Engineering Applications of Artificial Intelligence 39 (2015) 33–44

Contents lists available at ScienceDirect

Engineering Applications of Artificial Intelligence

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

A combined negative selection algorithm–particle swarm optimization for an email spam detection system

Ismaila Idris [a], Ali Selamat [a,b,e,*], Ngoc Thanh Nguyen [c], Sigeru Omatu [d], Ondrej Krejcar [e], Kamil Kuca [e], Marek Penhaker [f]

[a] Faculty of Computing, Universiti Teknologi Malaysia, 81310 UTM Johor Bahru, Johor, Malaysia
[b] UTM-IRDA Digital Media Center of Excellence, Universiti Teknologi Malaysia, 81310 UTM Johor Bahru, Johor, Malaysia
[c] Knowledge Management Systems Division, Wroclaw University of Technology, Wybrzeże Wyspiańskiego 27, 50-370 Wrocław, Poland
[d] Department of Electronics, Information and Communication Engineering, Faculty of Engineering, Osaka Institute of Technology, 5-16-1 Omiya, Asahiku, Osaka 535-8585, Japan
[e] University of Hradec Kralove, FIM, Center for Basic and Applied Research, Rokitanskeho, 62, Hradec Kralove, 500 03, Czech Republic
[f] Department of Cybernetics and Biomedical Engineering, Faculty of Electrical Engineering and Computer Science, VSB-Technical University of Ostrava, 17. Listopadu 15, 708 33 Ostrava-Poruba, Czech Republic

ARTICLE INFO

Article history:
Received 9 February 2014
Received in revised form
29 October 2014
Accepted 1 November 2014

Keywords:
Negative selection algorithm
Differential evolution
Particle swarm optimization
spam detectors

ABSTRACT

Email is a convenient means of communication throughout the entire world today. The increased popularity of email spam in both text and images requires a real-time protection mechanism for the media flow. The previous approach has been limited by the adaptive nature of unsolicited email spam. This research introduces an email detection system that is designed based on an improvement in the negative selection algorithm. Furthermore, particle swarm optimization (PSO) was implemented to improve the random detector generation in the negative selection algorithm (NSA). The algorithm generates detectors in the random detector generation phase of the negative selection algorithm. The combined NSA–PSO uses a local outlier factor (LOF) as the fitness function for the detector generation. The detector generation process is terminated when the expected spam coverage is reached. A distance measure and a threshold value are employed to enhance the distinctiveness between the non-spam and spam detectors after the detector generation. The implementation and evaluation of the models are analyzed. The results show that the accuracy of the proposed NSA–PSO model is better than the accuracy of the standard NSA model. The proposed model with the best accuracy is further used to differentiate between spam and non-spam in a network that is developed based on a client–server network for spam detection.

1. Introduction

Different techniques have been adopted to stop the threat of spam or to drastically reduce the amount of spam that attacks internet users across the world. An anti-spam law was enacted by legislating a penalty for spammers who distribute email spam (Bambauer, 2005). Additionally, two general approaches have been used in email spam detection: a knowledge engineering approach and a machine learning approach (Wamli et al., 2009). The knowledge engineering approach uses network information and internet protocol address techniques to determine whether a message is spam or non-spam; this approach is known as the origin-based filter. Sets of rules must be specified in the knowledge engineering approach to determine which email is to be categorized as spam or non-spam. Such rules could be created by the use of filters or by some other authority (Bambauer, 2005). An example of this process is a software company that provides specific rule-based spam filtering tools. However, the rules must be maintained continuously and must be updated, which is a waste of time and is inconvenient for most users (Thonnard et al., 2012). The machine learning approach is more efficient than the knowledge engineering approach (Guzella and Caminhas, 2009) and does not require specifying rules; instead, a set of pre-classified email messages is utilized. Specific algorithms are used to learn the classification rules from the email messages. Filtering techniques are the most commonly used methods; the system identifies whether a message is spam or non-spam based solely on the message content and some other characteristics of the message (Man and Mousoli, 2010). Despite the different approaches and techniques that have been adopted to fight the threat of email spam, the internet today still manifests an enormous amount of

* Corresponding author at: Faculty of Computing, Universiti Teknologi Malaysia, 81310 UTM Johor Bahru, Johor, Malaysia.
E-mail address: aselamat@utm.my (A. Selamat).

http://dx.doi.org/10.1016/j.engappai.2014.11.001
0952-1976/

spam (Zhang et al., 2004; Massey, 2003; Delany et al., 2012), and more attention is required with regard to how the threat can be drastically reduced if not totally eliminated. The battle against email spam is a very difficult battle; therefore, it makes sense to fight an adaptive email spam generator with an adaptive system. Most models emphasize applying and designing computational algorithms and techniques with the use of simplified models of different immunological processes (De Castro and Timmis, 2002; Dasgupta, 2006; Almeida and Yamakami, 2012; Sheikhan and Sharifi Rad, 2013). This paper proposes an improved solution to email spam detection by replacing the random detector generation in the negative selection algorithm (NSA) with particle swarm optimization (PSO). PSO is implemented with a local outlier factor as a fitness function to generate detectors in a negative selection algorithm.

The remainder of this paper is organized as follows. Section 2 discusses the related studies on negative selection algorithms. The proposed NSA–PSO and its constituent framework are presented in Section 3. An empirical study and dataset analysis are presented in Section 4. Section 5 presents the implementation, results and discussion. Finally, the conclusions and recommendations are presented in Section 6.

2. Related studies

The understanding of the artificial immune system (AIS) approach, which is based on the mammalian immune system, is vital for this study. A comprehensive artificial immune system survey has been provided in (Dasgupta et al., 2011). This paper discusses the history, recent developments and four major AIS algorithms. The main goal of the immune system is to distinguish between non-self and self elements, which is the basis of our implementation with the negative selection algorithm (NSA). This research will replace 'self' in the mammalian immune system with 'non-spam' in our system and 'non-self' in the mammalian immune system with 'spam' in our system. Most of the work on the negative selection algorithm (NSA) and particle swarm optimization (PSO) solves problems in anomaly detection and intrusion detection. No previous research implements PSO to generate detectors in a negative selection algorithm. The implementation of particle swarm optimization with the negative selection algorithm to maximize the coverage of the non-self space was proposed by Wang et al. (2009) to solve the problem of anomaly detection. In Gao et al. (2007), the focus is on non-overlapping detectors that have fixed sizes, to achieve maximal coverage of the non-self space; this approach is initiated after the generation of detectors by a negative selection algorithm. The artificial immune system (AIS) is a new mechanism that is implemented for the control of email spam. Pattern matching was used to represent detectors as regular expressions by (Oda and White, 2003a) in the analysis of messages. A weight is assigned to the detector; this weight is decremented or incremented when observing the expression in the spam message, and the classification of the message is based on the threshold sum of the weight of the matching detectors. This system is intended to be corrected by either increasing or decreasing all of the matching detector weights with 1000 detectors generated from a spam-assassin heuristic and a personal corpus. The results were acceptable based on the small number of detectors that was used. A comparison of two techniques to determine the message classification using a spam-assassin corpus with 100 detectors was proposed by (Oda and White, 2003b). This approach is similar to previous techniques, but the difference is the increment in the weight when there is recognition of patterns in spam messages. A random generation of detectors does not help in solving the problem of finding the best selected features; however, the feature weights are updated during the matching process. The weighting of the features complicates the performance of the matching process. More experiments were performed by Oda and White (2005) with the use of a spam-assassin corpus and a Bayesian combination of the detector weights. The messages are scored by the simple sum of the message matched by each non-spam in the detector space and also the use of Bayes scores. Words from the dictionary and patterns extracted from the set of messages are considered in the detector generation in addition to the commonly used filters, to be assured of the message classification. It was finally observed that the best results emerged when the heuristic was used and that it had a similar performance to the other two techniques. The immune system classifies correctly 90% of the messages. More specifically, it classifies 84% of the spam and 98% of the non-spam. The approach of scoring features or feature weighting during and after the matching process creates ambiguity in the selection of important features for spam detection due to its computational cost.

