Engineering Applications of Artificial Intelligence 28 (2014) 97–110

Contents lists available at ScienceDirect

Engineering Applications of Artificial Intelligence

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

Hybrid email spam detection model with negative selection algorithm and differential evolution

Ismaila Idris [a], Ali Selamat [a,*], Sigeru Omatu [b]

[a] Software Engineering Research Group (SERG), Knowledge Economy Research Alliance and Faculty of Computing, Universiti Teknologi Malaysia, 81310 UTM Johor Bahru, Johor, Malaysia
[b] Department of Electronics, Information and Communication Engineering, 5-16-1 Omiya, Asahiku, Osaka 535-8585, Japan

ARTICLE INFO

Article history:
Received 9 April 2013
Received in revised form
16 August 2013
Accepted 2 December 2013
Available online 27 December 2013

Keywords:
Negative selection algorithm
Differential evolution
email
Spam
Non-spam
Detector generation

ABSTRACT

Email spam is an increasing problem that not only affects normal users of internet but also causes a major problem for companies and organizations. Earlier techniques have been impaired by the adaptive nature of unsolicited email spam. Inspired by adaptive algorithm, this paper introduces a modified machine learning technique of the human immune system called negative selection algorithm (NSA). A local selection differential evolution (DE) generates detectors at the random detector generation phase of NSA; code named NSA–DE. Local outlier factor (LOF) is implemented as fitness function to maximize the distance of generated spam detectors from the non-spam space. The problem of overlapping detectors is also solved by calculating the minimum and maximum distance of two overlapped detectors in the spam space. From the experiments, the results show that the detection accuracy of NSA–DE is 83.06% while the standard negative selection algorithm is 68.86% at 7000 generated detectors.

1. Introduction

The cheapest and most important form of communication in the world today is email. It is effective, simple and available for all computer users. The simplicity of email makes it vulnerable to a lot of threats. One of the most important threat to email is spam; virtually all email users across the world suffer from email spam (Cormack et al., 2011). The word spam was used to describe unwanted, junk mails sent to an internet user's inbox. It is very convenient for spammers to send millions of email spam all over the world with no cost at all (Carpinter and Hunt, 2006). This makes it a common scenario for all internet users to receive junk mail hundred times daily. Different techniques have been adopted to stop the threat of spam or drastically reduce the amount of spam that attacks internet users across the world. An anti-spam law was enacted by legislating penalty for spammers that distribute email spam (Schryen, 2007). Also, two general approaches have been used in email spam detection; a knowledge engineering approach and a machine learning approach (Wamli et al., 2009). In the knowledge engineering approach, the use of network information and internet protocol address techniques to determine if a message is spam or non-spam is called origin-based filter. Sets of rules have to be specified in the knowledge engineering approach in order to determine which email is to be categorized as spam or non-spam. Such rules could be created by the use of filter or by some other authority. An example of this process is the software company that provides a particular rule based spam filtering tools. By the application of this method, there is promising result. However, the rules need to be maintained all the time and updated which is a waste of time and inconvenient for most users. Machine learning is more efficient than knowledge engineering approach (Guzella and Caminhas, 2009) and does not require specifying rules; a set of pre-classified email message (training sample) is applied. Specific algorithms are used to learn the classification rules from the email messages. The filtering techniques are the most commonly used methods; it identifies whether a message is spam or non-spam based solely on the message content and some other characteristics of the message. Despite different approaches and techniques adopted to fight the scourge called spam, the internet today still witnesses huge amount of spam (Zhang et al., 2004; Massey et al., 2003), and more attention is needed by adaptive techniques on how the menace can be drastically reduced if not totally eliminated.

Due to the wide knowledge of machine learning approach, several algorithms have been used for email spam detection (Guzella and Caminhas, 2009). They include artificial immune system (AIS), support vector machine (SVM), neural network

* Corresponding author.
E-mail addresses: ismi_idris@yahoo.co.uk (I. Idris), aselamat@utm.my (A. Selamat).

