	76.81 (0.042)
FPR	2.63 (0.008)	1.12 (0.005)	**0.04** (0.001)	0.37 (0.003)	0.50 (0.003)	2.24 (0.007)	4.41 (0.010)
F1s	**76.28** (0.037)	71.42 (0.039)	50.46 (0.057)	40.73 (0.063)	66.71 (0.047)	74.9 (0.038)	73.56 (0.035)
NPV	94.02 (0.009)	92.14 (0.009)	88.11 (0.008)	86.86 (0.008)	90.94 (0.009)	93.47 (0.009)	**94.12** (0.009)
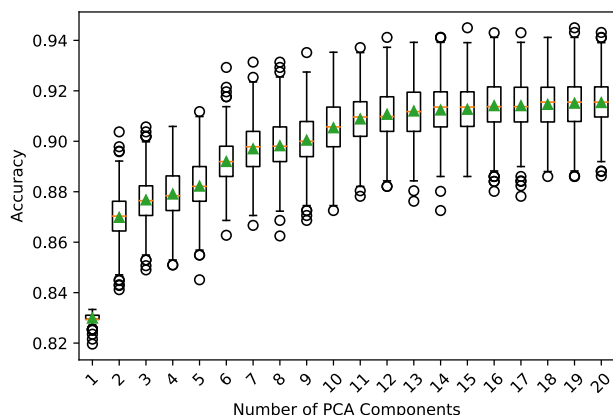
FIGURE 9. PCA feature selection and XGBoost. (Validated 50 × 10CV).

TABLE 20. Accuracy of PCA feature selection and XGBoost. (each is validated by 50 × 10CV).

PCA Components	Accuracy %	SD ±
01	82.99	0.001
02	86.98	0.009
03	87.68	0.009
04	87.91	0.010
05	88.22	0.010
06	89.20	0.010
07	89.70	0.010
08	89.82	0.010
09	90.04	0.011
10	90.53	0.011
11	90.88	0.011
12	91.07	0.010
13	91.20	0.011
14	91.24	0.011
15	91.27	0.010
16	91.41	0.010
17	91.41	0.010
18	91.46	0.011
19	91.51	0.010
20	91.53	0.010

of the model accuracy in relation to the number of principal components is shown in Figure 9. The accuracy metrics and the standard deviation are presented in Table 20.

It is apparent that 20 principal components will enable attaining 91.53% total accuracy in spam prediction; compared to 92.67% using our modified GA. Moreover, PCA will reduce significantly the feature space but makes the model interpretation much harder. It is apparent that the PCA based

TABLE 21. Accuracy of BERT model and DL. (20 epochs).

class	PPV (precision)	TPR (recall)	f1	support
'ham'	0.91	0.97	0.94	1058
'spam'	0.79	0.52	0.63	216
accuracy			**0.90**	1274
macro avg	0.85	0.75	0.78	1274
weighted avg	0.89	0.90	0.89	1274

model will converge after 15 PCA components. According to the illustration in Figure 9 and Table 20 the improvement in accuracy was less than 0.5 percentage absolute point within the last nine components (i.e., PCA components 12-20).

Therefore, PCA-based XGBoost models under-perform the modified GA-based XGboost models.

4) COMPARISON WITH BERT AND DEEP LEARNING

Most recent advancements in natural language processing research introduced pre-trained word embedding models that are coupled with Deep Learning (DL) algorithms [14], [23], [82]. BERT word embedding is used with DL to build spam classification models. The major issue of interest in DL is the computational complexity and extensive resource use. Despite such limitations, we were able to build a spam prediction model in this research and validated with the percentage split of the tweets dataset. BERT is used in text pre-processing and encoding, ''sigmoid'' activation function, Tensorflow [83], and Keras [84]. Table 21 summarizes the major performance metrics of the generated model over 20 epochs. The TPR of the class of interest (''Spam'') is relatively low (52%) which indicates a very low prediction power of the generated model in spam prediction.