The work of Wamli et al. (2009) studies the possibility of using negative selection in email spam detection without prior information of the email spam. The negative selection algorithm is divided into four concurrent working modules with two repositories: the random detector generation module, the detector maturing module, the antigen matching module and the detector aging module, with a selves' repository and a detectors repository. The TREC07 corpus (Cormack and Lynam, 2007) was used in its implementation. After the initial 1/3 of the time during the learning period, the spam detection rate is over 80%, and it is over 70% most of the time. A new solution to solve the spam detection problem, which is inspired by the adaptive immune system model, is called the cross-regulation model and was presented in Abi-Haidar and Rocha (2008). This research shows the relevance of the cross-regulation model as a biologically inspired algorithm in the detection of spam. The Enron corpus was used in its implementation with the 70% spam experiment. The accuracy and *F*-measure are 83% and 79%, respectively.

The analysis of major work performed on negative selection algorithms with a combination of two different algorithms in a hybrid email spam model is contained in Sirisanyalak and Sornil (2007). An AIS-based module that extracts features was designed and further used for a logistic regression model; the set of detectors was initially generated using terms that were extracted from the training message and using data from matched detectors that were used in the regression model. The experiment uses spam-assassin. A genetic optimized AIS culled old lymphocytes (replacing the old lymphocytes with new ones) and also checked for new interests for users, using an approach that was similar to that presented in Hamdan and Abu (2011), to update intervals such as the number of received messages. An interval is updated with respect to time, user requests and other factors; many choices were used in selecting the update intervals, which was the aim of using the genetic algorithm. The experiment was implemented with a spam-assassin corpus that had 4147 non-spam messages and 1764 spam messages. The optimized spam detector with 600 generated detectors gives a false positive rate of 1.1% and a false negative rate of 3.7%, while spam detection with AIS and 600 generated detectors gives a false positive rate of 1.2% and a false negative rate of 4.9%. Other optimized algorithms are presented in Yildiz (2013), Mazhoud et al. (2013), Tenne (2012). A proposed anti-spam filter with an evolutionary algorithm is presented in Yevseyeva et al. (2013). The scores of the anti-spam filters are optimized to improve their accuracy. The optimization problem is considered in a single- and multi-objective problem formulation. Rough set theory, which is a mathematical approach for approximate reasoning, was proposed in Wenqing and Zili (2005) to group messages into three classes while targeting a low false positive rate. The selection of features into spam, non-spam or

suspicious elements was first implemented on the training set, after which a genetic algorithm was implemented. The universe of messages was divided into three regions based on an induced set of rules. The experiment used only 11 features of the UCI corpus (Hopkins et al., 1999). It was concluded that the technique is very efficient in reducing the number of non-spam messages that are blocked. The work in Bereta and Burczyński (2007) combines the characteristics of negative selection and clonal selection to select the best subset of features for classification. A combination of support vector machine (SVM) and artificial immune system (AIS) was proposed by Guangchen and Ying (2007), with the use of binary features that have the same feature selection as in Bezerra et al. (2006). The support vectors that were acquired after training the SVM were implemented in the generation of the initial detector set for AIS; AIS was then used for classification. During the classification with AIS, the detector with the smallest Euclidean distance to the message was added to the committee set, from which there was a majority voting of the detectors in the set to obtain the classification. PU1 corpora and Ling-spam corpora were used for the experiment. The application of an integral evaluation methodology to compare eight different well-known content-based spam filtering techniques with the use of well-known accuracy measures was presented by Pérez-Díaz et al. (2012). The measures are based on the filter accuracy in four different complimentary scenarios. The scenarios are static, dynamic, adaptive and internationalization. Basically, the idea of an integral evaluation methodology is to cover the gap that was present between basic research and the deployment of existing machine learning algorithms for spam filtering.

3. Proposed NSA–PSO and its framework

A combination of negative selection algorithm-particle swarm optimization (NSA–PSO) was investigated to compliment the parameters of each component of the system; this approach uses the advantages of the individual system against its disadvantages while elevating each weak component member of both systems to achieve stability, consistency and accurate intelligent systems that are extendable for usage in classification. The importance of a combined system is not negotiable because of the fact that an individual system has its own weaknesses and a combined system is meant to compliment the weaknesses of these individual intelligent systems. The strength of the particle swarm optimization is combined with that of the negative selection algorithm to improve the weaknesses of both algorithms by means of the strength of their combination. A local outlier factor is also implemented as a fitness function for the particle swarm optimization.

3.1. The NSA–PSO

Random generation of detectors by the real value negative selection algorithm is improved with the introduction of particle swarm optimization (PSO) and the local outlier factor (LOF) as the fitness function. This improvement is a result of the quest for an efficiently trained negative selection algorithm model for purely normal detectors. The local outlier factor as a fitness function will model the data point by the implementation of a stochastic distribution (Sajesh and Srinivasan, 2011). This proposed technique can improve the traditional random generation of detectors in the real value negative selection algorithm and optimize the generated detectors in spam space at the same time. The section below explains the processes of its implementation.

3.1.1. Spam and non-spam space

In the case of the real value negative selection algorithm, there is a need to define the non-spam and spam space. The non-spam space is the normal state of a system, while the spam space is the abnormal state of a system.

Let us assume the non-spam space to be S, where S is defined as follows:

$$S = (s_{1\ldots}s_n) = \begin{bmatrix} s_{11} & \cdots & s_{1m} \\ \vdots & \ddots & \vdots \\ s_{n1} & \cdots & s_{nm} \end{bmatrix},$$

where $S_{ij} \in K^m$, $i = 1, \cdots, n$ and $j = 1, \cdots, m$

The space S is normalized as follows:

$$S_i = \frac{S_i}{||S_i||} \tag{1}$$

Therefore, s_i is the i^{th} non-spam unit, and s_{ij} is the j^{th} vector of the i^{th} non-spam unit.

3.1.2. Generate random candidate detectors

In this scenario, we generate random candidate detectors as follows:

$$r = (r_{1\ \ldots}\ r_n) = \begin{bmatrix} r_{11} & \cdots & r_{1m} \\ \vdots & \ddots & \vdots \\ r_{k1} & \cdots & r_{km} \end{bmatrix}, \tag{2}$$

where $r_{ij} \in (0,1)^m$, $i = 1, \ldots, k$ and $j = 1, \ldots, m$. Here, r_i is said to be the ith detector, and r_{ij} is the jth feature of the ith detector.

3.2. Detector generation parameters and implementation in particle swarm optimization

The particles are composed of 57 features $\{f_{57}\}$. The value of the accelerated constant C is 0.5. The position and velocity of the particle swarm optimization are represented, respectively, in N-dimensional vector space as follows:

$$P_i\ (p_{i,}^1\ \ p_{i,\ \ldots}^2 p_i^n) \tag{3}$$

and

$$V_i\ (v_{i,}^1\ \ v_{i,\ \ldots}^2 v_i^n) \tag{4}$$

Here, p_{id} denotes the binary bits $i = 1, 2, \ldots, m$ (m is the total number of particles), and $d = 1, 2, \ldots, n$ (n is the dimensionality of the data). Each particle in the generation updates its own position and velocity based on Eqs. (3) and (4). The initialization of the real-valued PSO is established by the population of particles (non-spam and spam). All of the particles move in problem space to find the optimal solution over all of the iterations.

Given n-dimensional space, the particles exhibit a potential solution while each particle possesses a direction and position vector for its movement and direction. To determine the best particles, we use the local outlier factor (LOF) as a fitness function for the system. The proposed PSO requires one best optimum solution, and each generated candidate detector (P_{best}) is used as the optimum solution in spam space. The global best (G_{best}) solution is eliminated because our solution does not need to jump from one optimum solution to another and each candidate detector is a potential optimum solution in the swarm; each local best (p_{best}) particle is the optimum solution that is reached after the particles in the swarm are compared. We do not have a unique optimal solution in our problem that will require the use of G_{best} to determine the optimal solution. The G_{best} solution takes us to cover another space instead of covering the immediate position or space. Therefore, the movement that was attained using G_{best} is too long

compared with the movement that is required to cover the spam space for the purpose of detector generation.

