Applied Soft Computing 22 (2014) 11–27

Contents lists available at ScienceDirect

Applied Soft Computing

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

Improved email spam detection model with negative selection algorithm and particle swarm optimization

Ismaila Idris*, Ali Selamat[1]

UTM-IRDA Digital Media COE, Office of Research Alliance & Faculty of Computing, Universiti Teknologi Malaysia, 81310 UTM Johor Bahru, Johor, Malaysia

ARTICLE INFO

Article history:
Received 13 May 2013
Received in revised form 17 February 2014
Accepted 2 May 2014
Available online 14 May 2014

Keywords:
Negative selection algorithm
Particle swarm optimization
Email
Spam
Non-spam
Detector generation

ABSTRACT

The adaptive nature of unsolicited email by the use of huge mailing tools prompts the need for spam detection. Implementation of different spam detection methods based on machine learning techniques was proposed to solve the problem of numerous email spam ravaging the system. Previous algorithm used in email spam detection compares each email message with spam and non-spam data before generating detectors while our proposed system inspired by the artificial immune system model with the adaptive nature of negative selection algorithm uses special features to generate detectors to cover the spam space. To cope with the trend of email spam, a novel model that improves the random generation of a detector in negative selection algorithm (NSA) with the use of stochastic distribution to model the data point using particle swarm optimization (PSO) was implemented. Local outlier factor is introduced as the fitness function to determine the local best (Pbest) of the candidate detector that gives the optimum solution. Distance measure is employed to enhance the distinctiveness between the non-spam and spam candidate detector. The detector generation process was terminated when the expected spam coverage is reached. The theoretical analysis and the experimental result show that the detection rate of NSA–PSO is higher than the standard negative selection algorithm. Accuracy for 2000 generated detectors with threshold value of 0.4 was compared. Negative selection algorithm is 68.86% and the proposed hybrid negative selection algorithm with particle swarm optimization is 91.22%.

1. Introduction

Email is now part of millions of people's life in the world today. It has changed the way man collaborates and works by being the most cheapest, popular and fastest means of communication [1]. Though, it recorded success in a lot of human activities, improving group communications, its impact was felt on the growth of business and also leads national development in a positive path. It is one of the technologies that has a direct impact on human life. The major short-coming of this technology is the increase in unsolicited email messages that a recipient receives. One significant and growing task that resulted from unsolicited email is the classification of email. This poses a problem among cooperate organizations and individuals trying to solve this menace called email spam. The task of email classification is shared into sub-tasks. The initial task is the collection of data and email message representation. Second task is the selection of a email feature and dimensional reduction of features [2], and the final task is the mapping of both training and testing set for classification of email. The essence of classification is to distinguish between spam and non-spam email. The problem of email spam is a global issue and is often encountered by all email users. It is defined as an unwanted junk email delivered to services on Internet mail. The amount of email spam has skyrocketed due to bulk mailing tools, this annoyed the receivers the more and the Internet service providers (ISP) are constantly under great pressure and complain on the problem of unsolicited email messages. Different techniques has been proposed in dealing with unsolicited email spam; the very first step in tackling spam is to detect spam email, this brought about the constant development of spam detection models; the models has two main approaches: the statistical and the non-statistical techniques. The statistical approach is a lot more effective than the non-statistical approach. Most of the statistical models in existence normally search for a specific keyword pattern in emails.

Quite a lot of machine learning techniques for email spam detection model have been proposed with little work on the negative

* Corresponding author. Tel.: +60 143813540.
E-mail addresses: ismi_idris@yahoo.co.uk (I. Idris), aselamat@utm.my (A. Selamat).
[1] Tel.: +60 7 5531008; fax: +60 7 5530160.

http://dx.doi.org/10.1016/j.asoc.2014.05.002
1568-4946/

selection algorithm. Research on the negative selection algorithm mainly focuses on anomaly detection, fault detection, malware detection and intrusion detection. Most work on negative selection algorithm (NSA) and particle swarm optimization (PSO) solves the problems of anomaly detection and intrusion detection. The implementation of particle swarm optimization with negative selection algorithm to maximize the coverage of the non-self space was proposed by Wang et al. [3] to solve the problem in anomaly detection. The research of Gao et al. [4] focuses on non-overlapping detectors with fixed sizes to achieve maximal coverage of non-self space; this is initiated after the generation of detectors by negative selection algorithm. There are few researches on the construction of email spam detection model with mammalian immune system functions; though, several immune system models are applied to virus detection [5], intrusion detection [6], anomaly detection [7] and malware detection [8]. If considerable effort is not made to find a technological solution to the menace of spam, the Internet email is in danger as an important medium of communication; in same way that the virus tried to disable the revolution of personal computers. The understanding of the artificial immune system based on the mammalian immune system is very vital in this study. The main goal of the immune system is to distinguish between non-self and self-element which is the basis for our implementation with negative selection algorithm, one amongst the algorithm of artificial immune system (AIS). This research will replace self in the mammalian immune system as non-spam in our system and non-self in the mammalian immune system as spam in our system. Details of negative selection algorithm (NSA) and its implementation will be discussed in Section 3. A battle against spam is a very difficult one; therefore, it makes for all a lot of sense to fight an adaptive pathogen with an adaptive system. This brought about the study of negative selection algorithm which is an adaptive algorithm in the fight against spam. The adaptive nature of the negative selection algorithm makes it able to supersede every other algorithm for email spam detection. The algorithm is able to learn from a previous attack, which is used to protect the system against the same attack in the future. Most models make emphasis on applying and designing computational algorithm and techniques with the use of simplified models of different immunological processes [9,10]. A review of machine learning approach for email spam classification was presented by Guzella et al. [11], the work discusses most of the techniques adopted in email spam classification like naïve Bayes (NB), support vector machine (SVM), artificial neural network (ANN), logistic regression (LR), lazy learning (LL), artificial immune system (AIS), boosting ensembles and other related approach. This paper proposes an improved solution for email spam detection inspired by the artificial immune system by the adoption of spam detection generation techniques with negative selection algorithm and particle swarm optimization. The particle swarm optimization (PSO) was implemented to generate detectors for training of negative selection algorithm to cover the spam space instead of the original random generation of detector use by negative selection algorithm. The paper is organized in to six sections. Section 1 is the introduction. Section 2 discusses the related work in negative selection algorithm. The proposed improved model and its constituent framework are discussed in Section 3. Empirical studies, results and discussion are in Section 4 and Section 5 discusses the experimental results. Model implementation and its advantages are presented in Section 6. Conclusion and recommendation is in Section 7.

2. Related work

Artificial immune system (AIS) is a new mechanism implemented for the control of email spam [12], it uses pattern matching in representing detectors as regular expression in the analysis of message. A weight is assigned to the detector which was decremented or incremented when observing the expression in the spam message with the classification of the message based on the threshold sum of the weight of matching detectors. The system is meant to be corrected by either increasing or decreasing all the matching detector weights with a 1000 detector generated from spam-assassin heuristic and personal corpus. The results were acceptable on the basis of the few number of detectors used. A comparison of the two techniques to determine message classification using spam-assassin corpus with 100 detectors was also proposed by [13]. This approach is like the previous techniques but the difference is the increment of weight where there is recognition of pattern in the spam messages. Random generation of the detector does not help in solving the problem of the best-selected features; though, feature weights are updated during and after the matching process of the generated detectors. The weighting of features complicates the performance of the matching process. In conclusion, the present techniques are better than the previous due to their classification accuracy and slightly improved false positive rate. More experimentation was performed by [14] with the use of spam-assassin corpus and Bayesian combination of the detector weight. Messages were scored by the simple sum of the message that was matched by each non-spam in the detector space and also by the use of Bayes scores. Words from the dictionary and patterns extracted from a set of messages are considered in detector generation besides the commonly used filters in order to be assured of the message classification. It was finally observed that the best results emerged when the heuristic was used with similar performance of the other two techniques. The approach of scoring features or feature weighting during and after the matching process does not help in the selection of important features for spam detection due to its computational cost.

Artificial immune system (AIS) collaborative filter that seems to learn the signature of a typical pattern of spam message with the aim of sampling words randomly from a message while removing words that exist in the non-spam message was proposed by [15]. This resulted in a robust system of obfuscation with respect to random words. Signatures that are to be distributed to other agents were selected with care in other to avoid the use of unreliable features. Spam-assassin corpus was used for the implementation of the experiment with promise of good result when there are few collaborated servers. Supervised real valued antibody network (SRABNET) that evolves detector population was also proposed by Bezerra et al. [16]. The sizes of the network are adjusted dynamically based on training data with the use of total cost ratio (TCR) as the training stopping criteria. The representation of messages are a bag of words (BoW) with features in binary, with a process of taking away words that appears below 5% in excess of 95% in all messages. The experiment adopts PU1 corpus with a 10 fold cross-validation. A genetic optimized spam detection using AIS algorithm was proposed by Mohammad and Zitar [17]. The genetic algorithm optimized AIS to cull old lymphocytes (replacing the old lymphocyte with new ones) and also to check for new interest for users in a way that is similar. In updating intervals such as the number of received messages, the interval is updated with respect to time, user request and so on; many choices were used in selecting the update intervals which was the aim of using the genetic algorithm. The experiment was implemented with spam-assassin corpus with 4147 non-spam messages and 1764 spam messages. The implementation of different pattern recognition scheme inspired by the biological immune system in order to identify uncommon situations like the email spam [17–20], unfortunately, has not been able to produce outstanding result.

It is quiet desirable to determine quantitatively the coverage of certain negative selection algorithm or make a conclusion on how detectors are distributed and their coverage in the spam space. The

work of D'Haeseleer [21] proposed a statistical technique as used in several other researches. The work analyses the probability of detecting random anomaly and also a survey on the relationship between the quantities of detectors was also analyzed while evaluating the number of detectors with a matching probability. The research of [22,23] work on the negative selection algorithm with similar analysis to support their theoretical findings while Wierzchon [24] uses statistic but emphasizes on different aspects such as lower bound for fault probability. In the case of finite binary or finite alphabet string representation, NSA's main focus is on different representation of data which becomes an extension for NSA, making it difficult to compute probability by a straightforward combination. An application that cannot be represented in its binary form makes use of real valued representation which is also vital for applications. The search space for real valued representation is continuous and hard to analyze, that is why the coverage in this case is less explored. Detectors are represented as hyper-spheres or hyperellipsoids in a real valued negative selection algorithm. These are generated through original generation elimination techniques or a host of other techniques. Fabio et al. [25] tried to minimize overlapping by implementing a random procedure to generate and redistribute detectors. Generation of detectors with genetic algorithm was implemented by Dasgupta and Gonzalez [26], the detectors are represented in a multi-dimensional real space or as per rules.

For the binary matching rules commonly used in negative selection algorithm (NSA), Balthrop et al. [27] first proposed the r-chunk matching rules which is an improvement over the r-contiguous matching rule originally proposed by Forest et al. [28]. The performance of the matching is evaluated by the number of detectors generated by the r-chunk matching rule over the binary representation. An evaluation of NSA with r-chunk and r-contiguous detector generation was also presented by Textor [29], this was achieved by making comparison against other methods based on kernels, finite state automata and n-gram frequencies. The evaluation confirms that NSA performed competitively by yielding an average better performance. The research of Chen and Yang [30] implement the generation of detectors in three main processes after listing and analysing the binary matching rules. The process includes gene library, clone selection and negative selection. Several new techniques are adopted to improve the performance of negative selection algorithm while constructing a co-evolutionary detector generation model. An improved negative selection algorithm by introducing a novel training is also proposed by Gong et al. [31]. The technique was implemented in the training phase to generate self-detectors to cover the self-region.