For the sake of comparison with recent advancement in text classification we implemented the BERT-DL model to assess its feasibility in spam prediction. In our case, the limited computational resources were the main barrier in tuning and seeking better prediction performance. However, the experiment shows that our modified GA approach outperforms the DL approach. Moreover, the resulting models are not intuitive to be interpreted and starve for computational resources.

5) EXPERIMENTING WITH SMS DATASET

The modified GA is applied to a public imbalanced SMS dataset [85]; about 13% of the messages are ''Spam''. Hyper parameter optimization and feature selection results are listed

TABLE 22. Optimized XGBoost parameters and the number of selected features obtained by the "SMS Spam Dataset" experiment using modified GA and XGBoost.

XGBoost Parameters	Optimized Value
learning_rate	0.31
n_estimators	47
max_depth	10
min_child_weight	0.15
gamma	0.33
subsample	1.0
colsample_bytree	0.69
Number of Selected Features	*706*

TABLE 23. Results of "SMS Spam Dataset" using modified GA and XGBoost; repeated 50 times with 10-Fold cross-validation per each run. (Best fitness obtained by GA was GMean = 97.29%).

Metric	Min.	Avg.	Max.	SD
Accuracy	95.15	97.4	99.1	0.006
GMean	85.94	92.49	98.35	0.021
AUC	93.08	97.99	99.85	0.011
TPR	74.32	86.35	97.33	0.039
TNR	97.72	99.12	100	0.004
PPV	84.29	93.88	100	0.028
FPR	0	0.88	2.28	0.004
F1s	81.38	89.89	96.69	0.025
NPV	96.18	97.92	99.59	0.006

in Table 22. The GA reduced the selected features to 9.52% of total dataset features (i.e., 706 out of 7419 features) attained a GMean value of 97.29%.

The outcomes in Table 22 initialized an XGBoost classifier to model SMS spam. The model is validated using a 50 times repeated run of 10-fold stratified cross-validation. Table 23 shows the performance metrics.

In comparison to the best results in [86]–[88] the modified GA shows a competitive performance. The authors in [86], [87], and [88] attained maximum of total accuracy equals 96%, 96.8% and 98.74% respectively. Its worth mentioning that Random Forest and SVM algorithms attained 99% accuracy in [86] but with TF-iDF features and oversampling. In essence, the maximum accuracy attained by the modified GA was 99.1%; which makes it outperform the majority of the related works utilizing the same SMS dataset.

Our proposed approach, modified GA, reduces all the features of the tweets by 9.45% (i.e., from 14343 to 1355 features) and maintains a competitive performance in comparison to the related studies. Therefore, the proposed approach in this research is expected to reduce the dimensionality by automating the process of feature selection and tuning the prediction model parameters simultaneously. The results of this work could be extended to list the features as words (i.e., specific words in the tweets) for further feature analysis. Furthermore, the XGBoost tree models have a higher level of interpretability compared to ANN-based models; which make it much easier to deeply analyse the models for the sake of spam understanding and modeling.

E. IMPLICATIONS AND LIMITATIONS

The reported experiments and outcomes of this research establish a basis for spam modeling. In essence, it outperformed many related works in SMS spam modeling. Future research may build on the outcomes to enhance understanding of spam behavior. Further, this research could be considered for generalization in other domains such as software engineering, construction engineering, internet of things, and smart cities. In contrast to black-box models, tree-based classifiers enable straightforward implementation to detect spam tweets. The tremendous growth of Online Social Networks (OSN) calls for efficient real-time spam detectors. The state of the art solutions recommend Deep Learning based solutions, however deep learning is resource consuming and overlooks unseen spam behaviors. Our proposed approach reduces learning time significantly compared to deep learning based solutions.

Usually, GA finds outstanding solutions once its parameters (i.e., Initial population, mutation and cross-over ratios, number of generations, ...) are well tuned. In this research tuning the GA using Grid Search is time consuming. Therefore, sensitivity analysis described in section III-D3 has been used to find the best GA parameters. In the near future we expect the reliability of the public twitter spam dataset to raise concerns due to subjective interpretations by different communities. Multi class labeling of spam text in sentiment analysis is not considered in this research.

The large number of experiment runs and the comprehensive set of performance metrics would direct further research activities.