In generating a random initial velocity matrix for the random candidate detectors, we have $v(0)$, as follows:

$$v(0)=\begin{bmatrix} v_{1\,1}(0) & \cdots & v_{1\,m}(0) \\ \vdots & \ddots & \vdots \\ v_{j\,1}(0) & \ldots & v_{j\,m}(0) \end{bmatrix} \tag{5}$$

Eqs. (6) and (7) calculate the new velocity and particle positions as follows:

$$v_{id}(t+1) = v_{id}(t) + c(\text{Pbest}_{id}(t) - x_{id}(t)) \tag{6}$$

$$x_{id}(t+1) = x_{id}(t) + v_{id}(t+1) \tag{7}$$

The process of the proposed method can be explained in the following steps:

Step 1: Define a stable behavior and the activities of a system as non-spam space (normal pattern), as shown in Eq. (1).
Step 2: From the population of spam and non-spam data, generate training and testing profiles with random candidate detectors as shown in Eq. (2).
Step 3: Use Eqs. (3) and (4) to initialize both the position and the velocity of the PSO.
Step 4:Calculate the reach-ability distance and the LOF for each candidate detector, as shown in Eqs. (22) and (23).
Step 5: Update each candidate detector position and velocity with Eqs. (6) and (7).
Step 6: Implement the distance measure with Eq. (42) and the threshold value with Eq. (43) to determine the P_{best} similarity in the non-spam space S. If p_{best} does not match S, then it is a valid detector.
Step 7: Continuously generate and match P_{best} against S to observe changes. Deviation of the system could occur if p_{best} matches S. P_{best} is not intended to match S.
Step 8: After the maximum coverage in spam space, employ the testing set for evaluation.

Fig. 1 presents the particle swarm optimization (PSO) algorithm.

```
//pb is the local best
//gb is the global best
//LOF is the local outlier factor
//p is the population size
[1]  Initialize all of the 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] //Because we need pbest as the optimal solution
[12]  //Update the local best position (pb_i)
[13]              If f(pb_i) = f(gb) then
[14]          pb_i = gb
[14]       End if
[15]  End for
[16]  //Update the particle velocity and position
     For each particle i in p do
[17]  For each dimension d in D do
[18] 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}]
[19]  x_{i,d} = x_{i,d} + v_{i,d}
[20]     End for
[21]  End for
[22]  While the maximum iteration or stop criteria is reached
```

Fig. 1. Particle swarm optimization algorithm.

3.2.1. Implementation model

The N-dimensional points and a non-spam radius Rs represent the training dataset.

Let Eq. (8) represents the non-spam space as follows:

$$S = \{X_i | i = 1, 2, \cdots, m; Rs = r\}, \tag{8}$$

where X_i are some points in the normalized N-dimensional space, i.e.

$$X_i = \{x_{i1}, x_{i2}, x_{i3} \cdots x_{iN}\}, i = 1, 2, 3, \cdots, m \tag{9}$$

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

$$x_i = (x_{i1}, x_{i2,\ldots}, x_{iD}), \tag{10}$$

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

$$v_i = (v_{i1}, v_{i2}, \ldots, v_{iD}). \tag{11}$$

The movements of the particles were influenced by an implemented memory. In the cognitive memory

$$p_i = (p_{i1}, p_{i2}, \ldots, p_{iD}). \tag{12}$$

The best previous position that was visited by each individual particle i is stored:

$$p_{best} = (p_{best1}, p_{best2}, \ldots, p_{bestD}). \tag{13}$$

The vector in Eq. (13) contains the position of the best point in the search space that was visited by all of the particles.

At each iteration, each p_{best} is used to compute the density of the local neighborhood.

$$\text{lrd}(i) = \frac{1}{\left(\frac{\sum_{s \in N_k(i)} \text{reachability} - \text{distance}_k(i,s)}{|N_K(i)|}\right)}. \tag{14}$$

Afterward, the local reach-ability density is compared with that of its neighboring reach-ability distances

$$\text{LOF}_K(i) = \frac{\sum_{s \in N_k(i)} (\text{lrd}(s))/\text{lrd}(i)}{|N_k(i)|} = \frac{\sum_{s \in N_k(i)} \text{lrd}(s)/|N_K(i)|}{\text{lrd}(i)}. \tag{15}$$

Giving each particle a degree of being an outlier, each iteration of the p_{best} velocity is updated according to Eq. (16)

$$v_i(t+1) = w.v_i(t) + n_1 r_1 (p_i - x_i(t)) + n_2 r_2 (p_{best} - x_i(t)), \tag{16}$$

where w is the local outlier factor for each particle of the velocity, n_1 and n_2 are positive constants called the "cognitive" and "social" parameters, which implement the local outlier factors of two different swarm memories, and r_1 and r_2 are the random numbers between 0 and 1. The proposed procedure does not require the swarm to perform a more global search with a large movement; it requires only a small movement and fine tuning at the end of the optimization process. After calculating the velocity vector, the position of the particles is updated based on the following equation:

$$x_i(t+1) = x_i(t) + v_i(t+1) \tag{17}$$

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.

$$d_j = \left(C_j, R_j^d\right). \tag{18}$$

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

Eq. (18) is a representation of one detector d_j with the center $C_j = \{C_{j1}, C_{j2}, C_{j3}, \cdots, C_{jN}\}$ as the detector center with respect to the numbers of the detectors d_j, while R_j is the detector radius of each detector d_j with respect to the diameter R^d. The Euclidean distance is used as the matching measurement. The distance between the non-spam sample X_i and the detector d_j can be

defined as follows:

$$L(X_i,\ d_j) = \sqrt{(x_{i1} - C_{j1})^2 + \cdots + (x_{iN} - C_{jN})^2} \tag{19}$$

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

$$\ltimes = L(X_i, d_j) - Rs. \tag{20}$$

The detector d_j does not match the non-spam sample X_i if $\ltimes > 0$; 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 follows:

$$R^d,\ j = \min(\ltimes),\ \text{if } \ltimes \leq 0 \tag{21}$$

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

3.3. Computation of the fitness function

The local outlier factor (LOF) was employed to calculate the fitness function in a quest for 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 others in a population of data with respect to a certain measure. This definition is used in a fitness function for the generation of unique features in spam space. This technique will model the data point with the use of a stochastic distribution (Sajesh and Srinivasan, 2011). The point is determined to be an outlier based on its relationship with the model. The outlier detection algorithm proposed as a fitness function in this study of spam detection generation is very unique in computing the full dimensional distance from one point to another (Ramaswamy et al., 2000; Knorr and Ng, 1998) while computing the density of the local neighborhood.

- We assume the k distance (i) to be the distance of the candidate detector or particle (i) to the nearest neighborhood (non-spam).
- The set of k nearest neighbors (non-spam elements) includes all of the particles that are at this distance.
- The set S of k nearest neighbors is denoted as $N_k(i)$.
- This distance defines the reach-ability distance.
- The reach-ability-distance $_k(i,s) = \max\{k-\text{distance}(s), d(i,s)\}$.
- The local reach-ability distance is then defined as follows:

$$\text{lrd}(i) = 1/\left(\frac{\sum s\epsilon N_k(i)\text{reachability} - \text{distance}_k\ (i,s)}{|N_K(i)|}\right) \tag{22}$$

Eq. (22) is the quotient of the average reach-ability distance of the candidate detector i from the non-spam element. This value is not the average reach-ability of the neighbor from i but instead is the distance from which it can be reached from its neighbor. We then compare the local reach-ability density with those of its neighbor using the equation below:

$$\text{LOF}_K(i) = \frac{\sum s\in N_k(i)(\text{lrd}(s))/(\text{lrd}(i))}{|N_k(i)|} = \left(\frac{\sum s\in N_k(i)\text{lrd}\ (s)}{|N_K(i)|}\right)/\text{lrd}(i) \tag{23}$$

Eq. (23) shows the average local reach-ability density of the neighbor divided by the local reach-ability density of the particle. In this scenario, a value of approximately 1 indicates that the particle is comparable with its neighbor (not an outlier). A value of 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 a degree of being an outlier. The degree is the LOF of the particle. The methodology for the computation of the LOF for all of the particles is explained in the following steps:

Step 1: For each particle i, compute the k distance element in non-spam space (the distance of k nearest neighbors in non-spam space s), as shown in Eq. (24)
Step 2: Using Eq. (25) computes the reach-ability distance for particle i in non-spam space as the reach-ability-distance $(i) = \max\{k\text{distance}(s), d(i,s)\}$, where $d(i,s)$ is the distance from particle i to non-spam space s.
Step 3: Compute the local reach-ability density of particle i as the inverse of the average reach-ability distance based on *Minpts* (minimum number of non-spam space) nearest neighbors of particle i in Eq. (26)
Step 4: Using Eq. (27), compute the LOF of particle i as the average of the ratios of the local reach-ability density of the neighbors in non-spam space divided by the number of objects that have the same local reach-ability density.