Major work implemented in the combination of two different algorithms in email spam was a hybrid model of AIS based on module, whose extracted features was designed by Sirisanyalak and Sornil [32] and further used for logistic regression model. A set of detectors was initially generated with the use of terms that are extracted from the training message, and also data on matched detector use in regression model. Spam-assassin was used for the experimental work. Rough set theory which is a mathematical approach for approximate reasoning in other to group messages in three class was proposed by Wenqing and Zili [33], targeting low false positive. The selection of feature, spam, non-spam or suspicious, was first implemented on the training set after which genetic algorithm was implemented. The universe of message is divided into three regions based on some induced set of rules. The experiment used only 11 features of the UCI corpus. It was concluded that the techniques are very efficient in reducing the number of non-spam message that are blocked and superior to naïve Bayes classifier. A data compression model operating at raw message level was proposed by Bratko et al. [34] making consideration for spam and non-spam class as a source of information. The message that was used for the training of the representative of each of the spam and non-spam class is made from a sample of generated data by each source. The compression model of each of the source is then built and used in the analysis of the incoming message. Evaluation was implemented using Ling-spam, PU1, PU3 dataset and 10-fold cross validation with online settings while comparing with other techniques. It was observed that the algorithm attains a high classification rate and was also superior to other techniques. A hidden Markov model was applied to the problem of finding observed words by Gordillo and Conde [35], with the assumption of a black list that is made up of words that are related to the spam messages. By a given list of variance for each model, a model is trained for each of the forbidden words. A word extracted from the message can be an input to each of the model, making sure it is a variant of the corresponding words. The techniques were discovered to be able to identify about 95% of the variant of words without result of classification. A combination of a support vector machine (SVM) and a artificial immune system (AIS) was proposed by Guangchen and Ying [36], with the use of binary features with same feature selection in [16]. The support vector acquired after training SVM is implemented in the generation of a initial detector set of the AIS and then the AIS was used in classification. During classification with AIS, detector with the smallest Euclidean distance to the message is added to the committee set with the major voting of detector in the set as the classification. PU1 and Ling-spam corpora are used for the experiment in an online fashion.

3. The proposed improved model and its constituent frameworks

Improved systems in recent times have extensive success in many real world complex problem solving. The importance of a combined system is not negotiable, based on the fact that a individual system has its weakness, and a improved system is meant to compliment the weakness of these individual intelligent systems. A smart combination of negative selection algorithm and particle swarm optimization is investigated in order to compliment the parameters of each component of the system by using the advantages of an individual system against its disadvantages while elevating each weak component member of both systems to achieve stability, consistency and an accurate intelligent system extendable for usage in classification. The proposed improved system is composed of negative selection algorithm and particle swarm optimization combined uniquely to form a better-improved system with local outlier factor (LOF) as fitness function.

3.1. The original negative selection algorithm (NSA)

Negative selection algorithm (NSA) has been used successfully for a broad range of applications in the construction of the artificial immune system [27]. The standard algorithm was proposed by Forest et al. [28]. The algorithm comprises of the data representation phase, the training phase and the testing phase. In the data representation phase, data are represented in a binary or in a real valued representation. The training phase of the algorithm or the detector generation phase randomly generate detector with binary or real valued data and then they are consequently used to train the algorithm [37], while the testing phase evaluates the trained algorithm. The random generation of detectors by a negative selection algorithm makes it impossible to analyze the type of data needed for the training algorithm. Figs. 1 and 2 show the training and testing phase of NSA.

The main concept of the NSA as developed by Forrest et al. [28] was meant to generate a set of candidate detectors, C, such that $\forall x_i \in C$ and $\forall z_p \in S$, $f_{\mathrm{MATCH}}(x_i, z_p) < r$ where x_i is a detector, z_p

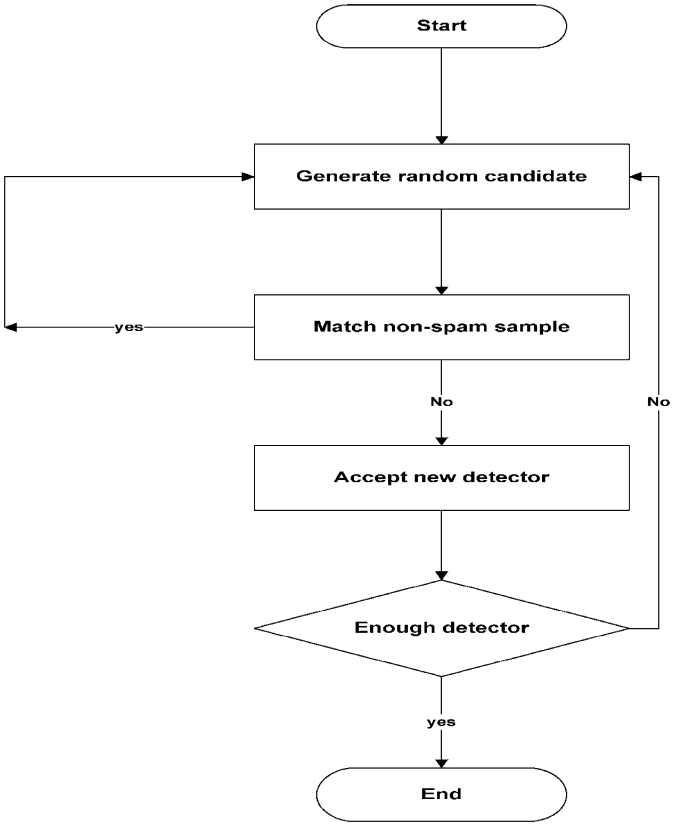

Fig. 1. Detector generation of negative selection algorithm.

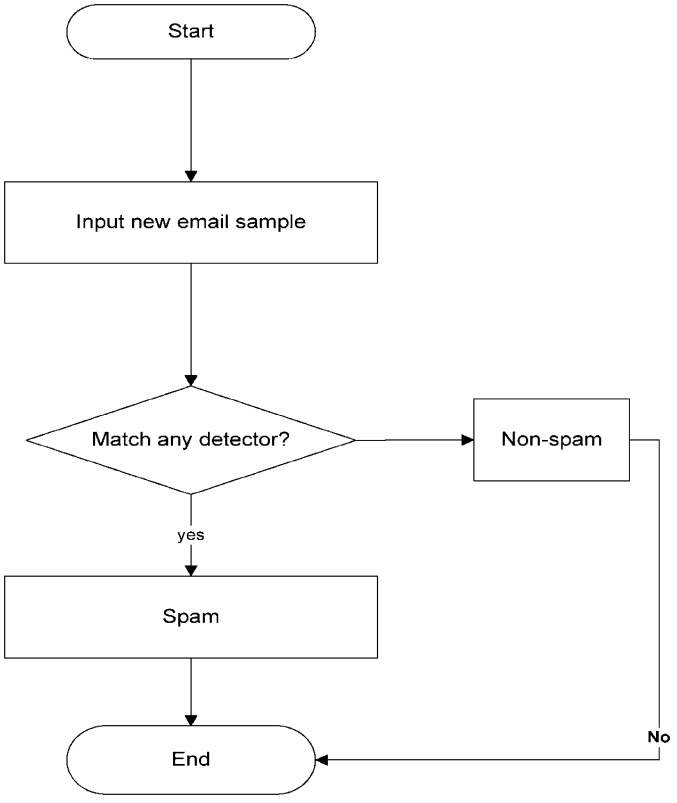

Fig. 2. Testing of negative selection algorithm.

Let counter, n_c , be the number of self detectors to train;
Let C be an empty set of self detectors;
Let r be the affinity threshold;
Create a training set, D_{TRAIN}, made up of self patterns;
While $/\ C\ / \neq\ n_c$ **do**
  Randomly generate a detector, x_i ;
  Matched: = false:
  For each self pattern, $z_p \in D_{TRAIN}$ **do**
    If $f_{MATCH}(x_i, z_p) < r$ **then**
      Matched:= true;
      Break;
    end
  end
  If matched = false **then**
    Add x_i *to* C;
end
end

Fig. 3. Original negative selection algorithm.

is a pattern and $f_{\mathrm{MATCH}}(x_i,z_p)$ is the affinity matching function. The algorithm of NSA as given by Forrest et al. [28] is presented in Fig. 3.

The original NSA uses binary *r*-contiguous bits (RCBITS) rules in conjunction with a global affinity threshold, *r* for each detector in population of detectors, *C*. The determination of the affinity threshold is by try and error, because the threshold value that gives the best system performance is selected as the target affinity threshold. AIS researchers have shown that the affinity matching distance is important and has an impact on NSA performance [27,38].

3.1.1. Implementation of negative selection algorithm

The proposed spam base dataset for the research is in real valued representation. The real value negative selection algorithm is encoded in real value for classifying non-spam and spam. In the case of real value, there is need to define the non-spam and the spam space. The non-spam space is the normal state of a system while the spam space is the abnormal state of a system. The candidate detectors are randomly generated and then compared to the non-spam samples. Candidate detectors that do not match any sample of the non-spam set are accepted as viable detectors. Candidate detectors that match the sample of the non-spam set are discarded as unwanted detectors. The generation of detector continues until the detector set attains the required coverage of the spam space. After the generation of detectors in the spam space, the generated detectors can then monitor the status of the system. If some other new (test) samples match at least one of the detectors in the system, it is assumed to be spam which is abnormal to the system; but if the new (test) sample does not match any of the generated detectors in the spam space, it is assumed to be non-spam. The non-spam sample in a real value negative selection algorithm is represented in N-dimensional points and a non-spam radius Rs, as training dataset. In clearer terms, let Eq. (1) represent the non-spam space.

$$S = \{X_i | i = 1, 2, \ldots m; \quad Rs = r\} \tag{1}$$

X_i is some point in the normalized N-dimensional space.

$$X_i = \{x_{i1}, x_{i2}, x_{i3}, \ldots, x_{iN}\}, \quad i = 1, 2, 3, \ldots, m \tag{2}$$

The entire normalized sample is $space^{I \subset [0,1]^N}$, the spam space can then be represented as $S = 1 - NS$ where S is spam and NS is non-spam.

$$d_j = (C_j, R_j^d) \tag{3}$$

Eq. (3) denotes one detector where $C_j = \{C_{j1}, C_{j2}, C_{j3}, \ldots, C_{jN}\}$ is the detector centre and R_j is the detector radius. The Euclidean

distance is used as the matching measurement. The distance between non-spam sample X_i and the detector d_j can be defined as:

$$L(X_i, d_j) = \sqrt{(x_{i1} - C_{j1})^2 + \cdots + (x_{iN} - C_{jN})^2} \quad (4)$$

$L(X_i,d_j)$ is compared with the non-spam space threshold Rs, obtaining the match value of $\propto$.

$$\propto = L(X_i, d_j) - Rs \quad (5)$$

The detector d_j fails to match the non-spam sample X_i if $\propto > 0$; therefore if d_j does not match any non-spam sample, it will be retained in the detector set. The detector threshold R_j^d of detector d_j can be defined as:

$$R_j^d = \min(\propto), \ if \ \propto \leq 0 \quad (6)$$

Also, if detector d_j matches the non-spam sample, the detector will be eliminated. The generation of detectors continues until the number of detectors needed to cover the spam space is attained. After the generation of detectors in the spam space, the detectors are then used to monitor the system status. If the testing dataset matches any detector in the spam space, it is labelled as spam but if the testing dataset set does not match any detector in the spam space, it is labelled as non-spam.

3.2. The proposed improved negative selection algorithm model

The detector generation as shown in real valued negative selection algorithm in Section 3.1.1 is vital in enhancing the performance of the negative selection algorithm. Random generation of detector by the real value negative selection algorithm was improved with the introduction of particle swarm optimization (PSO) with local outlier factor (LOF) as fitness function. These are as a result of the quest for efficiently trained negative selection algorithm model for purely normal detectors. The fitness function calculates the distance between the candidate detector and the non-spam space to generate the reach-ability distance. The best reach-ability distance of each candidate detector was then used to calculate the distance between the individual candidate detectors to generate the best detector known as the local outlier factor (LOF). The approach will model the data point by the implementation of stochastic distribution [39] using local outlier factor. The proposed technique is able to improve the traditional random generation of detectors in real value negative selection algorithm and optimize the generated detectors in spam space at the same time. The following sections explain the processes in its implementation.

3.2.1. Definition of spam and non-spam space

In the case of real value negative selection algorithm, there is a need to define the non-spam and the spam space. The non-spam space is the normal state of a system while the spam space is the abnormal state of a system. This is important so that the system will have a non-spam space that will be used to learn the generated detectors on the algorithm in the spam space.