0952-1976/$ - see front matter © 2013 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.engappai.2013.12.001

(NN), Naïve Bayes (NB), *k*-nearest neighbour (KNN), etc. In this paper, we propose a new approach that is inspired by artificial immune system model; that is a negative selection algorithm (NSA) with the combined effort of differential evolution (DE) which modifies the standard negative selection algorithm in order to generate more accurate results. The engineering goals required in hybrid negative selection algorithm can be viewed in three ways; first, is to generate an efficient detector set; secondly, is to limit the number of detectors that will be generated and thirdly, is to maximize the detector set distance as much as possible. Problems that require attention in this research work are: (i) generating detectors in the spam space; (ii) maximizing distance between spam detectors and the non-spam space and (iii) solving the problem of overlapping detectors in the spam space. These problems are solved by the implementation of local differential evolution for generating detectors, application of local outlier factor as fitness function to maximize the distance between generated detector in the spam space and the non-spam space, calculating the minimum and maximum distance between two overlapped generated detectors as fitness function. The performance of NSA is determined by detector generation and how effective it is able to utilize the detector coverage space of spam and non-spam. This paper is organized into 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 presented in Section 3. Empirical studies, results and discussions are presented in Section 4, Section 5 discusses the experimental results while conclusions and recommendations are presented in Section 6.

2. Related work

Over the past years, rapid expansion of computer network systems has changed the world. The expansion is essential for an effective computer security system because attacks and criminal intend are increasingly popular in computer network (Golovko et al., 2010). Negative selection algorithm, while not reacting to the self cells uses the immune system capability to detect unknown antigens. Its mechanism protects body against—reactive lymphocytes. Receptors are made through a pseudo-random genetic rearrangement process during the generation of T-cells (Wang and Zhao, 2008); they then undergo a censoring process in the thymus called the negative selection. In this process T-cells that do not bind to self-proteins are destroyed. Therefore, immunological function and protection of the body against foreign antigens is possible through circulation of matured T-cells (Zhang et al., 2010). Recent work uses immunological function to solve complex problems in negative selection algorithm. The work of Gong et al. (2012) introduced a further training strategy to generate more self-detectors to be able to cover the self-space for effective detectors. The technique reduces the false rate, as wrongly classified non-self will be re-classified for correctness. The drawback of the techniques was that it leads to scalability and part of the self-detector may be covered by overlapped detectors. An immune local concentration based detection approach was proposed by Wie et al. (2011), two element local concentration as a feature vector was combined with negative selection algorithm and optimized with genetic algorithm. The technique generates effective and efficient detectors as the local concentration feature vectors are optimized before training the features. The technique is computationally expensive and also time consuming in achieving desired feature detectors. A similar work was presented by Yildiz (2009, 2013a) and Prakash et al. (2008) to implement optimized immunological functions in solving complex problems in industry. This technique uses evolutionary algorithm to implement parameters optimization in the immune system. A detection model based on penalty factor was proposed by Zhang et al. (2010) to construct a model for spam detection; by redefining the harmfulness of self and non-self using the negative selection algorithm penalty factor to divide the candidate signature library into two detection signature libraries as a self-detector, and then splitting of the programs in an orderly way into various short bit strings. The work of Xin et al. (2010) and Yuebing et al. (2010) proposed a self-detector by the use of real valued negative selection algorithm. A variable size *r*-contiguous matching rule was implemented and the value of the variable size *r* was used to balance between more generalization and specification of the self-space. This technique is not very sufficient in generating self-detectors as it has a constant threshold value which may lead to over-fitting problems in most cases. A shape space as an occupancy of negative selection algorithm was proposed by Wanli et al. (2010). The work states the importance of full coverage of the shape space for effective detectors by suggesting a heuristic for detection generation which was demonstrated by antigen feedback mechanism. The issue of overlapping and scalability was not addressed by Wanli et al. (2010); it will definitely have effect on the shape space as effective detectors generated are unable to sufficiently cover the shape space. The work of Forrest and Perelson (1994) quantifies the number of resources that will be required by NSA in order to exhibit a very good detector capability rate and failure rate. Forrest and Perelson (1994) use a single global affinity threshold value *r* which ranges between a specific number with *r*-contiguous bits matching rule for each and every instance within its population. The affinity threshold in this case is determined through a trial and error method, where the best threshold with the best performance is targeted as the affinity threshold.