Let us assume G to be the population of particles, S to be the non-spam space and i to be the ith particle in G.

$$\text{For each particle } i, \text{we have } i \in G.\ \max(k - \text{dist.}(s)) \tag{24}$$

$$/\text{Reach} - \text{dist.G}/^*\ \max(\text{dist}(s,i)) \tag{25}$$

```
Algorithm LOF
Input:   G   //Random particle population
         S   //Non-spam space
         i   // i^th particle in G
Output: The degree of the local outlier factor for all records of the i^th
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) ∈ (G)
[8]   Find population G with maximum reach-ability distance to 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. 2. Algorithm for the fitness function.

$$|G|^*(\text{Minpt}\ (s,\ i)) \tag{26}$$

$$|G|^*(\text{similarity}\ (i, G) \tag{27}$$

Fig. 2 presents the local outlier factor (LOF) algorithm.

3.3.1. Fitness model

The proposed computation uses direct(x) to denote the mean value of direct$_{min}(x)$ and direct$_{max}$ (x). Additionally, indirect (x) is used to denote the mean value of indirect$_{min}(x)$ and indirect$_{max}(x)$.

For any particle, let direct$_{min}(x)$ denote the minimum reachability distance that is between x and a MinPts-nearest neighbor of x.

$$\text{direct}_{min}(x) = \text{Min}\left\{\text{reach} - \frac{\text{distance}(x,y)}{y} \in N_{MinPts}(x)\right\}. \tag{28}$$

In addition, let direct$_{max}(x)$ denote the corresponding minimum;

$$\text{direct}_{max}(x) = \text{Max}\ \{\text{reach} - \text{distance}(x,y)/y \in N_{MinPts}(x)\} \tag{29}$$

To further generalize the definitions of the MinPts – nearest neighbor y *of* x, let indirect$_{min}(x)$ denote the minimum reachability distance between y and a MinPts – nearest neighbor of y.

$$\text{indirect}_{min}(x) = \text{Min}\ \{\text{reach} - \text{dist}(y,\ z) \backslash y \in N_{MinPts}(x). \text{ and } \ z \in N_{MinPts}(y)\} \tag{30}$$

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

Theorem 1. *Let us assume x to be an object from the database D, and* $1 \le \text{MinPts} \le |D|$; *then,*

$$\frac{\text{direct}_{min}(x)}{\text{indirect}_{max}(x)} \le \text{LOF}(x) \le \frac{\text{direct}_{max}(x)}{\text{indirect}_{min}(x)} \tag{31}$$

Proof. First, we have

$$\frac{\text{direct}_{min}(x)}{\text{indirect}_{max}(x)} \le \text{LOF}(x) \tag{32}$$

Then, $\forall z \in N_{MinPts}(x)$, we have $\text{reach} - \text{dist}\ (x,z) \ge \text{direct}_{min}(x)$.

By the direct$_{min}(x)$

$$1/\frac{z \in N \sum_{MinPts(x)} \text{reach} - \text{dist}(x,\ z)}{|N_{MinPts}(x)|} \le \frac{1}{\text{direct}_{min}(x)} \tag{33}$$

and

$$\text{lrd}\ (x) \le \frac{1}{\text{direct}_{min}(x)} \tag{34}$$

For every $y \in N_{MinPts}(z)$, we have

$$\text{reach} - \text{dist}\ (z,\ y) \le \text{indirect}_{max}(x)$$

By indirect$_{max}(x)$

$$1/\frac{y \in N \sum_{MinPts(z)} \text{reach} - \text{dist}(z,\ y)}{|N_{MinPts}(z)|} \ge \frac{1}{\text{indirect}_{max}(x)}. \tag{35}$$

and

$$\text{lrd}(z) \ge \frac{1}{\text{indirect}_{max}(x)} \tag{36}$$

We then have

$$\begin{aligned} \text{LOF}(x) &= \frac{z \in N \sum_{MinPts}(x)(\text{lrd}(z))/(\text{lrd}(x))}{|N_{MinPts}(x)|} \\ &\ge \frac{z \in N \sum_{MinPts(x)}((1/(\text{indirect}_{max}(x)))/(1/(\text{direct}_{min}(x))))}{|N_{MinPts}(x)|} \\ &= \frac{\text{direct}_{min}(x)}{\text{indirect}_{max}(x)} \end{aligned} \tag{37}$$

$$\text{LOF}(x) \le \frac{\text{direct}_{min}(x)}{\text{indirect}_{max}(x)} \tag{38}$$

3.4. Computing the generated detector in the spam space

The proposed dataset for the research is in real values. The real value negative selection algorithm was enhanced by the generation of detectors with PSO. The process of detector generation takes place at the random detector generation phase of the real valued negative selection algorithm for classifying non-spam and spam. In the case of real values, the non-spam and the spam space are as defined in Eq. (1). The candidate detector is generated with PSO and then compared with the non-spam samples. Detectors that do not match any sample of the non-spam set are accepted as viable detectors. Detectors that match the sample of the non-spam set are discarded as unwanted detectors. The generation of detectors continues until the detector set reaches the required coverage in the spam space. The generated detectors can then monitor the status of the system. If a new (test) sample matches at least one of the detectors in the system, it is assumed to be spam (which is abnormal to the system). However, 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 samples in a real valued negative selection algorithm are represented as N-dimensional points, and a non-spam radius Rs is a training dataset. In clearer terms, let Eq. (39) represent the non-spam space:

$$S = \{X_i | i = 1,\ 2, \cdots, m; Rs = r\}\ . \tag{39}$$

Here, X_i denotes some points in the normalized N-dimensional space:

$$X_i = \{x_{i1}, x_{i2},\ x_{i3},\ \cdots,\ x_{iN}\},\ = 1,\ 2,\ 3, \cdots, m \tag{40}$$

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

$$d_j = (C_j, R^d, j) \tag{41}$$

Eq. (41) denotes a detector that is generated with particle swarm optimization, where $C_j = \{C_{j1},\ C_{j2}, C_{j3}, \cdots, C_{jN}\}$ in the detector is the center, and R_j is the detector radius. The Euclidean distance is used as the matching measurement. The distance between the non-spam sample X_i and the detector d_j can be defined as follows:

$$L(X_i, d_j) = \sqrt{(x_{i1} - C_{j1})^2 + \cdots + (x_{iN} - C_{jN})^2} \tag{42}$$

$L(X_i,\ d_j)$ is compared with the non-spam space threshold Rs, which generates the match value of $\ltimes$:

$$\ltimes = L(X_i, d_j - Rs) \tag{43}$$

The detector d_j fails to match the non-spam sample X_i if $\ltimes > 0$. Therefore, if d_j does not match any non-spam sample, it will be retained in the detector set. The detector threshold R^d, j of detector d_j can be defined as

$$R^d, j = \min(\ltimes), if \ltimes \le 0 \tag{44}$$

If detector d_j matches the non-spam sample, it will be discarded. This process will not stop until a detector set that attains the desired spam space coverage is reached. The generated detector set can then be used to monitor the entire system.

4. Empirical study and dataset analysis

The corpus benchmark is obtained from the spam base data set, which is an acquisition from email spam messages (Hopkins et al.,

Table 1
Feature relevance analysis for the spambase dataset.

Attribute number	Attribute type	Attribute description
A1–A48	word_freq_WORD	Percentage of words in the email that match WORD
A49–A54	char_freq_CHAR	percentage of characters in the e-mail that match CHAR
A55	capital_run_length_average	Average length of uninterrupted sequences of capital letters
A56	capital_run_length_longest	Length of longest uninterrupted sequence of capital letters
A57	capital_run_length_total	Total number of capital letters in the e-mail
A58	Class attribute	Denotes whether the email was considered spam (1) or not (0)