Let us assume the non-spam space to be S

S is defined as follows:

$$s = (s_1 \ldots s_n) = \begin{bmatrix} s_{11} & \cdots & s_{1m} \\ \vdots & \ddots & \vdots \\ s_{n1} & \cdots & s_{nm} \end{bmatrix} \quad (7)$$

$$S_{ij} \in K^m, \quad i = 1, \cdots, n; \quad j = 1, \ldots, m$$

S is normalized as follows:

$$S_i = \frac{s_i}{||s_i||} \quad (8)$$

Therefore, s_i is the ith-non-spam unit; and s_{ij} is the jth vector of the ith non-spam unit.

We obtain the non-spam data from the training dataset that is used to train the system, the same number of the non-spam set is used as the normal state of the system while the other part of the training set which is the spam and the non-spam set of the dataset is used to train this system.

3.2.2. Generate random candidate detector

The random generation of a candidate detector helps in the selection of data in negative selection algorithm, required in the process of generating detectors. In this scenario we generate random candidate detector as follows.

$$r = (r_1 \ldots r_n) = \begin{bmatrix} r_{11} & \cdots & r_{1m} \\ \vdots & \ddots & \vdots \\ r_{k1} & \cdots & r_{nm} \end{bmatrix} \quad (9)$$

$$r_{ij} \in (0, 1)^m, \quad i = 1, \cdots, k; \quad j = 1, \cdots, m$$

r_i is said to be the ith detector and rij is the jth feature of the ith detector.

3.2.3. Generation of candidate detector with particle swarm optimization (PSO)

Detector generation was implemented with particle swarm optimization instead of the traditional random generation of detectors. The modification of particle swarm optimization (PSO) was implemented in this research by eliminating the global best (Gbest) since the proposed system requires one optimum solution to cover the spam region. Therefore, the local best (Pbest) solution becomes the optimum solution for the system. The framework of the hybrid system is presented in Fig. 4. It shows the detector generation phase of the real valued negative selection algorithm with the use of Particle swarm optimization in the generation of detectors.

3.2.3.1. Detector generation parameters and implementation. Particle is made up of 57 features $\{f_{57}\}$ while the accelerated constant C value is 0.5.

The position and velocity of particle swarm optimization are represented in the N-dimensional space as:

$$P_i(p_i^1, p_i^2, \ldots, p_i^n) \quad (10)$$

$$V_i(v_i^1, v_i^2, \ldots, v_i^n) \quad (11)$$

where p_{id} is the binary bits, $i = 1, 2, \ldots, m$ (m is set to be the total number of particles), $d = 1, 2, \ldots, n$ (n is the dimensionality of the data). Each particle in the generation updates its own position and velocity based on Eqs. (10) and (11).

The initialization of the real value particle swarm optimization is established by the population of particles (non-spam and spam). All particles move in the problem space in other to find the optimal solution in individual iterations. Given n-dimensional space, the particles exhibit a potential solution while each particle possesses direction and position vector for its movement and direction.

To determine the best particles we use local outlier factor (LOF) as fitness function for our system. The proposed particle swarm optimization requires one best optimum solution, and each generated candidate detector (Pbest) is used as the optimum solution in the spam space; the global best (Gbest) solution is eliminated since our solution need not jump from one optimum solution to another and all candidate detectors are potential optimum solutions in the

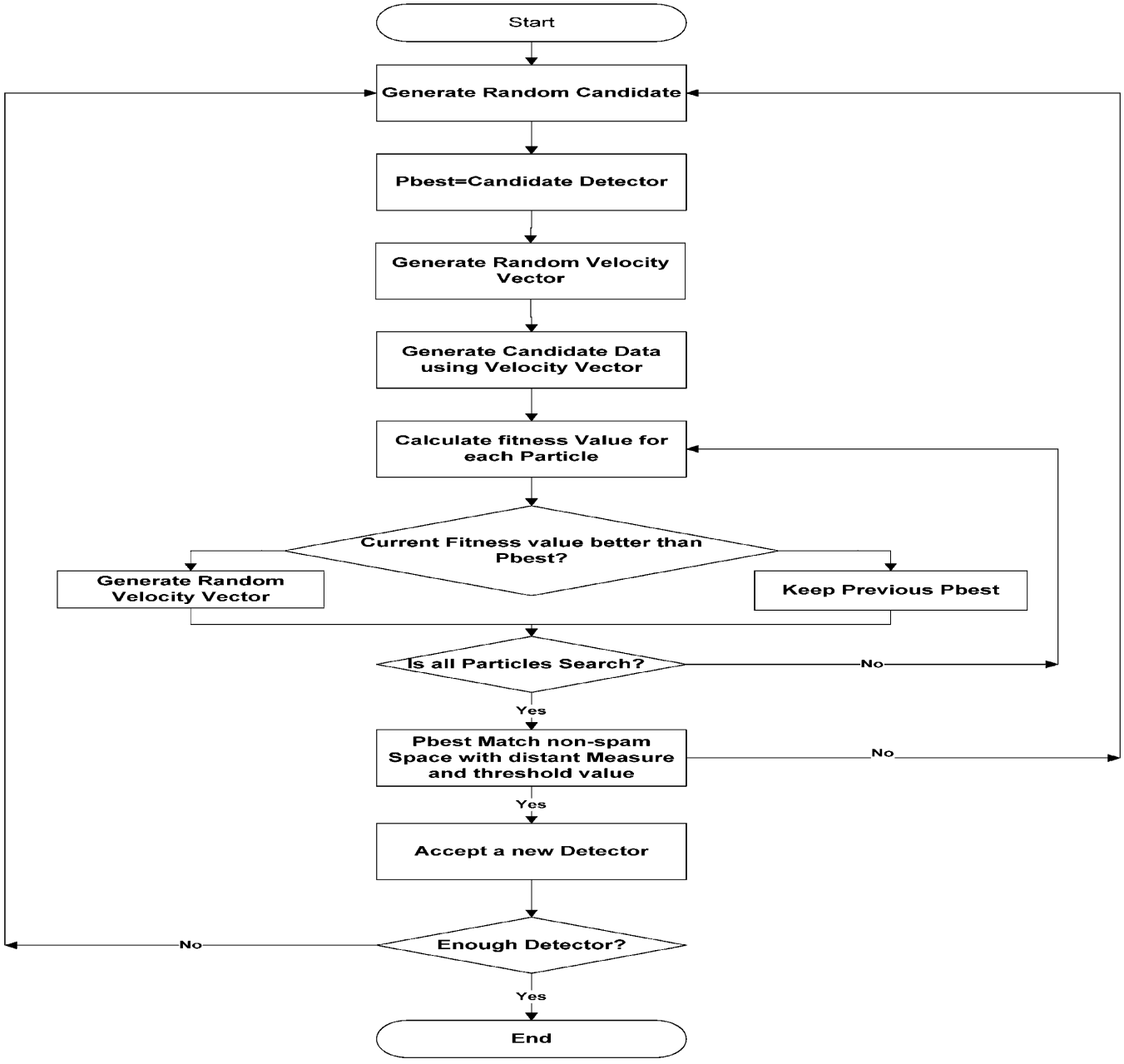

Fig. 4. Framework of proposed improved NSA–PSO detector generation model.

swarm and each local best (pbest) particle is the optimum solution reached after comparing all particles in the swarm. We do not have a unique optimal solution in our problem that will require using Gbest to determine it. The global best (Gbest) solution takes us to cover another space instead of covering the immediate position or space. Therefore, the movement that was attained using Gbest is too long compared with the movement that is required to cover the spam space in the cause of detector generation.

In generating a random initial velocity matrix for random candidate detectors we define $v(0)$.

$$v(0) = \begin{bmatrix} v_{11}(0) & \cdots & v_{1m}(0) \\ \vdots & \ddots & \vdots \\ v_{j1}(0) & \cdots & v_{jm}(0) \end{bmatrix} \quad (12)$$

Eqs. (13) and (14) updates the new velocity and particle position:

$$v_{id}(t+1) = v_{id}(t) + c(Pbest_{id}(t) - x_{id}(t)) \quad (13)$$

$$x_{id}(t=1) = x_{id}(t) + v_{id}(t+1) \quad (14)$$

where $i = 1, 2, 3, \ldots, n$, n is said to be the number of particles in the swarm. For instance $v_i(t)$ and $x_i(t)$ are the velocity and position of particle i respectively. The value r^1 and r^2 are the random variables which are generated in each update process in velocity. c is the user-supplied coefficients parameter. The velocity formula is made up of three terms. Each term has the specific role in the PSO algorithm. The first term $V_i(t)$ is known as the 'inertia' component. It is responsible for keeping the particle moving in the same direction. The second and third terms are known as the 'cognitive' and 'social' component. According to experience, cognitive component responsibility is to return the research space with high individual fitness. And, the social component is responsible for the particle

```
//pb  is the local best
//gb  is the global best
//LOF  is the local outlier factor
//p  is the population size
[1] Initialize all particles
[2] Initialize
[3] Repeat
[4] For each particle i in p do
[5] Compute the fitness function as shown in  equation (11) and (12) with LOF
[6] If fitness value better than best fitness x_i
[7] //Update each particle best position
[8]      If f(x_i) < f(pb_i) then
[9]     pb_i = x_i
[10] End if
[11] //Since we need pbest as the optimal  solution
[12] //Update local best position (pb_i)
[13]        If f(pb_i) = f(gb) then
[14]      pb_i = gb
[15]    End if
[16] End for
[17] //Update particle velocity and position
for each particle i in p do
[18] For each dimensional d in D do
[19] v_{i,d} = v_{i,d} + C_1 * Rnd(0,1) * [pb_{i,d} − x_{i,d}] + C_2 * Rnd(0,1) * [pb_d − x_{i,d}]
[20] x_{i,d} = x_{i,d} + v_{i,d}
[21]   End for
[22] End for
[23] While maximum iteration or stop criteria  reached.
```

Fig. 5. Particle swarm optimization algorithm.

moving to the best area in the search space. The velocity of a particle is limited to $[-V_{max}, V_{max}]$; the purpose of setting V_{max} is to provide a mechanism to avoid the excessive.

The process of the methodology can be explained in the following steps:

Step 1: Define a stable behaviour and activities of a system as non-spam space (normal pattern) as shown in Eq. (8).
Step 2: From the population of spam and non-spam data, generate training and testing profile with random candidate detector as shown in Eq. (9).
Step 3: Eqs. (10) and (11) initializes both position and velocity of particle swarm optimization.
Step 4: Calculate both reach-ability distance and the local outlier factor for each candidate detector as shown in Eqs. (26) and (27).
Step 5: Update each candidate detector position and velocity with Eqs. (13) and (14).
Step 6: Implement the distance measure in Eq. (4) and threshold value in Eq. (5) to determine the pbest similarity in the non-spam space region S. If pbest did not match S, it is a valid detector.
Step 7: Continuous generation and matching of pbest against S is observed for changes, deviation of the system may occur if pbest matches S. Pbest is not meant to match S.
Step 8: After maximum coverage in the spam space, the testing set is employed for evaluation (Fig. 5).

3.2.3.2. Model computation. We consider the movement of I particles in the space $\mathbb{R}^3$. The ith particle position is presented as:

$$x_i(t) = x_i(t)(i = 1, \ldots, I) \tag{15}$$

And the velocity is presented as:

$$V_i(t) = V_i(t)(i = 1, \ldots, I) \tag{16}$$

Therefore,

$$dx_i = V_i dt + \delta_i dB_i \tag{17}$$

$$dv_i = \left\{ -\propto \sum\nolimits_{j \neq i} \left[\left[\frac{r}{x_{ij}} \right]^p - \left[\frac{r}{x_{ij}} \right]^q \right] x_{ij} \right.$$
$$\left. -\beta \sum\nolimits_{j \neq i} \left[\left[\frac{r}{x_{ij}} \right]^p - \left[\frac{r}{x_{ij}} \right]^q \right] V_{ij} + f_i(t, x_i, v_i) \right\} dt \tag{18}$$

where $X_{ij} = x_i - x_j$ and $V_{ij} = x_i - v_j$

Eq. (17) is a stochastic equation for x_i while Eq. (18) is the deterministic equation for V_i. The term $\delta_i dB_i$ is the overlapping that exists between two particles during detector coverage of the ith particle where δ_i is the coefficient.