VI. CONCLUSION AND FUTURE DIRECTIONS

Spam modeling is a challenging task due to many issues such as the high dimensionality of the features space, the imbalanced class distributions, the bias of classification algorithms towards the majority class, and natural language processing issues. Many of the related research works lack solid validation of the generated models and usually report positive class-based performance metrics. In this paper, a modified genetic algorithm is designed in order to perform two main tasks; (1) an effective dimensionality reduction of an imbalanced tweets dataset and (2) hyper parameter optimization of XGBoost classification algorithm. Intensive validation of the generated prediction model illustrates the robustness of the modified algorithm and its competitive performance compared to other approaches. This research reports a comprehensive set of performance metrics and nonparametric statistical significance tests; which makes it easier to understand the outcomes and provide a basis for comparisons with related works. In tweets spam modeling, the proposed approach selected less than 10% of features to attain on average 92.67% and 82.32% total accuracy and geometric mean respectively. It outperformed the performance of Chi^2 and PCA based approaches in feature selection. In addition, it showed competitive performance compared to recent machine learning algorithms; including word embedding and deep learning based models.

The stochastic aspects of genetic algorithms, and parameter optimization are among the research limitations. Genetic algorithm based solutions usually require a large number of

initial population space or large number of generations to find an outperforming solution. The large number of experiment runs and the comprehensive set of performance metrics would direct further research activities. There are many unexplored issues by this research; issues include parallel processing to reduce time complexity of the approach, the effect of natural language processing on improving the accuracy, incorporating user account features in spam modeling, and experimenting with multi-language spam modeling. Further research that may build on the modified genetic algorithm to tackle different problems or domains such as sentiment analysis and multi-class modeling.

REFERENCES

[1] UTKML Electronic and Kaggle. (2019). *UtkMl's Twitter Spam Detection Competition*. [Online]. Available: https://www.kaggle.com/c/twitter-spam

[2] X. Liu, H. Lu, and A. Nayak, "A spam transformer model for SMS spam detection," *IEEE Access*, vol. 9, pp. 80253–80263, 2021.

[3] K. Chowdhury, "Spam identification on Facebook, Twitter and email using machine learning," *Central Eur. Res. J.*, vol. 6, no. 1, pp. 18–26, 2020.

[4] *Platform Manipulation and Spam Policy*, Twitter, Electron., San Francisco, CA, USA, Sep. 2020.

[5] T. Wu, S. Wen, Y. Xiang, and W. Zhou, "Twitter spam detection: Survey of new approaches and comparative study," *Comput. Secur.*, vol. 76, pp. 265–284, Jul. 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S016740481730250X

[6] B. Markines, C. Cattuto, and F. Menczer, "Social spam detection," in *Proc. 5th Int. Workshop Adversarial Inf. Retr. Web (AIRWeb)*. New York, NY, USA: Association for Computing Machinery, 2009, pp. 41–48, doi: 10.1145/1531914.1531924.

[7] B. Mukunthan and M. Arunkrishna, "Spam detection and spammer behaviour analysis in Twitter using content based filtering approach," *J. Phys., Conf. Ser.*, vol. 1817, no. 1, Mar. 2021, Art. no. 012014.

[8] H. Gao, Y. Chen, K. Lee, D. Palsetia, and A. N. Choudhary, "Towards online spam filtering in social networks," in *Proc. NDSS*, vol. 12, 2012, pp. 1–16.

[9] Z. Alom, B. Carminati, and E. Ferrari, "A deep learning model for Twitter spam detection," *Online Social Netw. Media*, vol. 18, Jul. 2020, Art. no. 100079. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2468696420300203

[10] B. Alatas and H. Bingol, "A physics based novel approach for travelling tournament problem: Optics inspired optimization," *Inf. Technol. Control*, vol. 48, no. 3, pp. 373–388, Sep. 2019.

[11] B. Alatas and H. Bingol, "Comparative assessment of light-based intelligent search and optimization algorithms," *Light Eng.*, vol. 28, no. 6, pp. 51–59, 2020.