1999). From acquiring these email spam messages, the set is composed of 4601 messages; 1813 (39%) of the messages are marked to be spam messages and 2788 (61%) are identified as non-spam. Acquisition of this corpus is already pre-processed, unlike most corpuses, which arrive in their raw form. The instances or features are represented as 58-dimensional vectors. In the corpus of 58 features, 48 of the features of the corpus are represented by words that are generated from the original messages with the absence of a stop-list or stemming, and they are considered and enlisted as the most unbalanced words for the class spam. The remaining 6 features are the percentage of manifestation of the special characters ";", "(", "[", "!", "$" and "#". Some other 3 features are a representation of various measures of the manifestation of capital letters that exist in the text of the messages. Last is the class label in the corpus; this label gives the condition of an instance to be spam or non-spam by a 1 and 0 representation. The spam base dataset is among one of the best test beds that performs well (Koprinska, 2007) during learning and with evaluation techniques. The entire dataset was divided using a stratified sampling approach into a training set and testing set. Of the entire dataset, 70% was used for training, and the remaining 30% was used for testing the model. The analysis of the features is presented in Table 1 below.

4.1. Criteria for performance evaluation

Different measures can be used to evaluate the accuracy and performance of the NSA and NSA–PSO models. To evaluate and compare the performance and accuracy of these models, statistical quality measures used in machine learning and data mining journals were employed. These measures are the sensitivity (SN), specificity (SP), positive prediction value (PPV), accuracy (ACC), negative prediction value (NPV), correlation coefficient (MC) and f-measure (F1). See Biggio et al. (2011) for more detailed mathematical formulae. These measures are discussed briefly, as shown below.

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

$$\mathrm{SN} = \frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FN}} \tag{45}$$

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

$$\mathrm{SP} = \frac{\mathrm{TN}}{\mathrm{TN}+\mathrm{FP}}. \tag{46}$$

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

$$\mathrm{PPV} = \frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FP}} \tag{47}$$

(iv) Negative prediction value (NPV): The NPV of a test also gives the measurement of the percentage of true negative instances out of the overall number of pattern instances that are recognized to be negative. The *NPV* measures the probability that a negatively predicted pattern instance is labeled as negative.

$$\mathrm{NPV} = \frac{\mathrm{TN}}{\mathrm{FN}+\mathrm{TN}} \tag{48}$$

(v) Accuracy (Acc): The Acc measures the percentage of samples that are correctly classified.

$$\mathrm{Acc} = \frac{\mathrm{TP}+\mathrm{TN}}{\mathrm{TP}+\mathrm{TN}+\mathrm{FN}+\mathrm{FP}} \tag{49}$$

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

$$\mathrm{CC} = \frac{[\,(\mathrm{TP})(\mathrm{TN}) - (\mathrm{FP})(\mathrm{FN})]}{\sqrt{(\mathrm{TP}+\mathrm{FN})(\mathrm{TP}+\mathrm{FP})(\mathrm{TN}+\mathrm{FP})(\mathrm{TN}+\mathrm{FN})}} \tag{50}$$

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

$$\boldsymbol{F}1 = 2 \cdot \frac{\text{Positive prdictive value} \times \text{Sensitivity}}{\text{Positive prdictive value} + \text{Sensitivity}} \tag{51}$$

(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 means of the populations differ. This test is a hypothesis test.

$$\boldsymbol{T} = \frac{\overline{\boldsymbol{X}}_{\boldsymbol{I}} - \overline{\boldsymbol{X}}_{2}}{\sqrt{(\boldsymbol{S}_{1^2/\boldsymbol{n}_1} + \boldsymbol{S}_{2^2/\boldsymbol{n}_2})}} \tag{52}$$

In the evaluation equation above, TP is the number of true positives, TN is the number of true negatives, FN is the number of false negatives and FP is the number of false positives.

5. Implementation, results and discussion

The amount of email spam that spreads across the network is a critical problem in today's world. Different means have been devised for the propagation of email spam and network security (Wang et al., 2013). Despite the improvement in technology, the

spammers adapt to new techniques. The proposed algorithm compares the NSA model and the NSA–PSO model. These 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 model. Therefore, the proposed spam detection architecture will be constructed based on the NSA–PSO model. This algorithm can be considered to be 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 occurrences 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, then that email 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 detects spam based on a probability ratio. The importance of the proposed model is that it computes the detection of spam based on existing patterns (Haiyan et al., 2009; Abaei and Selamat, 2014). The block messages are identified against their spamicity rather than their respective messages for a better and more efficient detection of the spam content in an email (Gong and Bhargava, 2013).

The tool represents a client and server connection in an organization. As shown in Fig. 3 above, clients 1 and 2 can communicate inside and outside a network. The sent and received messages between the client and other machines are routed through server 1. Server 1 sent the email and detects spams 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 the NSA–PSO spam detector model (Amayri and Bouguila, 2012), 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. The existing problem in the email spam detection that spammers could manipulate the spam messages that are sent to the system through the 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 detectors in the database of the proposed model can learn and keep records of the previous spam or non-spam email messages.

5.1. Spam detection process

The spam detector process is presented in Fig. 4. The design and developed architecture flow of execution is also presented. The NSA–PSO spam content detector takes input messages and verifies them with the files that are present in the database. The verification is performed message after message to determine the content of spam by the calculation of the different probabilities of spam occurrences. The architectural flow is divided into three different modules: the master server, the client module, and the spam report module.

5.1.1. The master server