$\{B_i(t), t \geq 0\}$ $(i = 1, \ldots, I)$ defined an independent movement of particles on a probability space with filtration $(\Omega, F, \{F_t\}_{t \geq 0}, P)$.

Therefore, a function of the form in Eq. (19) is taken by,

$$f_i = -cv_i - \gamma \nabla f(x_i) \tag{19}$$

where, $-cv_i$ represents a resistance for motion while $-\gamma \nabla f(x_i)$ represents an external force that is determined by a potential function $f : \mathbb{R}^3 \to \mathbb{R}$.

We assume $f(x)$ to be a real valued function defined by $x \in \mathbb{R}^D$ with positive integer D. The optimization problem is considered in Eq. (20):

$$\min_{x \in \mathbb{R}^D} f(x_i) \tag{20}$$

Eq. (20) presents an optimization problem with respect to Eqs. (17) and (18).

Since Eqs. (15) and (16) represent the position of I particles moving about the space $\mathbb{R}^D$ and their velocity respectively, we set $\beta = 0$ (ignore the overlapping of particles) and then take the function $f_i(t,x_i,v_i)$ in (18) in the form (19). This is then represented as:

$$x_i(t + \Delta t) = x_i(t) + v_i(t + \Delta t)\Delta t + \delta\xi_i(\Delta t) \tag{21}$$

$$v_i(t + \Delta t) - w(\Delta t)v_i(t) - \left\{ - \propto \sum\nolimits_{j \neq i} \left[\left[\frac{r}{x_{ij}} \right]^p - \left[\frac{r}{x_{ij}} \right]^q \right] x_{ij} - \gamma \Delta f[x_i(t)] \right\} \Delta t \tag{22}$$

where $x_{ij}(t) = x_i(t) - x_j(t)$.

$\xi_i(\Delta t) = B_i(t + \Delta t) - B_i(t)$, $i = 1, \ldots, I$ is a family of independent random functions that is defined on a probability space $(\Omega, F, \{F_t\}_{t \geq 0}, P)$ with values in $\mathbb{R}^D$ and the distributions are a normal distribution with mean 0 and variance Δt and $w(\Delta t)$ is given by $w(\Delta t) = 1 - c\Delta t$.

On setting the initial position and velocity we then have:

$$x_0 \equiv (x_1(0), \ldots, X_I(0)) \in \mathbb{R}^{DI} \tag{23}$$

$$v_0 \equiv (v_1(0), \ldots, v_I(0)) = 0 \in \mathbb{R}^{DI} \tag{24}$$

The algorithms (7) and (8) defines a discrete trajectory,

$$x_n \equiv (x_1(t_n), \ldots, X_I(t_n)) \in \mathbb{R}^{DI} \tag{25}$$

where $n = 0, 1, 2, \ldots, N$, $t_n = n\Delta t$.

In each n step, the minimal value is computed $f_n \equiv \min_{1 \leq i \leq I} f[x_i(t_n)]$ while memorizing its value together with the point $x_n \in \mathbb{R}^D$ attained. i.e. $f_i \equiv \min_{0 \leq n \leq N} f_n$ is an approximate value of Eq. (20) and $\bar{x}$ while $f(\bar{x}) = \bar{f}$ is an approximate solution of the scheme.

Since $\bar{x}$ is a member of swarm $x_1, \ldots, x_I$ having interaction on one another, the approximate solution $\bar{x}$ may not be satisfactory to the condition $\nabla f(\bar{x}) = 0$. This means that there is a point $\bar{x}$ in the neighbourhood of $\bar{x}$ that attains a local minimum of $f(x)$, i.e., $\nabla f(\bar{x}) = 0$. In this scenario, $\bar{x}$ which can easily be obtained by classical methods produces a good approximation solution of (20) than $\bar{x}$.

3.2.3.3. Fitness function. The local outlier factor (LOF) was employed to calculate the fitness function in quest for a purely normal data that will efficiently train our model. An outlier can be defined as a data point that is not the same as the remaining data with respect to some measure. It is employed as a fitness function for the generation of unique features in the spam space. The technique will model the data point with the use of a stochastic distribution [39] and the point is determined to be an outlier base on its relationship with the model. The outlier detection algorithm proposed as fitness function in this study of spam detection generation is very unique in computing the full dimensional distance from one point to another [40,41] while computing the density of the local neighbourhood.

- Let us assume k distance (i) to be the distance of the candidate detector or particle (i) to the nearest neighbourhood (non-spam).
- Set of k nearest neighbour (non-spam element) includes all particles at this distance.
- Set S of k nearest neighbour is denoted as $N_k(i)$.
- This distance defines the reach-ability distance.
- Reach-ability-distance$_k(i,s) = \max\{k - distance(s), d(i,s)\}$.

The local reach-ability distance is then defined as:

$$\text{lrd}(i) = 1/\left(\frac{\Sigma_{s \in N_k}(i) reachability - distance_k(i,s)}{|N_{K(i)}|} \right) \tag{26}$$

Eq. (26) is the quotient of the average reach-ability distance of the candidate detector i from the non-spam element. It is not the average reachability of the neighbour from i but the distance from which it can be reached from its neighbour. We then compare the local reach-ability density with those of its neighbour using Eq. (27):

$$\text{LOF}_K(i) = \frac{\sum s \in N_k(i) \frac{lrd(s)}{lrd(i)}}{|N_k(i)|} = \frac{\sum s \in N_k(i) lrd(s)}{|N_K(i)|} / lrd(i) \tag{27}$$

Eq. (27) shows the average local reach-ability density of the neighbour divided by the particle's own local reach-ability density. In this scenario, values of the particle as approximately 1 indicates that the particle is comparable to its neighbour (not an outlier), value below 1 indicates a dense region (which will be an inlier) while a value larger than 1 indicates an outlier. The major idea of this technique is to assign to each particle the degree of being an outlier. The degree is called the local outlier factor (LOF) of the particle. The methodology for the computation of LOF's for all particles is explained in the following steps:

Step 1: For each particle i compute k distance element in non-spam space (distance of k – the nearest neighbour is non-spam space s) as shown in Eq. (28)
Step 2: Eq. (29) computes reach-ability distance for particle i with non-spam space as: Reach-dist.$_k(i) = \max\{k - distance(s), d(i,s)\}$, when $d(i,s)$is the distance from particle i to non-spam spaces.
Step 3: computation of the local reach-ability density of particle i as inverse of the average reachability distance based on *Minpts* (minimum number of non-spam space) nearest neighbour of particle i in Eq. (30).
Step 4: Eq. (31) computes LOF of particle i as average of the ratios of the local reach-ability density of the neighbours in non-spam space divided by the number of the objects own local reach-ability density.

Let us assume G as the population of particles, S is the non-spam space and i is the ith particle in G,

$$\text{For each particle } i \text{ we have } i \in G. \quad \text{Max}\,(k - dist.(s) \tag{28}$$

$$\left|\text{Reach-dist} \quad G\right|^* \max(dist(s,i)) \tag{29}$$

$$|G|^*(Minpt(s,i)) \tag{30}$$

$$|G|^*(similarity\,(i,G)) \tag{31}$$

The algorithm is represented in Fig. 6.

3.2.3.4. Computation of fitness function. The proposed computation uses $direct(x)$ to denote the mean value of $direct_{min}(x)$ and $direct_{max}(x)$. Also, $indirect(x)$ is used to denote the mean value of $direct_{min}(x)$ and $direct_{max}(x)$.

Algorithm LOF

Input: G //Random particle population
S //Non-spam space
i // i^{th} particle in G
Output: The degree of local outlier factor for all record of i particle.

1. **Begin**
2. Population of random dataset G
3. Reach-dist: k = $dist\ (G, i)$
4. **For each** i *in* G ***do begin***
5. Reach-dist $i = max\ (dist(s, i))$
6. $|G|^*\ (Minpt\ (s, G))$
7. $Max - dist\ (i) \in (G)$
8. Find population G with max reachability distance with s
9. $Max\ (dist(i, s))$
10. Find population G with maximum similarity with i
11. **end**
12. **Return** $|G_{max}|^*$ similarity (i, G_{max})
13. **end**

Fig. 6. Algorithm for fitness function.

For any particle, let $direct_{min}(x)$ denote the minimum reach-ability distance that is between x and a *MinPts*-nearest neighbour of x.

$$direct_{min}(x) = Min\left\{reach - \frac{distance(x, y)}{y} \in N_{MinPts}(x)\right\}. \tag{32}$$

Also, let $direct_{max}(x)$ denote the corresponding minimum;

$$direct_{max}(x) = Max\{reach - distance(x, y)/y \in N_{MinPts}(x)\} \tag{33}$$

In order to further generalize the definitions of the $MinPts - nearest$ neighbour y of x, let $indirect_{min}(x)$ denote the minimum reach-ability distance between y and a $MinPts - nearest$ neighbour of y.

$$indirect_{min}(x) = Min\{reach - dist(y, z)\backslash y \in N_{MinPts}(x) \text{ and } z \in N_{MinPts}(y)\} \tag{34}$$

Let, $indirect_{max}(x)$ denote the corresponding maximum, therefore, $x's$ $MinPts - nearest$ neighbour is referred to as $x's$ indirect neighbour, wherever y is a $MinPts - nearest$ neighbour of x.

Theorem. Lets assume x as an object from the database D, and $1 \le MinPts \le |D|$, we then have

$$\frac{direct_{min}(x)}{indirect(x)} \le LOF(x) \le \frac{direct_{max}(x)}{indirect_{min}(x)} \tag{35}$$

Proof:

$$\frac{direct_{\min}(x)}{indirect_{\max}(x)} \le LOF(x) \tag{36}$$

$\forall z \in N_{\min Pts}(x) : reach - dish(x, \quad z) \ge direct_{\min}(x)$. By the of $direct_{min}(x)$

$$1/\frac{z \in N\sum_{MinPts(x)} reach - dist(x, z)}{|N_{\min Pts}(x)|} \le \frac{1}{direct_{\min}(x)} \tag{37}$$

$$lrd(x) \le \frac{1}{direct_{\min}(x)} \tag{38}$$

$\forall y \in N_{\min Pts}(z) : reach - dish(z, \quad y) \ge direct_{\min}(x)$. By the of $indirect_{max}(x)$

$$1/\frac{y \in N\sum MinPts(z) reach - dish(z, y)}{|N_{minPts}(z)|} \ge \frac{1}{indirect_{max}(x)} \tag{39}$$

$$lrd(z) \ge \frac{1}{indirect_{max}(x)} \tag{40}$$

We then have:

$$LOF(x) = \frac{z \in N\sum_{MinPts(x)} \frac{lrd(z)}{lrd(x)}}{|N_{MinPTs}(x)|} \ge \frac{z \in N\sum_{MinPts(x)} \frac{\frac{1}{indirect_{max}(x)}}{\frac{1}{direct_{min}(x)}}}{|N_{MinPTs}(x)|} = \frac{direct_{min}(x)}{indirect_{max}(x)} \tag{41}$$

$$LOF(x) \le \frac{direct_{min}(x)}{indirect_{max}(x)} \quad \text{Proved} \tag{42}$$

3.2.4. Implementation model

The N-dimensional points and a non-spam radius Rs represent the training dataset.Let Eq. (21) represent the non-spam space.

$$S = X_i | i = 1, 2, \ldots, m; \quad Rs = r \tag{43}$$

X_i are some points in the normalized N-dimensional space.

$$X_i = \{x_{i1}, x_{i2}, x_{i3}, \ldots, x_{iN}\}, \quad i = 1, 2, 3, \ldots, m \tag{44}$$

Each of the particles were initialized at a random position in the search space. The position of particle i is given by the vector:

$$x_i = x_{i1}, x_{i2}, \ldots, x_{iD} \tag{45}$$

where D is the problem dimensionality with velocity given by the vector:

$$v_i = v_{i1}, v_{i2}, \ldots, v_{iD} \tag{46}$$

The movement of the particles was influenced by an implemented memory, in the cognitive memory:

$$p_i = (p_{i1}, p_{i2}, \ldots, p_{iD}) \tag{47}$$

The best previous position visited by each individual particle i is stored.

$$p_{best} = (p_{best1}, p_{best2}, \ldots, p_{bestD}) \tag{48}$$

The vector in Eq. (48) contains the position of the best point in the search space visited by all the particles.