[12] S. Akyol and B. Alatas, "Plant intelligence based metaheuristic optimization algorithms," *Artif. Intell. Rev.*, vol. 47, no. 4, pp. 417–462, 2017.

[13] S. Kaddoura, G. Chandrasekaran, D. Elena Popescu, and J. H. Duraisamy, "A systematic literature review on spam content detection and classification," *PeerJ Comput. Sci.*, vol. 8, p. e830, Jan. 2022.

[14] D. Borse and S. Borse, "State of the art on Twitter spam detection," in *ICCET 2022: Applied Computational Technologies*, B. Iyer, T. Crick, and S.-L. Peng, Eds. Singapore: Springer, 2022, pp. 486–496.

[15] I. P. Benitez, A. M. Sison, and R. P. Medina, "An improved genetic algorithm for feature selection in the classification of disaster-related Twitter messages," in *Proc. IEEE Symp. Comput. Appl. Ind. Electron. (ISCAIE)*, Apr. 2018, pp. 238–243.

[16] H. Utama, "Sentiment analysis in airline tweets using mutual information for feature selection," in *Proc. 4th Int. Conf. Inf. Technol., Inf. Syst. Electr. Eng. (ICITISEE)*, Nov. 2019, pp. 295–300.

[17] J. D. Prusa, T. M. Khoshgoftaar, and D. J. Dittman, "Impact of feature selection techniques for tweet sentiment classification," in *Proc. 28th Int. Flairs Conf.*, 2015, pp. 1–6.

[18] A. Jain and V. Jain, "Sentiment classification of Twitter data belonging to renewable energy using machine learning," *J. Inf. Optim. Sci.*, vol. 40, no. 2, pp. 521–533, Feb. 2019.

[19] Z. Wang and Z. Lin, "Optimal feature selection for learning-based algorithms for sentiment classification," *Cogn. Comput.*, vol. 12, no. 1, pp. 238–248, Jan. 2020.

[20] H. Saif, Y. He, and H. Alani, "Alleviating data sparsity for Twitter sentiment analysis," in *Proc. CEUR Workshop*, 2012, pp. 1–8. [Online]. Available: https://CEUR-WS.org

[21] N. K. Suchetha, A. Nikhil, and P. Hrudya, "Comparing the wrapper feature selection evaluators on Twitter sentiment classification," in *Proc. Int. Conf. Comput. Intell. Data Sci. (ICCIDS)*, Feb. 2019, pp. 1–6.

[22] G. Jain, M. Sharma, and B. Agarwal, "Optimizing semantic LSTM for spam detection," *Int. J. Inf. Technol.*, vol. 11, no. 2, pp. 239–250, Jun. 2019.

[23] K. Taneja and J. Vashishtha, "Comparison of transfer learning and traditional machine learning approach for text classification," in *Proc. 9th Int. Conf. Comput. Sustain. Global Develop. (INDIACom)*, Mar. 2022, pp. 195–200.

[24] H. Al-Rubaiee, R. Qiu, K. Alomar, and D. Li, "Sentiment analysis of Arabic tweets in e-Learning," *J. Comput. Sci.*, vol. 12, no. 11, pp. 553–563, Nov. 2016. [Online]. Available: http://hdl.handle.net/10547/623004

[25] A. W. Whitney, "A direct method of nonparametric measurement selection," *IEEE Trans. Comput.*, vol. C-20, no. 9, pp. 1100–1103, Sep. 1971.

[26] T. Marill and D. Green, "On the effectiveness of receptors in recognition systems," *IEEE Trans. Inf. Theory*, vol. IT-9, no. 1, pp. 11–17, Jan. 1963.

[27] D. B. Fogel, *Evolutionary Computation: The Fossil Record*. Hoboken, NJ, USA: Wiley, 1998.

[28] M. Taherdangkoo, M. Paziresh, M. Yazdi, and M. Bagheri, "An efficient algorithm for function optimization: Modified stem cells algorithm," *Open Eng.*, vol. 3, no. 1, pp. 36–50, Mar. 2013.