The master server is a component that is used to keep track of client and spam details by the network administrator while logged into an application. Its functionality comprises adding clients, generating spam reports, sending data and receiving data. Any number of servers and clients can be added by the master server once the server is selected. The option of adding a client address is based on the internet protocol (IP) address. The IP address is validated as soon as it is specified. All clients have a user id and password assigned. As soon as a client is added, its client name, IP address and corresponding server are listed in the client list. The procedure can add a large number of servers, and many clients can also be added based on the number of servers added. The additional servers are based on the system IP address, which makes record of the number of clients on each server. The number of clients in a server can also result in network traffic, which can be resolved by additional servers. The spam report is based on the email content exchange between the client's records spam, which

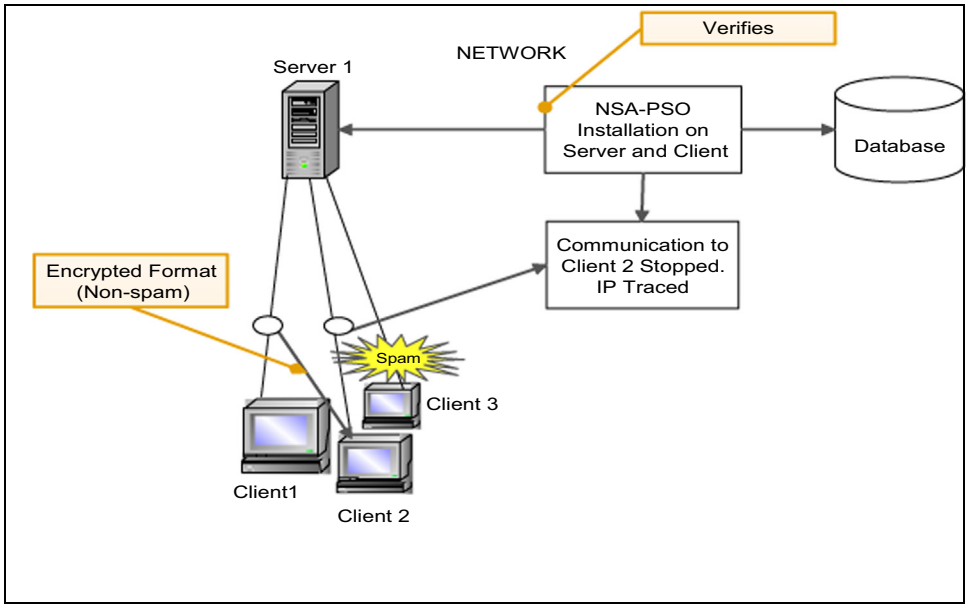

Fig. 3. System architecture

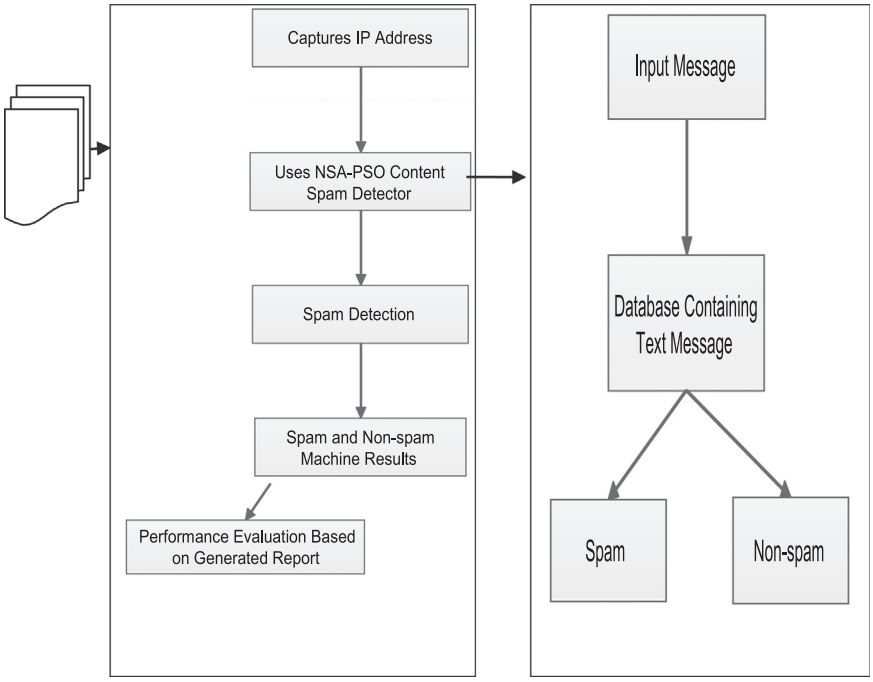

Fig. 4. Spam detection process.

thereby generates spam reports. This approach detects the compromised machine and the non-compromised machine due to the spam that it sent. The master server fields used in the spam report are the IP address, client name, server name and time stamps. A clear picture of the clients that are connected and the corresponding server is given by this field.

5.1.2. The client module

The function of the client is explained in Section 5.1.1. Any number of clients can be handled by a server. The data exchange and communication by the clients is based on sent data and received data. In sent data, there exists a mailbox for each client, whereby data can be sent. The message exchange is in the form of email. The sent data has the option of composing the email, in which the information goes through one client to the other client. This application involves the NSA–PSO spam detector model, and the email has the choice of sending both text and image messages. An email can be sent by a client based on the client's unique ID. Information such as the IP address, client name, date and time stamps are stored in the database once an email is sent. The received data accepts an email message transfer by a client; in this way, it gets into the inbox of another client machine. The client mailbox is similar to a general setup of email that can compose an inbox and send mail. If the content of the email to be delivered is spam, then the detector blocks the message from being delivered to the client machine. If the email message is non-spam, then there is an exchange of email messages between the clients.

5.1.3. Spam report module

The module runs under the server software. The record of spam and non-spam messages is recorded and maintained by the module. The system therefore identifies the compromised and non-compromised machines in the network. Tracks of the machine are recorded based on the server name, dates and client name. The system also has different records of spam machines and non-spam machines and other records that contain the client's name, spam details, date, IP address and time stamps; based on the NSA–PSO model, the spam is detected. The system reports the spam detail as noted while the non-spam reports and the messages are encrypted for the network administrator. This approach helps to take care of privacy among the clients in the network.

5.2. Results

In the evaluation of the NSA model and its improved models, they were 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 numbers of detectors generated have a tremendous impact on the final output measure. The comparison between the standard model and the proposed improved models using the validation of unseen data is summarized in Figs. 5–7 below. The performance of the improved NSA–PSO model outperforms the NSA model. The proposed model shows an improved accuracy when compared with the standard model, which performs poorly on all of the measurement standards.

Fig. 5 shows the accuracy of the negative selection algorithm (NSA) and negative selection algorithm-particle swarm optimization (NSA–PSO). The proposed improved model performs better than the NSA model, with the average accuracy of the proposed NSA–PSO model at 70.48% and the average accuracy of the NSA model at 65.15%. The accuracy with 5000 generated detectors and with a threshold value of 0.4 for the NSA–PSO model is 83.201% and for the NSA model is 68.863%.

Fig. 6 shows the *F*-measure for the negative selection algorithm (NSA) and negative selection algorithm-particle swarm optimization (NSA–PSO). The proposed improved model performs better than the NSA model, with the average f-measure of the proposed NSA–PSO model at 43.546% and the average f-measure of the NSA model at 22.09%.

For the *F*-measure with 5000 generated detectors and with a threshold value of 0.4, the NSA–PSO model obtains 76.85% and the NSA model obtains 36.01%. From the results obtained, it can be noted that the improved NSA–PSO model performed better in all aspects. This result proves the consistency of the quality of the measurements that were used, in every respect.

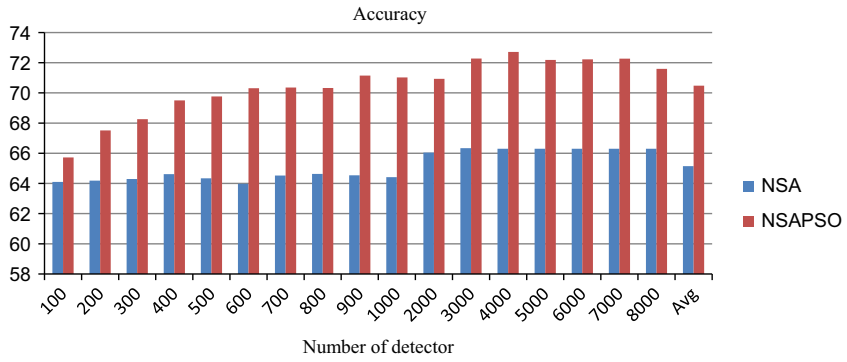

Fig. 5. Accuracy of the NSA–PSO and NSA models.

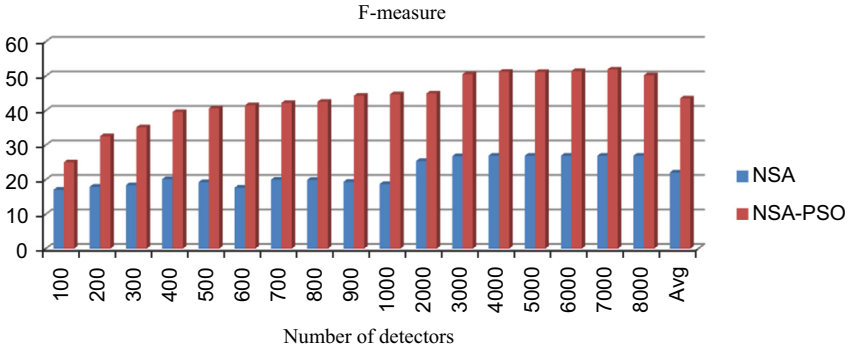

Fig. 6. F-measure of the NSA–PSO and NSA models.

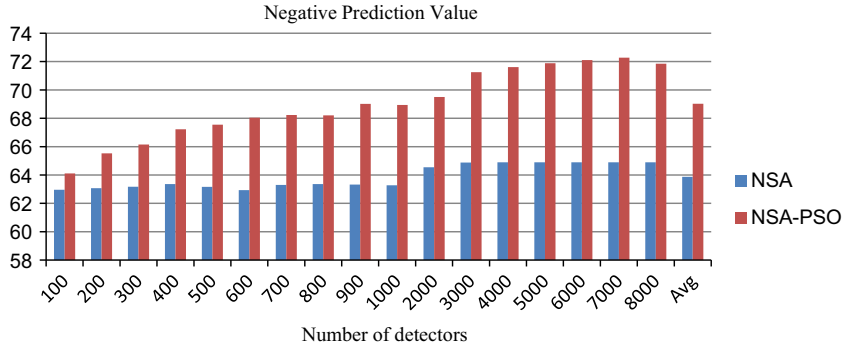

Fig. 7. Negative prediction value for the NSA–PSO and NSA models.

Fig. 7 shows the negative prediction value of the negative selection algorithm (NSA) and negative selection algorithm-particle swarm optimization (NSA–PSO). The proposed improved models perform better than the NSA model, with an average negative predictive value for the proposed improved NSA–PSO model at 69.03% and for the NSA model at 63.87%. For the negative prediction value at 5000 generated detectors with a threshold value of 0.4, the NSA–PSO model obtained 82.77% and the NSA model obtained 66.24%. The NSA model performs very low when compared with the improved models. The improvement is on a very large scale and shows the relevance of particle swarm optimization in improving the detector generation phase of the negative selection algorithm. This approach in practice solves the problem of detector generation and reduces the false rate because more reliable features are generated, which shows that the standard model is a robust and more effective model.

5.3. Statistical t-test

The *p*-value (probability) was used to determine if the population means differ or not. The *t*-test examines the *t*-statistic, *t*-distribution and degrees of freedom to establish this fact. The analysis presented in Table 2 below indicates that there is a high correlation between the means of NSA and NSA–PSO at the 0.05 alpha level. This finding shows that there is a mutual unity between NSA and NSA–PSO among the variables. The correlation between NSA and NSA–PSO is the relationship that exists between the two algorithms. This relationship is corroborated by means of both the NSA and NSA–PSO ranging between 65.1477 and 70.4763 in accuracy; additionally, the standard deviation indicated that there is a deviation between 0.98 and 1.89. Other evaluation measure analysis is represented in Table 3.