The understanding of artificial immune system (AIS) based on the mammalian immune system is vital for this study. A comprehensive artificial immune system survey was analysed by Dasgupta et al. (2011). The research discusses the history, recent development and the four major AIS algorithms. 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 (NSA), 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. Artificial immune system (AIS) is a new mechanism implemented in the control of email spam. Pattern matching was used to represent detectors as regular expression by Oda and White (2003) in the analysis of message. A weight is assigned to the detector which is decremented or incremented when observing the expression in spam message with the classification of the message based on threshold sum of the weight of matching detectors. The system is meant to be corrected by either increasing or decreasing of all matching detector weights with 1000 detectors generated from spam-assassin heuristic and personal corpus. The results were acceptable based on small number of detectors that was used. A comparison of two techniques to determine message classification using spam-assassin corpus with 100 detectors was proposed by Oda and White (2003). This approach is like the previous techniques but the difference is the increment of weight where there is recognition of pattern in spam messages. Random generation of detectors does not help in solving the problem of best selected features; though, feature weights are updated during the matching process. The weighting of features complicates the performance of the matching process. More experiments are performed by Oda and White (2005) with the use of spam-assassin corpus and Bayesian combination of detector weights. Messages are scored by 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 beside 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 other two techniques. A better balance seems to be provided by the weighted average. The immune system classifies correctly 90% of the messages. In specific terms, it classifies 84% of spam and 98% of 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 research 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, detector maturing module, the antigen matching module and the detector ageing module with selves' repository and detectors repository. After the initial 1/3 of the time during the learning period, the spam detection rate is over 80% and over 70% most of the time; the research was implemented with TREC07 corpus (Cormack and Lynam, 2007). A new solution to solve spam detection problem inspired by the adaptive immune system model called cross-regulation model was presented by Abi-Haidar and Rocha (2008). The research shows the relevance of cross-regulation model as a biological inspired algorithm in the detection of spam. Enron corpus was used in its implementation with 70% spam experiment. The accuracy and *F*-measure is at 83% and 79% respectively. The analysis of major work done in negative selection algorithms with the combination of two different algorithms in email spam hybrid model is the work of Sirisanyalak and Sornil (2007). An AIS based module which extracts features was designed and further used for logistic regression model; the set of detectors was initially generated with the use of terms that are extracted from the training message, and also data on matched detectors used in regression model. Spam-assassin (spamAssassin) was used for the experimental work. A genetic algorithm optimized AIS to cull old lymphocytes (replacing the old lymphocyte with new ones) and also check for new interest for users in a way that is similar was presented by Hamdan and Abu (2011) 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 optimized spam detector with 600 generated detectors gives a false positive rate of 1.117% and a false negative rate of 3.741% while spam detection with AIS and 600 generated detectors gives a false positive rate of 1.214% and a false negative rate of 4.906%. A proposed anti-spam filter with evolutionary algorithm was also presented by Yevseyeva et al. (2013). Scores of anti-spam filters are optimized to improve its accuracy. The optimization problem is considered as single and multi-objective problem formulation. Rough set theory which is a mathematical approach for approximate reasoning in other to group messages in three classes was proposed by Wenqing and Zili (2005), targeting low false positive. The selection of features; spam, non-spam or suspicious elements was first implemented on the training set after which genetic algorithm was implemented. The universe of message was divided into three regions based on some 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 in order 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 with same feature selection in Bezerra et al. 2006). The support vector acquired after training SVM was implemented in the generation of initial detector set of AIS and then AIS was used in classification. During classification with the AIS, the detector with smallest Euclidean distance to the message was added to committee set with major voting of detectors in the set as classification. PU1 corpora and Ling-spam corpora (csmining) were used for the experiment.

Other literatures on spam filtering is 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 by Pérez-Diaz 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 algorithm for spam filtering. An intelligent hybrid spam filtering framework (IHSFF) was proposed by Hu et al. (2010) to detect spam by the analysis of email headers only. The framework is suitable mainly for very big email servers due to its efficiency and scalability. No one ever combined negative selection algorithm with differential evolution; though, there are several combined or optimized techniques (Khilwani et al., 2008; Yildiz, 2013b,c,d) implemented with differential evolution and other evolutionary algorithm that solve complex problems.

3. The proposed improved model and its constituent frameworks

Hybrid systems in recent times have extensive success in many real world complex problem solving. The importance of a hybrid system is not negotiable, based on the fact that an individual system has its weakness, and a hybrid system is meant to compliment the weaknesses of these individual intelligent systems. A smart hybridization of negative selection algorithm and differential evolution 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 (Selamat et al., 2012) to achieve stability, consistency and an accurate intelligent system extendable for usage in classification.