[29] J. Zhang, H. S.-H. Chung, and W.-L. Lo, "Clustering-based adaptive crossover and mutation probabilities for genetic algorithms," *IEEE Trans. Evol. Comput.*, vol. 11, no. 3, pp. 326–335, Jun. 2007.

[30] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proc. Int. Conf. Neural Netw. (ICNN)*, vol. 4, 1995, pp. 1942–1948.

[31] M. Dorigo, M. Birattari, and T. Stutzle, "Ant colony optimization," *IEEE Comput. Intell. Mag.*, vol. 1, no. 4, pp. 28–39, Nov. 2006.

[32] R. Leardi, "Application of genetic algorithm–PLS for feature selection in spectral data sets," *J. Chemometrics, J. Chemometrics Soc.*, vol. 14, nos. 5–6, pp. 643–655, 2000.

[33] G. Wei, J. Zhao, Y. Feng, A. He, and J. Yu, "A novel hybrid feature selection method based on dynamic feature importance," *Appl. Soft Comput.*, vol. 93, Aug. 2020, Art. no. 106337.

[34] H. Liu, L. Liu, and H. Zhang, "Feature selection using mutual information: An experimental study," in *PRICAI 2008: Trends in Artificial Intelligence*, T.-B. Ho and Z.-H. Zhou, Eds. Berlin, Germany: Springer, 2008, pp. 235–246.

[35] G. Forman, "An extensive empirical study of feature selection metrics for text classification," *J. Mach. Learn. Res.*, vol. 3, pp. 1289–1305, Mar. 2003.

[36] A. Fraser and D. Burnell, *Computer Models in Genetics*. New York, NY, USA: McGraw-Hill, 1970.

[37] I.-S. Oh, J.-S. Lee, and B.-R. Moon, "Hybrid genetic algorithms for feature selection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 26, no. 11, pp. 1424–1437, Nov. 2004.

[38] B. Baudry, F. Fleurey, J.-M. Jézéquel, and Y. Le Traon, "Automatic test case optimization: A bacteriologic algorithm," *IEEE Softw.*, vol. 22, no. 2, pp. 76–82, Mar./Apr. 2005.

[39] R. Akbari and K. Ziarati, "A multilevel evolutionary algorithm for optimizing numerical functions," *Int. J. Ind. Eng. Comput.*, vol. 2, no. 2, pp. 419–430, Apr. 2011.

[40] F. Hadizadeh, S. Vahdani, and M. Jafarpour, "Quantitative structure-activity relationship studies of 4-imidazolyl-1,4-dihydropyridines as calcium channel blockers," *Iranian J. Basic Med. Sci.*, vol. 16, no. 8, p. 910, 2013.

[41] O. Babatunde, L. Armstrong, J. Leng, and D. Diepeveen, "A genetic algorithm-based feature selection," *Int. J. Electron. Commun. Comput. Eng.*, vol. 5, pp. 889–905, Jul. 2014. [Online]. Available: https://www.ijecce.org/index.php/component/jresearch/?view=publication&task=show&id=866&Itemid=437

[42] J. Huang, Y. Cai, and X. Xu, "A hybrid genetic algorithm for feature selection wrapper based on mutual information," *Pattern Recognit. Lett.*, vol. 28, no. 13, pp. 1825–1844, May 2007.

[43] F. Iqbal, J. M. Hashmi, B. C. M. Fung, R. Batool, A. M. Khattak, S. Aleem, and P. C. K. Hung, "A hybrid framework for sentiment analysis using genetic algorithm based feature reduction," *IEEE Access*, vol. 7, pp. 14637–14652, 2019.

[44] S. Aalaei, H. Shahraki, A. Rowhanimanesh, and S. Eslami, "Feature selection using genetic algorithm for breast cancer diagnosis: Experiment on three different datasets," *Iranian J. Basic Med. Sci.*, vol. 19, no. 5, p. 476, 2016.

[45] V. Kumar and S. Minz, "Mood classifiaction of lyrics using Senti-WordNet," in *Proc. Int. Conf. Comput. Commun. Informat.*, Jan. 2013, pp. 1–5.