At each iteration, each pbest is used to compute the density of the local neighbourhood.

$$lrd(i) = 1/\frac{\sum s \in N_k(i) reachability - distance_k(i, s)}{|N_K(i)|} \tag{49}$$

After which the local reach-ability density is compared with those of its neighbour's reach-ability distance

$$\mathrm{LOF}_k(i) = \frac{\sum s \in N_k(i) \frac{lrd(s)}{lrd(i)}}{|N_k(i)|} = \frac{\sum s \in N_k(i) lrd(s)}{|N_k(i)|} / lrd(i) \tag{50}$$

Given each particle a degree of being an outlier each iteration of pbest velocity were updated according to Eq. (51).

$$v_i(t + 1) = w \cdot v_i(t) + n_1 r_1 (p_i - x_i(t)) + n_2 r_2 (p_{best} - x_i(t)) \tag{51}$$

where w is the local outlier factor for each particle of the velocity, n_1 and n_2 are positive constants called "cognitive" and "social" parameters that implements the local outlier factor of two different swarm memories and r_1 and r_2 are a random number between 0 and 1. The proposed procedures does not require the swarm to perform a more global search with large movement, it only requires a small movement and fine-tuning in the end of the optimization process. After calculating the velocity vector, the position of the particles is updated based on Eq. (52).

$$x_i(t + 1) = x_i(t) + v_i(t + 1) \tag{52}$$

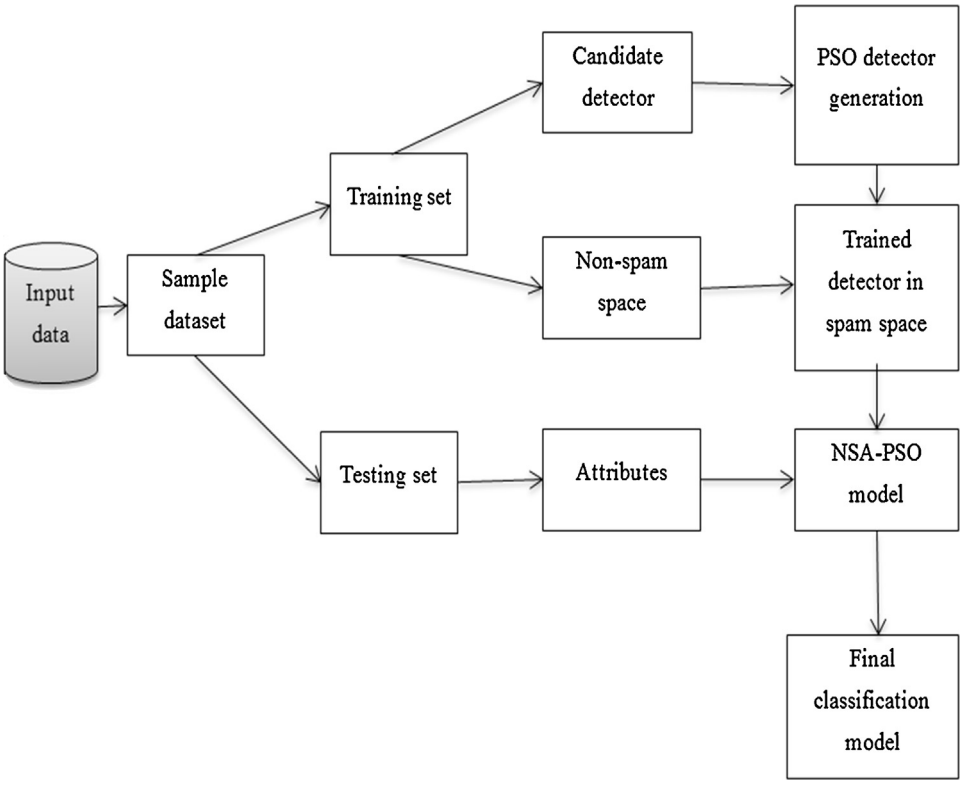

Fig. 7. Data flow diagram of proposed NSA–PSO improved model.

In the normalized samples $space^{I\subset[0,1]^N}$, the spam space is represented as $S = I - NS$ where S is spam and NS is non-spam.

$$d_j = (C_j, R_j^d) \tag{53}$$

We then employ a maximum number of iteration as termination condition for the algorithm based on the task.

Eq. (53) is a representation of one detector d_j with centre $C_j = \{C_{j1}, C_{j2}, C_{j3}, \ldots, C_{jN}\}$ as the detector centre with respect to numbers of detector d_j, while R_j is the detector radius of each detector d_j with respect to diameter R^d. The Euclidean distance is used as the matching measurement. The distance between non-spam sample X_i and the detector d_j can be defined as:

$$L(X_i, d_j) = \sqrt{(x_{i1} - C_{j1})^2 + \cdots + (x_{iN} - C_{jN})^2} \tag{54}$$

A comparison of $L(X_i,d_j)$ with the non-spam space threshold Rs result in the match value $\propto$ where,

$$\propto = L(X_i, d_j) - Rs \tag{55}$$

The detector d_j does not match the non-spam sample X_i if $\propto > 0$, therefore if d_j does not match any non-spam sample, it is accepted in the detector set. The detector threshold R^d, j of detector d_j is defined as:

$$R^d, j = \min(\propto), \quad \text{if } \propto \leq 0 \tag{56}$$

If detector d_j matches the non-spam sample, it will be discarded. This will not stop the generation of detector until the required detector set is reached and the required spam space coverage is attained. After the generation of detectors in the spam space, the generated detectors can then monitor the status of the system. If some other new email (test) samples matches at least one of the detectors in the system, it is assumed to be spam which is abnormal to the system but if the new email (test) sample does not match any of the generated detectors in the spam space, it is assumed to be a non-spam email.

3.2.5. Flow of proposed model.

The data flow diagram for the NSA–PSO hybrid model is proposed in Fig. 7. This becomes imperative in others to make clearer steps that are followed to attain the proposed model. The flow depicts the standard training and testing procedures with strict adherence. The diagram shows how the training set is kept separately from the testing set without any known knowledge of the testing set. After training, the testing set is used to evaluate the new model. This work follows the implementation of a improved model in a sequential manner in lure of respect to standard practice.

Though, the proposed improved model works at the very heart of negative selection algorithm as particle swarm optimization is initialized in the detector generation phase of the algorithm. Several research has proved that the effectiveness of any computational algorithm depends on how effective the data is represented for the learning process [31]. This is the main reason we choose to explore the combination of negative selection algorithm at the detector generation phase. When compared with other combined techniques in the field of hybrid computational system, most research implement the two systems separately by feeding the output data of one model as an input data of the next model. This makes the proposed model different from other models as the computational model was implemented at the detector generation phase of negative selection algorithm.

Fig. 7 shows the input data (sample dataset) divided in to training and testing set. The training set was used as a prospective detector (candidate detector) by implementing particle swarm optimization with local outlier factor (LOF) as a fitness function to generate detector by acquiring the local best (Pbest) of the candidate data. Euclidean distance with threshold value was further used to measure Pbest in order to generate detector in the spam space. If Pbest matched with the non-spam space, it is discarded but if

it among those that do not match with non-spam, it is accepted as a valid detector. The iteration process continues until the maximum coverage area of the spam space is attained. As represented, the testing set is separated from the training set. The testing set attributes were used in the NSA–PSO model for testing; at the end of the testing, a final output is generated through the classification of the combined scheme.

4. Empirical study, results and discussion

To carry out an empirical study, spam base dataset was acquired. The entire dataset was divided using stratified sampling approach into training and testing set in order to evaluate the performance of negative selection algorithm and the proposed improved model. 70% of the entire dataset was used for training and construction of the proposed implementation model while 30% of the remaining dataset was used for testing and validating the model. For effective comparative study of testing and validating negative selection algorithm and the newly proposed model, the commonly used standard statistical quality measure used in data mining and machine learning journals was adopted in this research. They are discussed briefly in Section 4.2.

4.1. Spam base dataset analysis

The corpus bench-mark is obtained from spam base dataset which is an acquisition from email spam message. In acquiring this email spam message, it is made up of 4601 messages and 1813 (39%) of the messages are marked to be spam messages and 2788 (61%) are identified as non-spam and was acquired by [42]. The non-spam message was contributed by Forman; this was acquired from a single mail box. Acquisition of this corpus is already pre-processed, unlike most corpuses that come in their raw form. The Instances or features are represented as 58-dimensional vectors. In the corpus of 57 features, 48 of the features of the corpus is represented by words generated from the original messages with the absence of stop list or stemming and they are considered and enlisted as most unbalanced words for the class spam. The remaining 6 features is the percentage of manifestation of the special characters ";", "(", "[", "!", "$" and "#". Some other 3 features are a representation of various measure of manifestation of capital letters that exist in the text of the messages. Lastly, it is the class label in the corpus; it gives the condition of an instance to be spam or non-spam by 1 and 0 representation. Spam base dataset is among one of the best test bed that performs good [43] during learning and evaluation techniques.

4.2. Criteria for performance evaluation

Different measures can be used to evaluate the accuracy and performance of NSA and NSA–PSO model. To evaluate and compare performance and accuracy of both models, statistical quality measure used in machine learning and data mining journals are employed. They are Sensitivity (SN), Specificity (SP), Positive prediction value (PPV), Accuracy (ACC), Negative prediction value (NPV), Correlation coefficient (CC) and F-measure (F1). See [44] for a more detailed mathematical formula. Though, they are briefly discussed below.

(i) Sensitivity (SN): The SN measures the proportion of positive pattern that are correctly recognized as positive.

$$SN = \frac{TP}{TP + FN} \tag{57}$$

(ii) Specificity (*SP*): The *SP* measures the proportion of negative pattern that are correctly recognized as negative.

$$SP = \frac{TN}{TN + FP} \tag{58}$$

(iii) Positive prediction value (*PPV*): *PPV* of a test gives a measurement of the percentage of true positives to the overall number of patterns that are recognized to be positive. It measures the probability of a positively predicted pattern as positive.

$$PPV = \frac{TP}{TP + FP} \tag{59}$$

(iv) Negative prediction value (*NPV*): *NPV* of a test also gives the measurement of percentage of true negative to the overall number of patterns recognized to be negative. It measures the probability of a negatively predicted pattern as negative.

$$NPV = \frac{TN}{FN + TN} \tag{60}$$

(v) Accuracy (*Acc*): *Acc* measures the percentage of samples correctly classified.

$$Acc = \frac{TP + TN}{TP + TN + FN + FP} \tag{61}$$

(vi) Correlation coefficient (*CC*): is used as a measure of the quality of binary (two class) classification in machine learning.

$$CC = \frac{[(TP)(TN) - (FP)(FN)]}{\sqrt{(TP + FN)(TP + FP)(TN + FP)(TN + FN)}} \tag{62}$$

(vii) *F*-measure (*F*1): It is a measure that combines both positive predictive value and sensitivity. The positive predictive value and sensitivity are evenly weighted.

$$F1 = 2 \cdot \frac{Positive\ prdictive\ value \cdot sensitivity}{Positive\ prdictive\ value + sensitivity} \tag{63}$$

(viii) Statistical *T*-test: Looks at the *t*-statistics, *t*-distribution and degrees of freedom to determine the *p* value (probability) that can be used to determine whether the mean population differ. It is a hypothesis test.

$$T = \frac{X_I - x_2}{\sqrt{\left(s_{i^2}/n_1 + s_{2^2}/n_2\right)}} \tag{64}$$

In the evaluation equation (64), TP is the number of true positive, TN is the number of true negative, FN is the number of false negative and FP is the number of false positive.