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 artificial immune systems (Balthrop et al., 2002). The standard algorithm was proposed by Forrest and Perelson (1994). The algorithm comprises of the data representation, the training phase, and the testing phase. In the data representation phase, data are represented in a binary or a real valued form. The training phase of the algorithm or the detector generation phase randomly generates detectors with binary or real valued data which is used consequently to train the algorithm (Wang and Zhao, 2008); while the testing phase evaluates the trained algorithm. The random generation of detectors by a negative selection algorithm makes it impossible to analyse the type of data needed for the training algorithm. Figs. 1 and 2 show the training and testing phase of NSA.

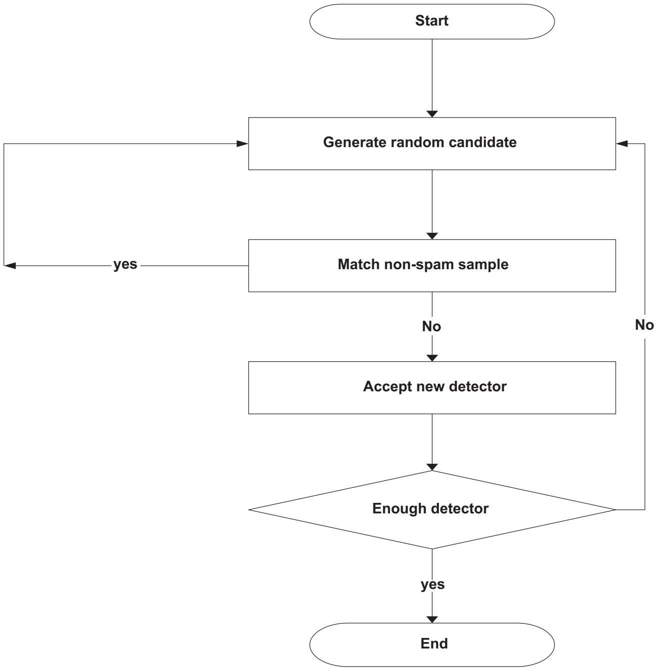


Fig. 1. Detector generation of negative selection algorithm.

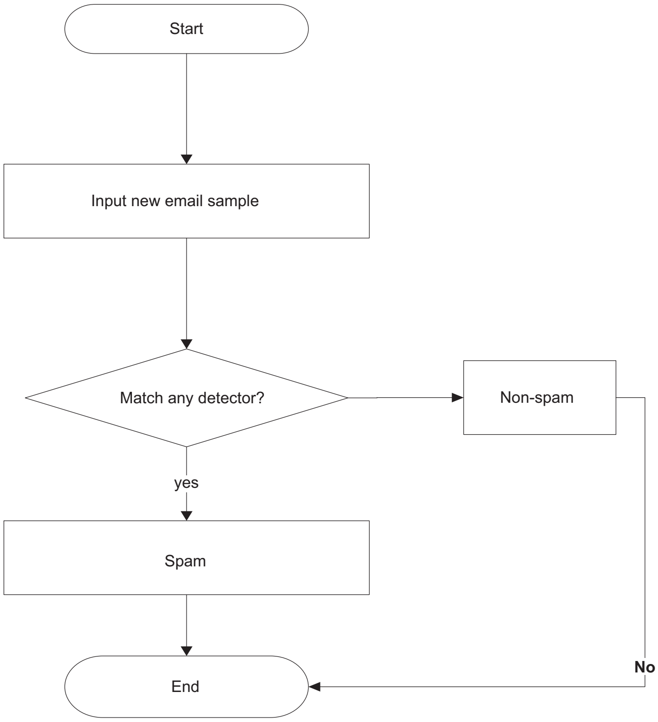
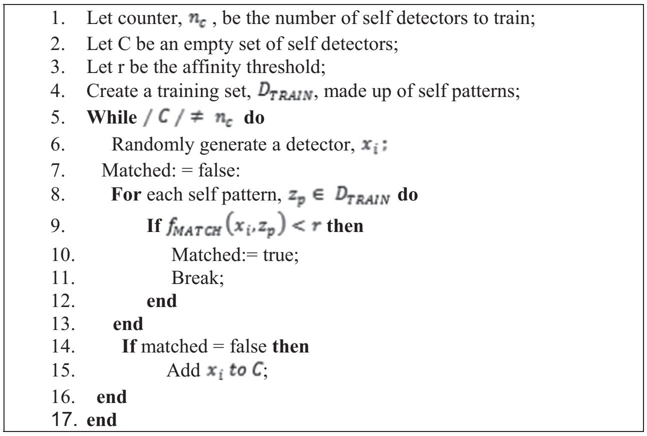


Fig. 2. Testing of negative selection algorithm.