[46] J. G. Dy, C. E. Brodley, A. Kak, L. S. Broderick, and A. M. Aisen, "Unsupervised feature selection applied to content-based retrieval of lung images," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 25, no. 3, pp. 373–378, Mar. 2003.

[47] W. Lee, S. J. Stolfo, and K. W. Mok, "Adaptive intrusion detection: A data mining approach," *Artif. Intell. Rev.*, vol. 14, no. 6, pp. 533–567, 2000.

[48] J. H. Holland, "Genetic algorithms," *Sci. Amer.*, vol. 267, no. 1, pp. 66–73, 1992.

[49] D. Javaheri, P. Lalbakhsh, and M. Hosseinzadeh, "A novel method for detecting future generations of targeted and metamorphic malware based on genetic algorithm," *IEEE Access*, vol. 9, pp. 69951–69970, 2021.

[50] J. Jeyaranjani and D. Devaraj, "Improved genetic algorithm for optimal demand response in smart grid," *Sustain. Comput., Informat. Syst.*, vol. 35, Sep. 2022, Art. no. 100710. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S221053792200049X

[51] C. Peng, X. Wu, W. Yuan, X. Zhang, Y. Zhang, and Y. Li, "MGRFE: Multilayer recursive feature elimination based on an embedded genetic algorithm for cancer classification," *IEEE/ACM Trans. Comput. Biol. Bioinf.*, vol. 18, no. 2, pp. 621–632, Mar. 2021.

[52] F. Amini and G. Hu, "A two-layer feature selection method using genetic algorithm and elastic net," *Expert Syst. Appl.*, vol. 166, Mar. 2021, Art. no. 114072. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0957417420308320

[53] T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, New York, NY, USA, Aug. 2016, pp. 785–794. [Online]. Available: http://doi.acm.org/10.1145/2939672.2939785

[54] A. Occhipinti, L. Rogers, and C. Angione, "A pipeline and comparative study of 12 machine learning models for text classification," *Expert Syst. Appl.*, vol. 201, Sep. 2022, Art. no. 117193. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0957417422005802

[55] S. Pafka. (2019). *Benchm-ML*. [Online]. Available: https://github.com/szilard/benchm-ml

[56] V. Morde and V. A. Setty. (Apr. 2019). *XGBoost Algorithm: Long May She Reign!* [Online]. Available: https://towardsdatascience.com/vishalmorde-xgboost-algorithm-long-she-may-rein-edd9f99be63d

[57] D. Nielsen, "Tree boosting with XGBoost-why does XGBoost win 'every' machine learning competition?" M.S. thesis, Dept. Math. Sci., Norwegian Univ. Sci. Technol., Trondheim, Norway, 2016. [Online]. Available: https://ntnuopen.ntnu.no/ntnu-xmlui/handle/11250/2433761

[58] S. Pan, Z. Zheng, Z. Guo, and H. Luo, "An optimized XGBoost method for predicting reservoir porosity using petrophysical logs," *J. Petroleum Sci. Eng.*, vol. 208, Jan. 2022, Art. no. 109520. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0920410521011608

[59] N. T. T. Linh, M. Pandey, S. Janizadeh, G. S. Bhunia, A. Norouzi, S. Ali, Q. B. Pham, D. T. Anh, and K. Ahmadi, "Flood susceptibility modeling based on new hybrid intelligence model: Optimization of XGboost model using GA Metaheuristic algorithm," *Adv. Space Res.*, vol. 69, no. 9, pp. 3301–3318, May 2022.

[60] T. Kavzoglu and A. Teke, "Advanced hyperparameter optimization for improved spatial prediction of shallow landslides using extreme gradient boosting (XGBoost)," *Bull. Eng. Geol. Environ.*, vol. 81, no. 5, pp. 1–22, May 2022.

[61] X. Deng, M. Li, S. Deng, and L. Wang, "Hybrid gene selection approach using XGBoost and multi-objective genetic algorithm for cancer classification," *Med. Biol. Eng. Comput.*, vol. 60, no. 3, pp. 663–681, Mar. 2022.

[62] D. Tao, X. Li, X. Wu, and S. J. Maybank, "Geometric mean for subspace selection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 2, pp. 260–274, Feb. 2009.