4.3. Experimental settings and implementation

The evaluation of the NSA model and the proposed NSA–PSO improved model was implemented by the division of dataset using a stratified sample approach with 70% training set and 30% testing set to investigate the performance of the new model on an unseen data. The training set was used in the construction of the model by training the dataset on both models while evaluating the capability of the model with the testing set. The process of implementation did not use any ready-made code and all functions needed are coded using Delphi 5 platform. The evaluation of both NSA model and its improved model are implemented with a threshold value of between 0.1 and 1 while the number of generated detector is between 100 and 8000. The different threshold value and the number of detector generated has tremendous impact on the final output measure.

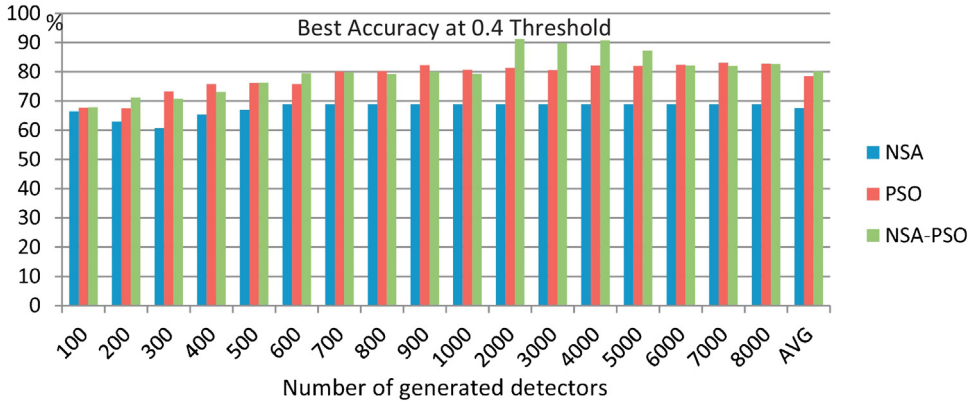

Fig. 8. Best accuracy at 0.4 threshold value for NSA, PSO and NSA–PSO.

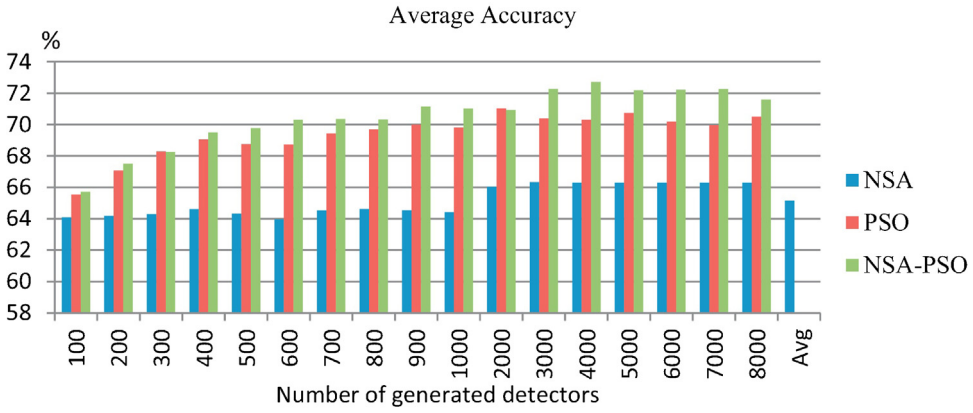

Fig. 9. Accuracy of NSA, PSO and NSA–PSO models.

5. Experimental results and discussion

The evaluation of NSA, PSO and its improved NSA–PSO model is implemented with a threshold value of between 0.1 and 1 while the number of generated detectors was between 100 and 8000. The different threshold values and number of detectors generated has tremendous impact on the final output measure. The comparison between the standard NSA and PSO model and proposed improved NSA–PSO model using validation of an unseen data is summarized in Figs. 8–11. The performance of improved NSA–PSO model outperforms the standard NSA and PSO model while the PSO model performs better than the NSA model. The proposed model shows an improved accuracy when compared with the standard models.

Fig. 8 presents the threshold value with the best accuracy and number of generated detectors with the NSA, PSO and NSA–PSO models. The threshold value is best at 0.4 and the result of all the number of generated detectors are displayed at this value. It shows the accuracy at 68.86% for the NSA, 81.31% for the PSO and 91.22% for the improved NSA–PSO. The results in Figs. 9–11 present the average of each generated detector with its threshold value for accuracy, *f*-measure and negative prediction value.

Fig. 9 shows the average accuracy of each number of generated detectors for negative selection algorithm (NSA), particle swarm optimization (PSO) and negative selection algorithm-particle swarm optimization (NSA–PSO). The proposed improved model performs better than the NSA and PSO model with the average

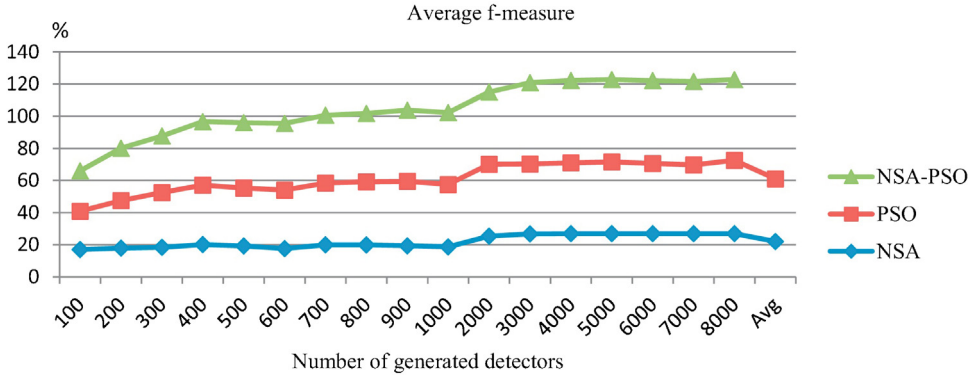

Fig. 10. F-measure of NSA, PSO and NSA–PSO models.

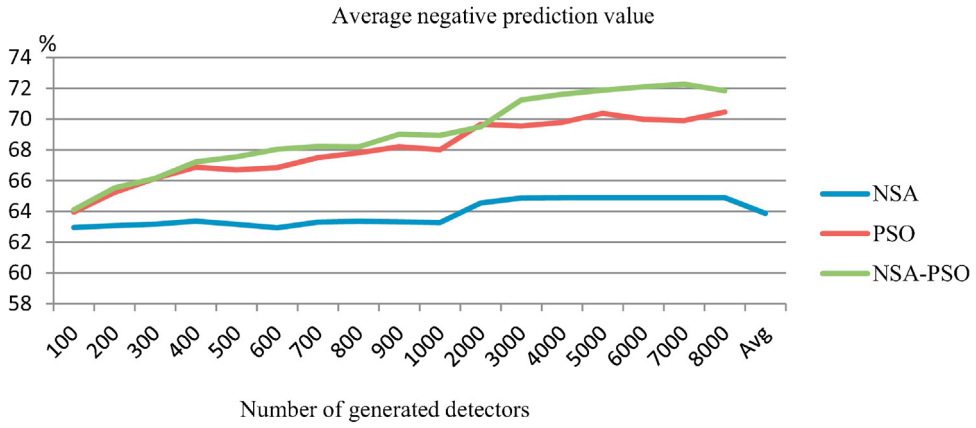

Fig. 11. Negative prediction value for NSA, PSO and NSA–PSO models.

accuracy of the proposed NSA–PSO model as shown in the graph above is at 73.61%, 66.29% for the NSA and 71.02% for the PSO model.

Fig. 10 shows the average *f*-measure of negative selection algorithm (NSA), particle swarm optimization (PSO) and negative selection algorithm-particle swarm optimization (NSA–PSO). The proposed improved models perform better than the NSA and PSO model with average f-measure of the proposed NSA–PSO model at 51.95%, 26.94% for the NSA model and 45.60% for the PSO model. *F*-measure at 100 generated detector with threshold value of 0.4, NSA–PSO model is 74.95%, 36.01% for the NSA, and 71.83% for the PSO model.

Fig. 11 shows the average negative prediction value of negative selection algorithm (NSA), particle swarm optimization (PSO) and negative selection algorithm-particle swarm optimization (NSA–PSO). The proposed improved models perform better than the NSA and PSO model with average accuracy of the proposed improved NSA–PSO model at 72.27%, 64.89% for the NSA and 70.46% for the PSO model. Negative prediction value at 6000 generated detectors with threshold value of 0.4, NSA–PSO model is 83.15%, 66.24% for the NSA and 80.41% for the PSO model. The NSA and PSO model performs lower when compared with the improved model. The improvement is on a very big scale and it shows the relevance of particle swarm optimization in improving the detector generation phase of negative selection algorithm. This practically solves the problem of detector generation and reduces the false rate as more reliable features are generated, making the standard model a robust and more effective model (Table 1).

5.1. Statistical T-test

The p value (probability) is used to determine if the populations mean differ or not. *T*-test examines the t-statistic, t-distribution and the degree of freedom in order to establish this fact. The analysis presented in Table 2 indicates a high correlation between the mean of negative selection algorithm, particle swarm optimization and improved negative selection algorithm-particle swarm optimization at 0.05 alpha levels.

Table 1
Summary and comparison of results in percentage for NSA, PSO and NSA–PSO model at 5000 generated detectors with threshold value of 0.4.

Model	ACC	CC	F1	SN	PPV	SP	NPV
NSA	68.86	48.33	36.01	22.24	94.53	99.16	66.24
PSO	81.32	60.95	71.84	60.48	88.44	94.86	78.69
NSA–PSO	91.22	63.37	74.95	65.99	86.72	93.43	80.86

Note: ACC, accuracy, CC, correlation coefficient, F1, *F* measure, SN, sensitivity, PPV, positive prediction value; SP, specificity and NPV, negative prediction value.

This shows that there is a mutual unity between negative selection algorithms, particle swarm optimization and improved negative selection algorithm-particle swarm optimization among their variables. This is corroborated by the mean of each of the negative selection algorithm, particle swarm optimization and improved negative selection algorithm-particle swarm optimization ranges between 65.1477, 69.3828 and 70.4763, respectively for accuracy and also the standard deviation indicated that there is a deviation between 0.98, 1.40 and 1.89, respectively. Other evaluation measure analysis is represented in Table 2. There is significant correlation between the mean of negative selection algorithm, particle swarm optimization and improved negative selection algorithm-particle swarm optimization. This also shows a high level of accuracy between them.

5.2. Computational complexity of NSA, PSO and NSA–PSO

The analysis of the sensitivity of different configurations is presented for NSA, PSO and NSA–PSO to determine the complexity of each algorithm in terms of its run time, change in population and the inertia weight during the computation of fitness function. All algorithms were run with the same training and testing sample over all test problems. For the run time of each algorithm, we assume T_q to be the random variable describing the time needed for each algorithm to find solution of quality q.

The cumulative distribution functions $RT_q^d(t)$ of T_q^d is called the run time distribution based on detector number d and it is defined as:

$$RT_q^d(t) = P(T_q^d \leq t) \tag{65}$$

$P(T_q^d \leq t)$ is the probability that T_q^d takes a value that is less than or equal to t

In order to estimate the run time for each of the algorithms, it is based on the number of generated detectors at each run time and the termination criteria for each run time is also based on the number of generated detectors assigned to each run time. The algorithms are run various times with different threshold value and a number of detectors. Information on solution improvement is recorded. In every run, we need to record the time or the number of critical operations and the best quality solution wherever the algorithms found a new best solution.

For each of the algorithms n run time, we compute the empirical run-time distribution as:

$$RT_q^d(t) = \frac{1}{n}\sum_{i=1}^{n} I(rt_i^d(q) \leq t) \tag{66}$$

Table 2
T-test for negative selection algorithm, particle swarm optimization and improved negative selection algorithm-particle swarm optimization.