The main concept of the NSA as developed by Forrest and Perelson (1994) was meant to generate a set of candidate detectors, C, such that $\forall x_i \in C$ and $\forall z_p \in S, f_{MATCH}(x_i, z_p) < r$, where x_i is a detector, z_p is a pattern and $f_{MATCH}(x_i, z_p)$ is the affinity matching function. The algorithm of negative selection algorithm as given by Forrest and Perelson (1994) is presented in Fig. 3.

The original NSA uses a binary r-contiguous bit (RCBITS) rule in conjunction with a global affinity threshold, r for each detector in a population of detectors, C. The determination of the affinity threshold is by trial and error (Forrest and Perelson, 1994) 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 (Balthrop et al., 2002; Gonzalez and Gomez, 2003).



```
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 / ≠ $n_c$ 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.

3.1.1. Implementation of negative selection algorithm

The proposed dataset for the research is in real value form (Prabhakar and Basavaraju, 2010). The real value negative selection algorithm is encoded in real valued form for classifying non-spam and spam. In the case of real value, there is need to define the non-spam and the spam space (Forrest and Perelson, 1994). The non-spam space is the normal state of a system while the spam space is the abnormal state of a system. The candidate detector is 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 samples of the non-spam set are discarded as unwanted detectors. The generation of detectors continues until the detector set reaches 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 samples in a real value negative selection algorithm are represented in N-dimensional points and a non-spam radius Rs, as training dataset. In a clearer term, Eq. (1) represents the non-spam space.

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

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

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

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

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

Eq. (3) is a representation of one detector d_j with centre $C_j = \{C_{j1}, C_{j2}, C_{j3} \cdots C_{jN}\}$ 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

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

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

$$\ltimes = L(X_i,\ d_j) - Rs \tag{5}$$

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

$$R^d,\ j = \min(\ltimes) \quad \text{if} \ \ltimes \le 0 \tag{6}$$

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

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 detectors by the real value negative selection algorithm was improved with the introduction of differential evolution (DE) and the local outlier factor (LOF) (Sajesh and Srinivasan, 2011) as fitness function. These are as a result of the quest for efficiently trained negative selection algorithm model for purely normal detectors. The local outlier factor maximized the distance between the generated spam detector and non-spam space. The approach will model the data point by the implementation of stochastic distribution (Sajesh and Srinivasan, 2011) using local outlier factor. Another vital issue in spam space is the overlapping of generated detectors. A fitness function that calculates the distance between two overlapped detectors in the spam space was proposed in this research. 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 sections below 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 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.

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

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

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

S is normalized as follows:

$$S_i = \frac{S_i}{||S_i||} \tag{8}$$

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