[63] M.-J. Kim, D.-K. Kang, and H. B. Kim, "Geometric mean based boosting algorithm with over-sampling to resolve data imbalance problem for bankruptcy prediction," *Expert Syst. Appl.*, vol. 42, no. 3, pp. 1074–1082, Feb. 2015.

[64] J. Akosa, "Predictive accuracy: A misleading performance measure for highly imbalanced data," in *Proc. SAS Global Forum*, vol. 12, 2017, pp. 1–4.

[65] M. F. Porter, "An algorithm for suffix stripping," *Program*, vol. 14, no. 3, pp. 130–137, 1980.

[66] G. Salton and M. J. McGill, *Introduction to Modern Information Retrieval*. New York, NY, USA: McGraw-Hill, 1986.

[67] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *J. Mach. Learn. Res.*, vol. 12, pp. 2825–2830, Nov. 2011.

[68] J. Chen, F. Zhao, Y. Sun, and Y. Yin, "Improved XGBoost model based on genetic algorithm," *Int. J. Comput. Appl. Technol.*, vol. 62, no. 3, pp. 240–245, 2020.

[69] Y. Jiang, G. Tong, H. Yin, and N. Xiong, "A pedestrian detection method based on genetic algorithm for optimize XGBoost training parameters," *IEEE Access*, vol. 7, pp. 118310–118321, 2019.

[70] A. Saltelli, "Sensitivity analysis for importance assessment," *Risk Anal.*, vol. 22, no. 3, pp. 579–590, Jun. 2002.

[71] C. Sammut and G. I. Webb, Eds., *Encyclopedia of Machine Learning and Data Mining* (Springer Reference), 2nd ed. New York, NY, USA: Springer, 2017.

[72] N. V. Chawla, "Data mining for imbalanced datasets: An overview," in *Data Mining and Knowledge Discovery Handbook*, O. Maimon and L. Rokach, Eds. Boston, MA, USA: Springer, 2005, pp. 853–867, doi: 10.1007/0-387-25465-X_40.

[73] N. Ghatasheh, H. Faris, R. Abukhurma, P. A. Castillo, N. Al-Madi, A. M. Mora, A. M. Al-Zoubi, and A. Hassanat, "Cost-sensitive ensemble methods for bankruptcy prediction in a highly imbalanced data distribution: A real case from the Spanish market," *Prog. Artif. Intell.*, vol. 9, no. 4, pp. 361–375, Dec. 2020.

[74] N. Ghatasheh, H. Faris, I. AlTaharwa, Y. Harb, and A. Harb, "Business analytics in telemarketing: Cost-sensitive analysis of bank campaigns using artificial neural networks," *Appl. Sci.*, vol. 10, no. 7, p. 2581, Apr. 2020.

[75] J. A. Hanley and B. J. McNeil, "The meaning and use of the area under a receiver operating characteristic (ROC) curve," *Radiology*, vol. 143, no. 1, pp. 29–36, 1982.

[76] J. Brownlee, *Statistical Methods for Machine Learning: Discover How to Transform Data Into Knowledge with Python*. Machine Learning Mastery, 2018. [Online]. Available: https://machinelearningmastery.com/statistics_for_machine_learning/ and https://books.google.com.pe/books?id=386nDwAAQBAJ

[77] R. F. Woolson, *Wilcoxon Signed-Rank Test*. Hoboken, NJ, USA: Wiley, 2008, pp. 1–3. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/9780471462422.eoct979

[78] Y. Chan and R. P. Walmsley, "Learning and understanding the Kruskal–Wallis one-way analysis-of-variance-by-ranks test for differences among three or more independent groups," *Phys. Therapy*, vol. 77, no. 12, pp. 1755–1761, Dec. 1997.

[79] R. Cekik and A. K. Uysal, "A new metric for feature selection on short text datasets," *Concurrency Comput., Pract. Exp.*, vol. 34, no. 13, Jun. 2022, Art. no. e6909.