There is a significant correlation between the mean of the negative selection algorithm and the negative selection algorithm-

Table 2
At 2000 generated detectors with a threshold value of 0.4, Table 2 shows a summary and comparison of the results in terms of the percentages for the NSA and NSA–PSO models.

Model	ACC	CC	F1	SN	PPV	SP	NPV
NSA	68.86	36.06	36.01	22.24	94.53	99.16	66.24
NSA–PSO	82.62	63.37	74.95	65.99	86.71	93.42	80.87

Note: ACC = accuracy, CC = correlation coefficient, *F*1 = *F*measure, SN = sensitivity, PPV = positive prediction value, SP = specificity and NPV = negative prediction value.

Table 3
The *t*-test for the negative selection algorithm and negative selection algorithm-particle swarm optimization.

Measure	Algorithm	Df($n-1$)	SD	Sig (2-tailed)	Comment
ACC	NSA	16	0.9840	0.000	Higher correlation
	NSA–PSO	16	1.8960	0.000	
F1	NSA	16	4.0694	0.000	Higher correlation
	NSA–PSO	16	7.5510	0.000	
PPV	NSA	16	1.5308	0.000	Higher correlation
	NSA–PSO	16	4.5497	0.000	
CC	NSA	16	2.3800	0.000	Higher correlation
	NSA–PSO	16	3.7216	0.000	
SN	NSA	16	3.2748	0.000	Higher correlation
	NSA–PSO	16	8.6248	0.000	
SP	NSA	16	0.5221	0.000	Higher correlation
	NSA–PSO	16	2.7648	0.000	
NPV	NSA	16	0.8496	0.000	Higher correlation
	NSA–PSO	16	2.5052	0.000	

Note: C = accuracy, CC = correlation coefficient, *F*1 = *F* measure, SN = sensitivity, PPV = positive prediction value, SP = specificity and NPV = negative prediction value.

particle swarm optimization. There is also a high level of accuracy in both of them.

5.4. Result comparison of NSA, NSA–PSO and other schemes

The result obtained from the proposed NSA–PSO model is compared with the NSA model and other standard machine learning algorithms in this research. The enhanced models will be compared against the support vector machine (SVM) proposed in (Fagbola et al., 2012), the distinguishing feature selection and support vector machine (DFS–SVM) proposed in (Uysal and Gunal, 2012) and the naïve Bayes (NB) proposed in (Zhang et al., 2008). These standard machine learning tools are used for comparison with our proposed model. The proposed model shows a high accuracy in detecting email spam. Table 4 shows in summary the results analysis of all of the models. The discussion of the results on the individual models will be presented shortly.

However, the differences in the performances between the proposed NSA–PSO model and the NSA model are very significant. The best accuracy of the proposed model is 83.20%, while for the NSA model, it is 68.86%. In general, the proposed model outperforms the standard NSA model. The comparison of the proposed model with state-of-the-art machine learning models shows that our model performs better than all of the models that were listed in Table 4 above. The proposed model outperforms the standard naïve Bayes models that were proposed in Zhang et al. (2008) and Abu-Nimeh et al. (2008), with the accuracy of the two models being 78.8% and 79.3%, respectively. The model also performs better than the support vector machine (SVM) proposed in Fagbola et al. (2012), which has an accuracy of 90%, and the distinguishing feature selection and support vector machine (DFS–SVM) proposed in Uysal and Gunal (2012), which has an accuracy of 71%. This finding shows that the accuracy of the proposed model is better than the existing models that were proposed by other authors using state-of-the-art machine learning tools.

Table 4
Testing results comparison for NSA, PSO, NSA–PSO and other schemes.

Classifier	Accuracy
SVM	90%
Fagbola et al. (2012)	
NB	78.8%
Zhang et al. (2008)	
DFS–SVM	71%
Uysal and Gunal (2012)	
NSA	68.86%
Forest et al. (1994)	
NSA–PSO	83.20%
Proposed model	

6. Conclusions and recommendations

A new and improved model that combines the negative selection algorithm with particle swarm optimization (PSO) was proposed and implemented. The uniqueness of this model is that PSO was implemented at the random detector generation phase of NSA. The detector generation phase of NSA determines how robust and effective the algorithm will perform. PSO implementation with the local outlier factor (LOF) as a fitness function no doubt improved the detector generation phase of the NSA. The proposed improved model serves as a better replacement to the NSA model. This performance investigation has shown that the proposed improved model can detect email spam better than the NSA model. In total, the empirical report shows the superiority of the proposed NSA–PSO over the NSA model. The improved NSA–PSO model was then used to design and develop an architectural system of a client-and-server network. Additionally, the media flow of the NSA–PSO spam detector model, which receives input from email messages, was also presented. The model provides sensitivity to the client and can adapt very well to changes in spam techniques in the future by noting the spam content in a network despite modifications to the spam messages. It is suggested that this research be considered a viable tool for any newly proposed system in the email spam detection problem that is based on detector generation and network implementation. Future work will implement a parallel hybridization of two evolutionary algorithms to perform the single task of detecting email spam in a network.

Acknowledgments

The Universiti Teknologi Malaysia (UTM) and Ministry of Education Malaysia under research Grant 01G72 and Ministry of Science, Technology & Innovations Malaysia, under research Grant 4S062 are hereby acknowledged for some of the facilities that were utilized during the course of this research work. This paper has been elaborated in the framework of the project "Support research and development in the Moravian-Silesian Region 2013 DT 1 - International research teams" (RRC/05/2013), financed from the budget of the Moravian-Silesian Region. A portion of the work is also supported from the FIM excellence project.

References

Almeida, T.A., Yamakami, A., 2012. Facing the spammers: a very effective approach to avoid junk e-mails. Expert Syst. Appl. 39 (7), 6557–6561.

Abi-Haidar, A., Rocha, L., 2008. Adaptive spam detection inspired by a cross-regulation model of immune dynamics: a study of concept drift. In: Bentley, P., Lee, S., Jung, S. (Eds.), Artificial Immune Systems. Springer, Berlin Heidelberg, pp. 36–47.

Abaei, G., Selamat, A., 2014. A survey on software fault detection based on different prediction approaches. Vietnam J. Comput. Sci. 1 (2), 2014. http://dx.doi.org/10.1007/s40595-013-0008-z.

Amayri, O., Bouguila N., 2012. Content-based spam filtering using hybrid generative discriminative learning of both textual and visual features. In: Proceedings of the 2012 IEEE International Symposium on Circuits and Systems (ISCAS).