3.2.2. *Generation of candidate detectors with differential evolution*

Detector generation was implemented with differential evolution instead of the traditional random generation of detectors.

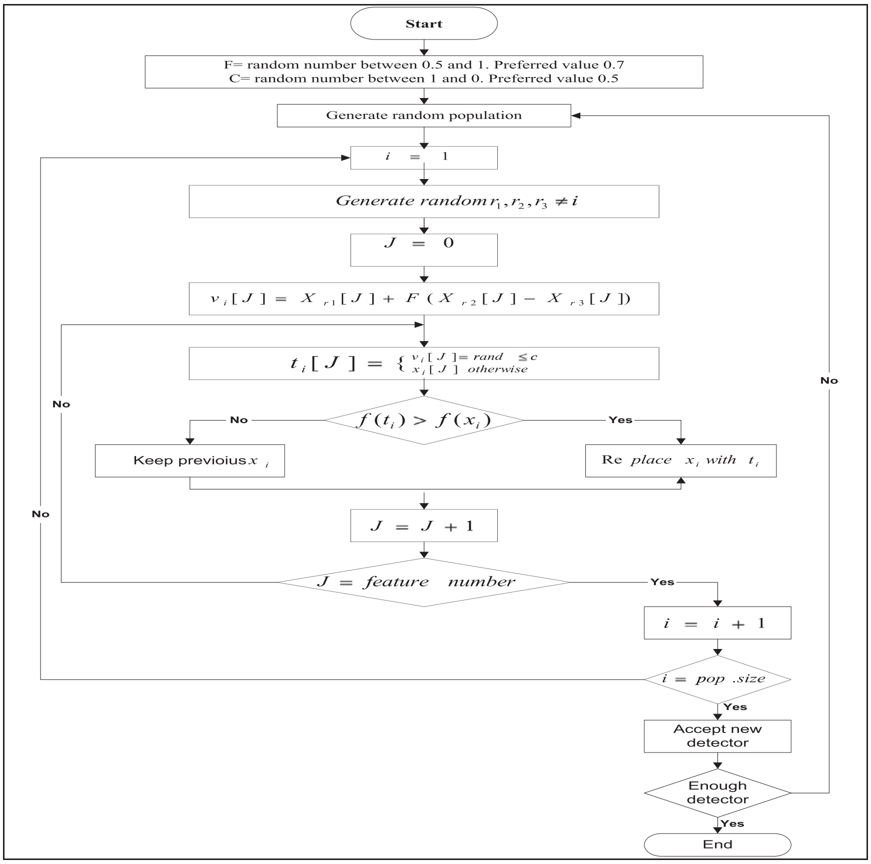

Start
F= random number between 0.5 and 1. Preferred value 0.7
C= random number between 1 and 0. Preferred value 0.5
Generate random population
$i = 1$
Generate random $r_1, r_2, r_3 \neq i$
$J = 0$
$v_i[J] = X_{r1}[J] + F(X_{r2}[J] - X_{r3}[J])$
$t_i[J] = \{ v_i[J] = rand \le c;\ x_i[J]\ otherwise$
$f(t_i) > f(x_i)$
No
Yes
Keep previoius x_i
Re place x_i with t_i
$J = J + 1$
$J = feature\ number$
No
No
Yes
$i = i + 1$
$i = pop.size$
Yes
Accept new detector
Enough detector
No
Yes
End

Fig. 4. Proposed hybrid NSA–DE detector generation model.

The local differential evolution was used to generate detectors one after the other in order to attain the best combination. Each generated detector only needs to cover a short distance to attain coverage in the spam space. The model of the hybrid system is presented in Fig. 4. It shows the detector generation phase of the real valued negative selection algorithm using differential evolution in the generation of detectors.

3.2.2.1. Detector generation parameters and implementation. The population size = 100; the mutation factor F = random number between 0.5 and 1. Preferred value is 0.7; crossover rate C = random number between 1 and 0, where the preferred value is 0.5.

Initializing the population:

$j=0;\ \ i=1,2,3,\ldots,P;$

where P is the size of the population.

A differential candidate vector is added to the population of the vector by mutation. For each target detector vector $x_{i,}[J]$, a mutation vector is produced as follows:

$$V_i[J]=X_{r_1}[J]+F(X_{r_2}[J]-X_{r_3}[J]) \tag{9}$$

F is the mutation factor; it provides the amplification between two individual differences $(X_{r_2}[J]-X_{r_3}[J])$. It is usually taken in the range [0,1] to avoid search stagnation where $r_1, r_2, r_3 \in \{1,2,3,\ldots,p\}$ is chosen randomly and p is the number of population.

By replacing parameters from the target candidate detector vector to generate a trial candidate detector vector with the corresponding parameters of randomly generated mutant, we apply recombination or crossover CR to the population.

Therefore, crossover constant $CR=(0\le c\le 1)$

rand $J \in [0,1]$; is a random number that is less than c.

$$U_i[J]=\begin{cases} v_i\,[J] & \text{if } rand \le CR \\ x_i\,[J] & \text{otherwise} \end{cases} \tag{10}$$

where J = 1, 2, 3,…, d, where d is the number of parameters to be optimized.

If the trial candidate detector vector $U_i[J]$ has equal or lower value than the target candidate detector vector $x_i[J]$ the target candidate detector vector is replaced in the next generation. e.g. replace x_i with u_i or else x_i is retained in the population for at least one more generation. This is represented as follows:

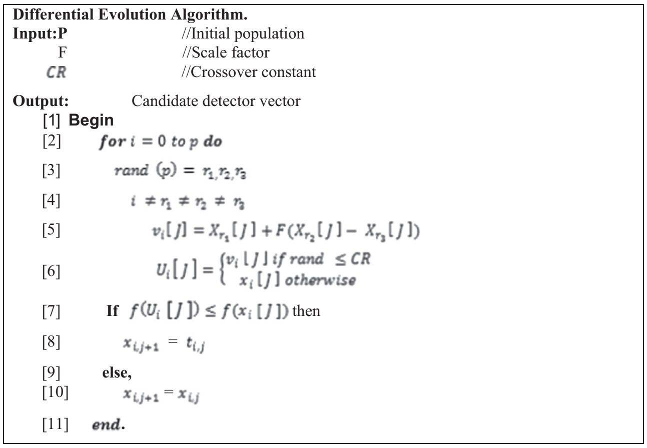
$$f(U_i[J])\le f(x_i[J]) \tag{11}$$

The process of mutation, recombination, and selection is required once a new population is installed until specific termination criteria are reached.

$J\leftarrow J+1$ determine the incremental features until the maximum number of generated detectors is reached. This makes the local differential evolution unique as best features are acquired one after the other in order to attain best combination (Fig. 5).

The process of the methodology can be explained as follows:

Step 1: Choose parameters of population size P, mutation factor F and crossover rate C.
Step 2: Initialization of the population of $i=1,2,\ldots,p$ real value d-dimensional solution vectors with generated random values based on a probability distribution in the d-dimensional problem space.
Step 3: A candidate detector vector differential is added to a population of vectors by mutation as represented in Eq. (9). For each target candidate detector vector $x_i[J]$ a mutant is produced.