[80] H. Liu and R. Setiono, "Chi2: Feature selection and discretization of numeric attributes," in *Proc. 7th IEEE Int. Conf. Tools Artif. Intell.*, Nov. 1995, pp. 388–391.

[81] I. Jolliffe, "Principal component analysis," in *Encyclopedia of Statistics in Behavioral Science*. Wiley, 2005. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/0470013192.bsa501, doi: 10.1002/0470013192.bsa501.

[82] C. Oswald, S. E. Simon, and A. Bhattacharya, "SpotSpam: Intention analysis driven SMS spam detection using BERT embeddings," *ACM Trans. Web*, May 2022, doi: 10.1145/3538491.

[83] B. Pang, E. Nijkamp, and Y. N. Wu, "Deep learning with tensorflow: A review," *J. Educ. Behav. Stat.*, vol. 45, no. 2, pp. 227–248, 2020.

[84] N. Ketkar, "Introduction to Keras," in *Deep Learning With Python: A Hands-On Introduction*. Berkeley, CA, USA: Apress, 2017, pp. 97–111, doi: 10.1007/978-1-4842-2766-4_7.

[85] T. A. Almeida, J. M. G. Hidalgo, and A. Yamakami, "Contributions to the study of SMS spam filtering: New collection and results," in *Proc. 11th ACM Symp. Document Eng.*, 2011, pp. 259–262.

[86] M. A. Abid, S. Ullah, M. A. Siddique, M. F. Mushtaq, W. Aljedaani, and F. Rustam, "Spam SMS filtering based on text features and supervised machine learning techniques," *Multimedia Tools Appl.*, pp. 1–19, May 2022, doi: 10.1007/s11042-022-12991-0.

[87] O. Abayomi-Alli, S. Misra, and A. Abayomi-Alli, "A deep learning method for automatic SMS spam classification: Performance of learning algorithms on indigenous dataset," *Concurrency Comput., Pract. Exp.*, vol. 34, no. 17, Aug. 2022, Art. no. e6989, doi: 10.1002/cpe.6989.

[88] T. Singh, T. A. Kumar, and P. G. Shambharkar, "Enhancing spam detection on SMS performance using several machine learning classification models," in *Proc. 6th Int. Conf. Trends Electron. Informat. (ICOEI)*, Apr. 2022, pp. 1472–1478.

ISMAIL ALTAHARWA received the B.Sc. degree in computer science and its applications from The Hashemite University, Jordan, in 2005, the M.Sc. degree in computer science (emphasizes in AI techniques especially computational intelligence and evolutionary computations) from Al-Balqa Applied University, Jordan, in 2008, and the Ph.D. degree in computer science and information engineering (emphasized in machine learning techniques and information security) from the National Taiwan University of Science and Technology, Taiwan, in 2014. He is currently an Associate Professor at the Department of Computer Information Systems, The University of Jordan, Aqaba Campus.

NAZEEH GHATASHEH received the B.Sc. degree in computer information systems from The University of Jordan, Amman, Jordan, in 2004, and the M.Sc. degree in e-business management and the Ph.D. degree in e-business from the University of Salento, Italy, in 2008 and 2011, respectively. He served as the Chairperson of the Information Technology Department and Computer Information Systems Department, a University Liaison Officer for Quality and Development, the Dean's Assistant for Quality and Development, a Faculty Board Member, and the Director of the Computer Center at The University of Jordan, Aqaba, Jordan. He is currently an Associate Professor at the Department of Information Technology, The University of Jordan. His research interests include e-business, business analytics, applied computational intelligence, and data mining. He was awarded merit-based scholarships to pursue his M.Sc. degree.
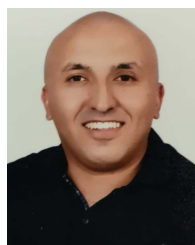
KHALED ALDEBEI received the B.S. degree in software engineering and the M.S. degree in computer science from Al-Balqa Applied University, in 2006 and 2009, respectively, and the Ph.D. degree in computer science from the University of Technology, Sydney, Australia, in 2018. His research interests include natural language processing (*NLP*), machine learning, authorship analysis, data mining, and text analysis.

• • •