Measure	Algorithm	t	Df($n-1$)	Mean	SD	Sig (2-tailed)	Comment
ACC	NSA	273.003	16	65.1477	0.9840	0.000	Higher correlation
	NSA–PSO	153.264	16	70.4763	1.8960	0.000	
	PSO	203.56	16	69.3828	1.4053	0.000	
F1	NSA	22.385	16	22.0938	4.0694	0.000	Higher correlation
	NSA–PSO	23.774	16	43.5390	7.5510	0.000	
	PSO	27.304	16	38.9436	5.8809	0.000	
PPV	NSA	229.009	16	85.0243	1.5308	0.000	Higher correlation
	NSA–PSO	73.737	16	81.3664	4.5497	0.000	
	PSO	93.457	16	81.7766	3.6078	0.000	
CC	NSA	40.210	16	23.2112	2.3800	0.000	Higher correlation
	NSA–PSO	40.390	16	36.4570	3.7216	0.000	
	PSO	49.363	16	33.6978	2.8146	0.000	
SN	NSA	17.166	16	13.6344	3.2748	0.000	Higher correlation
	NSA–PSO	15.450	16	32.3162	8.6248	0.000	
	PSO	17.468	16	28.6430	6.7610	0.000	
SP	NSA	778.802	16	98.6269	0.5221	0.000	Higher correlation
	NSA–PSO	142.088	16	95.2780	2.7648	0.000	
	PSO	166.871	16	95.8612	2.3686	0.000	
NPV	NSA	309.986	16	63.8717	0.8496	0.000	Higher correlation
	NSA–PSO	113.600	16	69.0260	2.5052	0.000	
	PSO	145.310	16	68.0587	1.9311	0.000	

Note: ACC, accuracy; CC, correlation coefficient; F1, *F* measure; SN, sensitivity; PPV, positive prediction value; SP, specificity and NPV, negative prediction value.

where I denotes the indicator function which is defined as:

$$I(x \le y) = \begin{cases} 1 & \textit{if, indeed, } x \le y \\ 0 & \textit{otherwise} \end{cases} \tag{67}$$

$rt_i^d(q)$ is the time required by the ith run with d number of detectors to find a solution of quality at least as good as q. A maximum time limit is set based on the number of detectors that is required at each run time. The system presents a number of detectors for each stop time from 100 to 8000 generated detectors. This determines the time each algorithm will take to reach its maximum time limit.

The population of the data required to run the system also affects the output of the systems. The number of generated detectors at each run time has tremendous effect on the output of the algorithms. An increase in the number of generated detectors at each run time increases the estimated processing time of the algorithm and also increases the performance of the algorithm; and a decrease in the number of generated detectors at each run time decreases the estimated processing time of the algorithms and also its performance. Assigning inertia weight to each candidate detector results in a run time complexity, thereby increasing the algorithm estimated run time. The best-ranked generated detectors have a maximum inertia weight during computation with fitness function. The best inertia weight during computation of reachability distance is compared with the inertia weight of its neighbour in order to attain the local outlier factor. This generates the best-ranked detectors. The computational complexity increases the run time of the algorithm.

The graph below presents a run time analysis for NSA, PSO and NSA–PSO at 1000–5000 generated detectors.

The graph in Fig. 12 shows the run time algorithm as a function of NSA, PSO and the NSA–PSO represented as $f(x)$, $g(x)$ and $h(x)$ on some subset of real valued data.

We have

$$f(x) = O(g(x)) = O(h(x)) \tag{68}$$

Or $(f(x)) = O(g(x)) = O(h(x))$ for $x \to \infty$ to be exact, *iff*$\forall$ constant N and C such that

$|f(x)| \le c|g(x)| \le |h(x)|$ for all $x > N$

This means that f does not grow faster than g and g does not grow faster than h as observed in the graph.

If a is some real value number, we then have

$$f(x) = O(g(x)) = O(h(x)) \quad \text{for } x \to a \tag{69}$$

If and only if, there exist constants $d > 0$ and c such that

$$|f(x)| \le c|g(x)| \le c|h(x)| \text{ for all } x \text{ with } |x - a| < d \tag{70}$$

The graph shows the time t in seconds it takes to generate 100 to 8000 detectors for NSA, PSO and NSA–PSO. The graph explains the run time complexity based on the number of generated detector. The NSA algorithm at 100 generated detectors has a running time of 0.764 s, at 1000 generated detector has a running time of 1.202 s and it is increased to 6.334 s at 5000 generated detectors. The PSO algorithm at 100 generated detectors has a running time of 0.935 s, at 1000 generated detector has a running time of 2.358 s and it is increased to 12.865 s at 5000 generated detectors while the NSA–PSO algorithm at 100 generated detectors has a running time of 1.365 s, at 1000 generated detectors has a running time of 4.893 s and it is increased to 17.876 s at 5000 generated detectors. The difference in computational time of NSA, PSO and NSA–PSO is due to the computational complexity during the generation of the inertial weight by the fitness function. The graph also shows the increase in the running time of the system as the number of generated detectors increased. At 8000 generated detectors, which is the highest number of generated detectors, the running time of NSA is 9.282 s, PSO is 18.023 s and NSA–PSO is 24.031 s. The run time of all generated detectors with threshold value of 0.4 is represented in Fig. 12.

5.3. Result comparison of NSA, PSO, NSA–PSO and other scheme

The result obtained from the proposed NSA–PSO model is compared with NSA, PSO and other standard machine learning algorithms in this research. The enhanced models will be compared against support vector machine (SVM) proposed by Fagbola et al. [45], distinguishing feature selection and support vector machine (DFS-SVM) proposed by Uysal and Gunal [46], artificial neural network proposed by Özgür et al. [47], naïve Bayes (NB) proposed

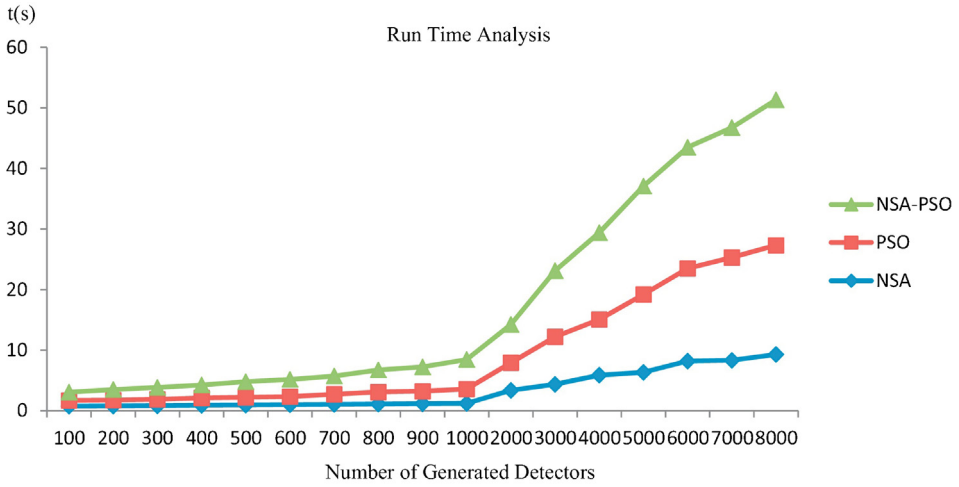

Fig. 12. Run time analysis for NSA, PSO and NSA–PSO.

by Zhang [48] and interval type-2 fuzzy set proposed by Ariaeinejad and Sadeghian [49]. These standard machine-learning tools are used for comparison with our proposed model. The proposed model shows high accuracy in detecting email spam. Table 3 shows in summary the result analysis of all the models. The discussion of the result on the individual model is presented shortly.

Though, the difference in performance between the proposed NSA–PSO model with the NSA and PSO models are very significant, the best accuracy of the proposed model is 91.22%, while the NSA model is 68.86% and the PSO model is 81.32% respectively. In general, the proposed model outperforms the standard NSA and PSO models. The comparison of the proposed model with the state of the art machine learning models shows that our model performs better than all the models listed in Table 3 as shown above. The proposed model outperforms the standard naïve Bayes models proposed by Zhang et al. [48,50] with accuracy of both models at 78.8% and 79.3% respectively. The model also performs better than the support vector machine (SVM) proposed by Fagbola et al. [45] with accuracy of 90% and the distinguishing feature selection and support vector machine (DFS-SVM) proposed by Uysal and Gunal [46] with accuracy of 71%. The accuracy of artificial neural network (ANN) model proposed by Özgür et al. [47] is 86% while accuracy of interval type-2 fuzzy set proposed by Ariaeinejad and Sadeghian [49] is 86.9% when compared to the proposed model.

6. Model implementation and advantages

The amount of email spam spreading across the network is a critical problem in today's world. Different means have been devised in the propagation of email spam and network security [51]. Despite the improvement in technology, the spammers adapt to new techniques. The proposed algorithm compares the NSA, PSO and the NSA–PSO models. The models were evaluated with statistical tools to determine the best model to be used for email spam detection. From our analysis, the NSA–PSO model performs better than the NSA and the PSO models. Therefore, the proposed spam detection architecture is constructed based on the NSA–PSO model. The algorithm can be considered as a powerful approach in the detection of email spam due to its adaptive nature. The spam and non-spam email can be separated based on the adaptation of the email spam detector; the probability of spam future occurrence is based on the spam best occurrence. If there is a frequent occurrence of any part of the email in the spam email and not in the non-spam email, it is prone to be identified as spam. There is a certification of the content of the email by the NSA–PSO model against the information exchange in the database. With respect to the information, the bounded knowledge of spam is deleted. Messages used in an email could be spam in a database and non-spam in another database. The proposed algorithm makes verification with reference to the number of times it occurred in a database and detect spam base on probability ratio. The importance of the proposed model is that it computes the detection of spam based on patterns [52]. The blocked messages are identified against its spamicity rather than the respective messages for a better and more efficient detection of spam contents in an email [53].

Table 3
Testing results comparison for NSA, PSO, NSA–PSO and other scheme.

Classifier	Accuracy (%)
NSA–PSO	91.22
PSO	81.32
NSA	68.86
NB [48]	79.3
SVM [45]	90
DFS-SVM [46]	71
ANN [47]	86
Type-2 Fuzzy Set [49]	86.9

The tool represents a client and server connection in an organization. As shown in Fig. 13, client 1 and 2 are able to communicate in and outside a network. The sent and received messages by the client to other machines are routed through server 1. The server 1 sent the email and detects spam's that are sent by the client. The server receives the email and delivers it to the corresponding destination nodes if the email is spam free. The spam is detected by the server software based on NSA–PSO spam detector model [54] which differentiates between the spam email and the non-spam email.

The proposed architecture with the NSA–PSO implementation model is important in securing the system based on its adaptive nature. Existing problems of email spam detection where spammers are able to manipulate the spam messages that are sent to the system through obfuscation of messages is eliminated due to the adaptive nature of the proposed model. Messages that pass through the proposed model are recognized by this model through adaptation as spam or non-spam. A frequent occurrence in the spam email on the database of the model that is not in the non-spam email allows for the system to be identified as a spam email. The memory of the spam and non-spam detector in the database of the proposed model is able to learn and keep records of the previous spam or non-spam email message.

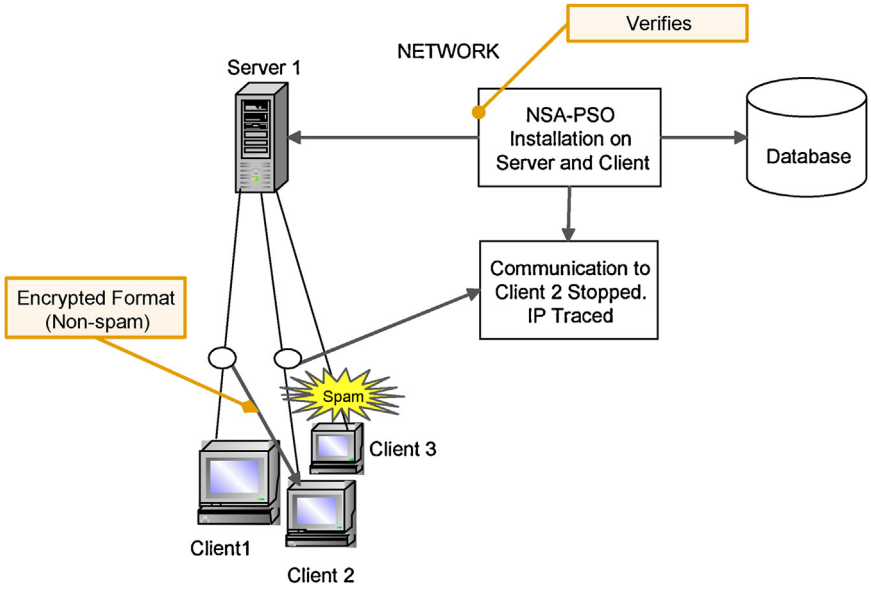

Fig. 13. System architecture.