```
Differential Evolution Algorithm.
Input:P              //Initial population
      F              //Scale factor
      CR             //Crossover constant

Output:       Candidate detector vector
  Begin
      for $i = 0$ to $p$ do
        $rand\ (p) = r_{1,} r_{2,} r_3$
          $i \neq r_1 \neq r_2 \neq r_3$
            $v_i[J] = X_{r_1}[J] + F(X_{r_2}[J] - X_{r_3}[J])$
            $U_i[J] = \begin{cases} v_i\,[J] \text{ if } rand \le CR \\ x_i\,[J] \text{ otherwise} \end{cases}$
      If $f(U_i\,[J]) \le f(x_i\,[J])$ then
          $x_{i,j+1} = t_{i,j}$
      else,
          $x_{i,j+1} = x_{i,j}$
    end.
```

Fig. 5. Differential evolution algorithm.

Step 4: Recombination is applied to the population to generate a trial candidate detector vector as shown in Eq. (10)
Step 5: Compare both trail candidate detector vector and target candidate detector vector to produce a better offspring as in Eq. (11)
Step 6: Repeat process until maximum generation of detectors is attained and specified termination criteria is satisfied.

3.2.2.2. Implementation model. Lets assume the jth parameter has its lower and upper bound as $x_{min,\,j}$ and $x_{max,\,j}$ respectively.

Initializing the jth parameter of the ith population (P)

$$x_{i,\,j}=x_{min,\,j}+rand\,j\,(0,1)(x_{max,\,j}-x_{min,\,j}) \tag{12}$$

Suppose $F(x)$ is a function of single variable x

$\{x_1,\ x_2,\ldots,x_P\}$ is a trial solution with P as the population size

x_m be the mth individual of the population.

$m=1(1)P$ is used as a target vector in differential evolution iteration.

This undergoes a modelled step of mutation, crossover, and selection were the upper case number denotes random variables.

Three trial solutions are chosen from the population at random.

Let $X_{r_1}, X_{r_2}, X_{r_3}$ be the trial solution from the population at random

$X_{r_1}, X_{r_2}, X_{r_3}$ are independent of each other based on the assumption of the probability Pr as:

$$Pr(X_{r_i}=X_l\cap X_{r_j}=X_k)=Pr(X_{r_i}=X_l)\,Pr(X_{r_j}=X_k) \tag{13}$$

where $i,\ j=1,\ 2,\ \ 3$ and $k,l=1(1)P$ and $i\ \neq j$

The difference between X_{r_2}, X_{r_3} is scaled by factor F while X_{r_1} is added with the scaled difference.

We assume V_m as the generated donor vector

$$V_m=X_{r_1}+F(X_{r_2}-X_{r_3}) \tag{14}$$

The crossover CR equals the time probability of the event that $U_m=V_m$. Let us assume that a trial component vector must come from the donor vector. This assumption leads to the following theorem.

Theorem. *The expected value E of* U_m^2 *with* $U_m=V_m$ *can be represented as*

$$E(\,U_m^2)=(1-CR)x_m^2+CR(2F^2+1)var(x)+CRx_{av}^2 \tag{15}$$