Abu-Nimeh, S., et al., 2008. Bayesian Additive Regression Trees-Based Spam Detection for Enhanced Email Privacy. In: Proceedings of the Third International Conference on Availability, Reliability and Security, ARES'08.

Bambauer, D.E., 2005. Solving the inbox paradox: an information-based policy approach to unsolicited E-mail advertising. Va J. Law Technol. 10 (5), 1–94.

Bereta, M., Burczyński, T., 2007. Comparing binary and real-valued coding in hybrid immune algorithm for feature selection and classification of ECG signals. Eng. Appl. Artif. Intell. 20 (5), 571–585.

Bezerra, G., et al., 2006. An immunological filter for spam. In: Bersini, H., Carneiro, J. (Eds.), Artificial Immune Systems. Springer, Berlin, Heidelberg, pp. 446–458.

Biggio, B., et al., 2011. A survey and experimental evaluation of image spam filtering techniques. Pattern Recogn. Lett. 32 (10), 1436–1446.

Cormack, G., T., Lynam, 2007. TREC Public Spam Corpus. ⟨http://plg.uwaterloo.ca/~gvcormac/treccorpus07⟩ [cited 15 January 2009].

Delany, S.J., Buckley, M., Greene, D., 2012. SMS spam filtering: methods and data. Expert Syst. Appl. 39 (10), 9899–9908.

De Castro, L.N., Timmis, J., 2002. Artificial immune systems, A New Computational Approach. Springer verlag, London, UK p. 357.

Dasgupta, D., 2006. Advances in artificial immune systems. Comput. Intell. Mag. IEEE 1 (4), 40–49.

Dasgupta, D., Yu, S., Nino, F., 2011. Recent advances in artificial immune systems: models and applications. Appl. Soft Comput. 11 (2), 1574–1587.

Fagbola, T., Olabiyisi, S., Adigun, A., 2012. Hybrid GA–SVM for efficient feature selection in E-mail classification. Comput. Eng. Intell. Syst. 3 (3), 17–28.

Forrest, S., Perelson, A.S., Allen, L., Cherukuri, R., 1994. Self-nonself discrimination in a computer. In: Proceedings of the IEEE Computer Society Symposium on Research in Security and Privacy. 16–18 pp. 202–212.

Guzella, T.S., Caminhas, W.M., 2009. A review of machine learning approaches to spam filtering. Expert Syst. Appl. 36 (7), 10206–10222.

Gao, X.Z., S.J. Ovaska, X. Wang, 2007. Particle Swarm Optimization of detectors in Negative Selection Algorithm. In: Proceedings of the IEEE International Conference on Systems, Man and Cybernetics, ISIC'07.

Guangchen, R., T. Ying, 2007. Intelligent Detection Approaches for Spam. In: Proceedings of the Third International Conference on Natural Computation, ICNC'07.

Gong, T., Bhargava, B., 2013. Immunizing mobile ad hoc networks against collaborative attacks using cooperative immune model. Secur. Commun. Netw. 6 (1), 58–68.

Hamdan, Mohammad Adel, Abu., Z.R., 2011. Application of genetic optimized artificial immune system and neural networks in spam detection. Appl. Soft Comput. 11 (4), 3827–3845.

Hopkins, M., et al., 1999. Spam Base Dataset. Hewlett-Packard Labs.

Haiyan, W., Z. Runsheng, W. Yi, 2009. An anti-spam filtering system based on the naive bayesian classifier and distributed checksum clearinghouse. In: Proceedings of the Third International Symposium on Intelligent Information Technology Application, IITA'09.

Knorr, E.M., R.T. Ng, 1998. 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., pp. 392–403.

Koprinska, I., 2007. Learning to classify e-mail. Inf. Sci.: Int. J. Arch., 177.

Man, Q., R. Mousoli, 2010. Semantic analysis for spam filtering. In: Proceedings of the Seventh International Conference on Fuzzy Systems and Knowledge Discovery (FSKD).

Massey, B., et al., 2003. Learning spam: simple techniques for freely-available software. In: Proceedings of the annual conference on USENIX Annual Technical Conference, USENIX Association: San Antonio, Texas, pp. 13.

Mazhoud, I., et al., 2013. Particle swarm optimization for solving engineering problems: a new constraint-handling mechanism. Eng. Appl. Artif. Intell. 26 (4), 1263–1273.

Oda, T., White, T., 2003a. In: Cantú-Paz, E., et al. (Eds.), Developing an Immunity to Spam, in Genetic and Evolutionary Computation – GECCO 2003. Springer, Berlin, Heidelberg, pp. 231–242.

Oda, T., T. White, 2003b. Increasing the accuracy of a spam-detecting artificial immune system. In: Proceedings of the 2003 Congress on Evolutionary Computation, CEC'03.

Oda, T., White, T., 2005. Immunity from spam: an analysis of an artificial immune system for junk email detection. In: Jacob, C., et al. (Eds.), Artificial Immune Systems. Springer, Berlin, Heidelberg, pp. 276–289.

Pérez-Díaz, N., et al., 2012. SDAI: An integral evaluation methodology for content-based spam filtering models. Expert Syst. Appl. 39 (16), 12487–12500.

Ramaswamy, S., Rastogi, R., Shim, K., 2000. Efficient algorithms for mining outliers from large data sets. SIGMOD Rec. 29 (2), 427–438.

Sheikhan, M., Sharifi Rad, M., 2013. Using particle swarm optimization in fuzzy association rules-based feature selection and fuzzy ARTMAP-based attack recognition. Secur. Commun. Netw. 6 (7), 797–811.

Sirisanyalak, B., Sornil, O., 2007. An artificial immunity-based spam detection system. In: Proceedings of the IEEE Congress on Evolutionary Computation, CEC'07.

Sajesh, T.A., Srinivasan, M.R., 2011. Outlier detection for high dimensional data using the Comedian approach. J. Stat. Comput. Simul. 82 (5), 745–757.

Thonnard, O., Vervier, P.-A., Dacier, M., 2012. Spammers operations: a multifaceted strategic analysis. Secur. Commun. Netw, http://dx.doi.org/10.1002/sec.640.

Tenne, Y., 2012. A computational intelligence algorithm for expensive engineering optimization problems. Eng. Appl. Artif. Intell. 25 (5), 1009–1021.

Uysal, A.K., Gunal, S., 2012. A novel probabilistic feature selection method for text classification. Knowl.-Based Syst. 36 (0), 226–235.

Wamli, M., T. Dat, S. Dharmendra, 2009. A novel spam email detection system based on negative selection. In: Proceedings of the Fourth International Conference on Computer Science and Convergence Information Technology.

Wang, H., et al., 2009. PSO-optimized negative selection algorithm for anomaly detection, Applications of Soft Computing. Springer, Berlin,Heidelberg, pp. 13–21.

Wenqing, Z., Z. Zili., 2005. An email classification model based on rough set theory. In: Proceedings of the 2005 International Conference on Active Media Technology(AMT'05).

Wang, Y., et al., 2013. Modeling and security analysis of enterprise network using attack–defense stochastic game Petri nets. Secur. Commun. Netw. 6 (1), 89–99.

Yildiz, A.R., 2013. Comparison of evolutionary-based optimization algorithms for structural design optimization. Eng. Appl. Artif. Intell. 26 (1), 327–333.

Yevseyeva, I., et al., 2013. Optimising anti-spam filters with evolutionary algorithms. Expert Syst. Appl. 40 (10), 4010–4021.

Zhang, L., Zhu, J., Yao, T., 2004. An evaluation of statistical spam filtering techniques. ACM Trans. Asian Lang. Inf. Process. (TALIP) 3 (4), 243–269.

Zhang, Y., et al., 2008. Applying cost-sensitive multiobjective genetic programming to feature extraction for spam E-mail filtering. In: O'Neill, M., et al. (Eds.), Genetic Programming. Springer, Berlin, Heidelberg, pp. 325–336.