7. Conclusion and recommendation

In this research, a new improved model that combines negative selection algorithm (NSA) with particle swarm optimization (PSO) has been proposed and implemented. The uniqueness of this model is that PSO was implemented at the random generation phase of NSA. The detector generation phase of NSA determines how robust and effective the algorithm will perform. PSO implementation with local outlier factor (LOF) as fitness function no doubt improved the detector generation phase of NSA. The proposed improved model serves as a better replacement to NSA model. Spam-base dataset was used to investigate the performance of NSA and PSO model against improved NSA–PSO model. Performance and accuracy investigation has shown that the proposed improved model is able to detect email spam better than the NSA and PSO model. In totality, the empirical report has shown the superiority of the proposed NSA–PSO improved model over the NSA and PSO model. The proposed improved systems will be useful in other applications since negative selection algorithm solves a vast number of complex problems. This research should be viewed as an improvement in the field of computational intelligence. Based on the promising result generated from the research; as future work, it is suggested that this research should be considered as a viable tool for any newly proposed system in email spam detection problem that is based on detector generation. Future work will be a parallel hybridization of the two evolutionary algorithms to perform a single task of detector generation.

Acknowledgements

The Universiti Teknologi Malaysia (UTM) under IDF Scholarships, Ministry of Higher Education (MOHE) Malaysia under research grant Vot: R.J130000.7828.4F087 and under Research University Funding Scheme Universiti Teknologi Malaysia (Q.J130000.2510.03H02) are hereby acknowledged for some of the facilities utilized during the course of this research work and for supporting the related research.

References

[1] S. Whittaker, V. Bellotti, P. Moody, Introduction to this special issue on revisiting and reinventing e-mail, Hum.-Comput. Interact. 20 (1–2) (2005) 1–9.
[2] W.A. Awad, S.M. ELseuofi, Machine learning methods for spam e-mail classification, Int. J. Comput. Sci. Inf. Technol. 3 (1) (2011).
[3] H. Wang, et al., PSO-Optimized Negative Selection Algorithm for Anomaly Detection in Applications of Soft Computing, Springer, Berlin/Heidelberg, 2009, pp. 13–21.
[4] X.Z. Gao, S.J. Ovaska, X. Wang, Particle swarm optimization of detectors in negative selection algorithm, in: IEEE International Conference on Systems, Man and Cybernetics (ISIC, 2007), 2007.
[5] S. Afaneh, R.A. Zitar, A. Al-Hamami, Virus detection using clonal selection algorithm with genetic algorithm (VDC algorithm), Appl. Soft Comput. 13 (1) (2013) 239–246.
[6] S.X. Wu, W. Banzhaf, The use of computational intelligence in intrusion detection systems: a review, Appl. Soft Comput. 10 (1) (2010) 1–35.
[7] Z. Jinquan, et al., A self-adaptive negative selection algorithm used for anomaly detection, Prog. Natl. Sci. 19 (2) (2009) 261–266.
[8] V. Golovko, et al., Evolution of immune detectors in intelligent security system for malware detection, in: IEEE 6th International Conference on Intelligent Data Acquisition and Advanced Computing Systems (IDAACS), 2011.
[9] L.N. De Castro, J. Timmis, Artificial Immune Systems: A New Computational Approach, Springer-Verlag, London, UK, 2002, pp. 357.
[10] D. Dasgupta, Advances in artificial immune systems, IEEE Comput. Intell. Mag. 1 (4) (2006) 40–49.
[11] T.S. Guzella, W.M. Caminhas, A review of machine learning approaches to spam filtering, Expert Syst. Appl. 36 (7) (2009) 10206–10222.
[12] T. Oda, T. White, in: E. Cantú-Paz, et al. (Eds.), Developing an Immunity to Spam, in Genetic and Evolutionary Computation — GECCO 2003, Springer, Berlin/Heidelberg, 2003, pp. 231–242.
[13] T. Oda, T. White, Increasing the accuracy of a spam-detecting artificial immune system, in: The 2003 Congress on Evolutionary Computation (CEC 2003), 2003.
[14] T. Oda, T. White, Immunity from spam: an analysis of an artificial immune system for junk email detection, in: C Jacob, et al. (Eds.), Artificial Immune Systems, Springer, Berlin/Heidelberg, 2005, pp. 276–289.
[15] S. Sarafijanovic, J.-Y. Le Boudec, Artificial immune system for collaborative spam filtering, in: The Second Workshop on Nature Inspired Cooperative Strategies for Optimization (NICSO 2007), Acireale, Italy, November 8–10, 2007, Springer-Verlag, Acireale, Italy, 2008.
[16] G. Bezerra, et al., in: H. Bersini, J. Carneiro (Eds.), An Immunological Filter for Spam, in Artificial Immune Systems, Springer, Berlin/Heidelberg, 2006, pp. 446–458.
[17] A.H. Mohammad, R.A. Zitar, Application of genetic optimized artificial immune system and neural networks in spam detection, Appl. Soft Comput. 11 (4) (2011) 3827–3845.
[18] T.S. Guzella, et al., Identification of SPAM messages using an approach inspired on the immune system, Biosystems 92 (3) (2008) 215–225.
[19] A. Visconti, H. Tahayori, Artificial immune system based on interval type-2 fuzzy set paradigm, Appl. Soft Comput. 11 (6) (2011) 4055–4063.
[20] N. Pérez-Díaz, et al., SDAI: an integral evaluation methodology for content-based spam filtering models, Expert Syst. Appl. 39 (16) (2012) 12487–12500.
[21] P. D'Haeseleer, An immunological approach to change detection: theoretical results, in: Proceedings of 9th IEEEComputer Security Foundations Workshop, 1996.
[22] M. Ayara, et al., Negative selection: how to generate detector, in: 1st International Conference on Artificial Immune Systems, 2002, pp. 89–98.

[23] F. Esponda, S. Forrest, P. Helman, A formal framework for positive and negative detection schemes, IEEE Trans. Syst. Man. Cybern. 34 (1) (2004) 357–373.
[24] S.T. Wierzchon, Discriminative power of the receptors activated by k-contiguous bits rule, J. Comput. Sci. Technol. 1 (3) (2000) 1–13.
[25] G. Fabio, et al., A randomized real-valued negative selection algorithm, in: Proceedings of Second International Conference on Artificial Immune System (ICARIS), 2003.
[26] D. Dasgupta, F. Gonzalez, An immunity-based technique to characterize intrusions in computer networks, IEEE Trans. Evol. Comput. 6 (3) (2002) 281–291.
[27] J. Balthrop, S. Forrest, M.R. Glickman, Revisiting LISYS: parameters and normal behavior, in: Proceedings of the 2002 Congress on Evolutionary Computing, 2002.
[28] S. Forrest, A.S. Perelson, Self Nonself Discrimination in Computer, 1994.
[29] J. Textor, A comparative study of negative selection based anomaly detection in sequence data, in: C. Coello Coello, et al. (Eds.), Artificial Immune Systems, Springer, Berlin/Heidelberg, 2012, pp. 28–41.
[30] J. Chen, D. Yang, A study of detector generation algorithms based on artificial immune in intrusion detection system, in: 3rd International Conference on Computer Research and Development (ICCRD, 2011), 2011.
[31] M. Gong, et al., An efficient negative selection algorithm with further training for anomaly detection, Knowl.-Based Syst. 30 (2012) 185–191.
[32] B. Sirisanyalak, O. Sornil, An artificial immunity-based spam detection system, in: IEEE Congress on Evolutionary Computation (CEC, 2007), 2007.
[33] Z. Wenqing, Z. Zili, An email classification model based on rough set theory, in: Proceedings of the 2005 International Conference on Active Media Technology, 2005 (AMT, 2005), 2005.
[34] A. Bratko, et al., Spam filtering using statistical data compression models, J. Mach. Learn. Res. 7 (2006) 2673–2698.
[35] J. Gordillo, E. Conde, An HMM for detecting spam mail, Expert Syst. Appl. 33 (3) (2007) 667–682.
[36] R. Guangchen, T. Ying, Intelligent detection approaches for spam, in: Third International Conference on Natural Computation (ICNC), 2007.
[37] C. Wang, Y. Zhao, A new fault detection method based on artificial immune systems, Asia-Pac. J. Chem. Eng. 3 (6) (2008) 706–711.
[38] F. Gonzalez, et al., An evolutionary approach to generate fuzzy anomaly (attack) signatures, in: IEEE Systems, Man and Cybernetics Society Information Assurance Workshop, 2003, 2003.
[39] T.A. Sajesh, M.R. Srinivasan, Outlier detection for high dimensional data using the Comedian approach, J. Stat. Comput. Simulat. 82 (5) (2011) 745–757.
[40] S. Ramaswamy, R. Rastogi, K. Shim, Efficient algorithms for mining outliers from large data sets, SIGMOD Rec. 29 (2) (2000) 427–438.
[41] E.M. Knorr, R.T. Ng, Algorithms for mining distance-based outliers in large datasets, in: Proceedings of the 24rd International Conference on Very Large Data Bases, Morgan Kaufmann Publishers Inc., 1998, pp. 392–403.
[42] M. Hopkins, et al., UCI Machine Learning Repository: Spambase Data Set, Hewlett-Packard Labs, 1999, https://archive.ics.uci.edu/ml/datasets/Spambase
[43] I. Koprinska, Learning to classify e-mail, Inf. Sci. 177 (2007).
[44] B. Biggio, et al., A survey and experimental evaluation of image spam filtering techniques, Pattern Recognit. Lett. 32 (10) (2011) 1436–1446.
[45] T. Fagbola, S. Olabiyisi, A. Adigun, Hybrid GA-SVM for efficient feature selection in e-mail classification, Comput. Eng. Intell. Syst. 3 (3) (2012).
[46] A.K. Uysal, S. Gunal, A novel probabilistic feature selection method for text classification, Knowl.-Based Syst. 36 (2012) 226–235.
[47] L. Özgür, T. Güngör, F. Gürgen, Spam mail detection using artificial neural network and bayesian filter, in: Z. Yang, H. Yin, R. Everson (Eds.), Intelligent Data Engineering and Automated Learning – IDEAL 2004, Springer, Berlin/Heidelberg, 2004, pp. 505–510.
[48] Y. Zhang, et al., Applying cost-sensitive multiobjective genetic programming to feature extraction for spam e-mail filtering, in: M. O'Neill, et al. (Eds.), Genetic Programming, Berlin/Heidelberg, Springer, 2008, pp. 325–336.
[49] R. Ariaeinejad, A. Sadeghian, Spam detection system: A new approach based on interval type-2 fuzzy sets, in: 24th Canadian Conference on Electrical and Computer Engineering (CCECE, 2011), 2011.
[50] S. Abu-Nimeh, et al., Bayesian additive regression trees-based spam detection for enhanced email privacy, in: Third International Conference on Availability, Reliability and Security (2008 ARES 08), 2008.
[51] Y. Wang, et al., Modeling and security analysis of enterprise network using attack–defense stochastic game Petri nets, Secur. Commun. Netw. 6 (1) (2013) 89–99.
[52] W. Haiyan, Z. Runsheng, W. Yi, An anti-spam filtering system based on the naive Bayesian classifier and distributed checksum clearinghouse, in: Third International Symposium on Intelligent Information Technology Application (IITA), 2009.
[53] T. Gong, B. Bhargava, Immunizing mobile ad hoc networks against collaborative attacks using cooperative immune model, Secur. Commun. Netw. 6 (1) (2013) 58–68.
[54] O. Amayri, N. Bouguila, Content-based spam filtering using hybrid generative discriminative learning of both textual and visual features, in: IEEE International Symposium on Circuits and Systems (ISCAS), 2012.