Proof. The probability of the event $r\le CR=Pr\,(r\le CR)$ = area of the spam region $=1\times CR=CR$

$r \le CR$ and $r > CR$ are mutually exclusive events.

$$Pr(r > CR) = 1 - Pr(r \le CR) = 1 - CR \tag{16}$$

$$E(U_m) = Pr(r < CR)x_m + \sum_{i=1}^{P}\sum_{j=1}^{P}\sum_{k=1}^{P}[Pr\{ r \le CR) \cap ((X_{r_1} = X_i) \cap (X_{r_2} = X_j) \cap (X_{r_3} = X_K))\}\{X_i + F(x_j - x_k)\}] \tag{17}$$

$$E(U_m) = (1-CR)x_m + CR\frac{1}{NP}\sum_{i=1}^{P} x_i$$

$$E(U_m) = (1-CR)x_m + CRx_{av} \tag{18}$$

Since mutation and crossover are independent of one another; r is independent of X_{r_1}, X_{r_2}, X_{r_3}. They are independent random variables.

$$E(U_m^2) = Pr(r > CR)x_m^2 + \sum_{i=1}^{P}\sum_{j=1}^{P}\sum_{k=1}^{P}[Pr\{ r \le CR) \cap ((X_{r_1} = X_i) \cap (X_{r_2} = X_j) \cap (X_{r_3} = X_K))\}\{X_i + F(x_j - x_k)\}^2] \tag{19}$$

$$E(U_m^2) = (1-CR)x_m^2 + CR\frac{1}{P^3}\sum_{i=1}^{P}\sum_{j=1}^{P}\sum_{k=1}^{P}\{x_i + F(x_j - x_k)\}^2 \tag{20}$$

$$E(U_m^2) = (1-CR)x_m^2 + CR\frac{1}{P^3}\sum_{i=1}^{P}\sum_{j=1}^{P}\sum_{k=1}^{P}\left\{x_i^2 + F^2(x_j - x_k)^2 + 2Fx_i(x_j - x_k)\right\} \tag{21}$$

$$E(U_m^2) = (1-CR)x_m^2 + CR\left[(2F^2+1)\frac{1}{P}\sum_{i=1}^{P} x_i^2 - 2F^2\left(\frac{1}{P}\sum_{i=1}^{P} x_i\right)^2\right] \tag{22}$$

$$E(U_m^2) = (1-CR)x_m^2 + CR\left[(2F^2+1)\frac{1}{P}\sum_{i=1}^{P} x_i^2 - \left(\frac{1}{P}\sum_{i=1}^{P} x_i\right)^2\right] + CR\left(\frac{1}{P}\sum_{i=1}^{P} x_i\right)^2 \tag{23}$$

where variable

$$(x) = \frac{1}{P}\sum_{i=1}^{P} x_i^2 - \left(\frac{1}{P}\sum_{i=1}^{P} x_i\right)^2 \text{ and } x_{av} = \frac{1}{P}\sum_{i=1}^{P} x_i \tag{24}$$

Therefore,

$$E(U_m^2) = (1-CR)x_m^2 + CR[(2F^2+1)var(x) + CRx_{av}^2 \quad \text{proved.} \tag{25}$$

Eq. (13) is for discrete function as proved. Since we are dealing with a continuous function, the variance of a continuous function is represented as

$$\text{Mean} = \int_a^b x_i(t)f(t)dt \tag{26}$$

$$\text{Variance} = \int_a^b x_i(t)^2 f\,(t)dt - \int_a^b x_i(t)f(t)dt \tag{27}$$

$f(t) = Mutation \Rightarrow f(x_j\,(t) - x_k(t)$, where t is the time taken for each mutation process.

From Eq. (12), for any change in U_m there will be ΔU_m

$$\therefore E[(U_m + \Delta U_m)^2] = E(U_m^2) + 2E(U_m)E(\Delta U_m) + E(\Delta U_m^2) \tag{28}$$

Substituting Eqs. (14) and (15) into $E(U_m^2)$, $E(\Delta U_m^2)$, $E(U_m)$ and $E(\Delta U_m)$ respevtively, we have:

$$E(U_m^2) = (1-CR)x_i(t)^2 + CR[(2F^2+1)\left[\int_a^b x_i(t)^2 f(t)dt - \int_a^b x_i(t)f(t)dt\right] + CR\left[\int_a^b x_i(t)f(t)dt\right]^2 \tag{29}$$

$$E(\Delta U_m^2) = (1-CR)\Delta x_i(t)^2 + CR[(2F^2+1)\left[\int_a^b \Delta x_i(t)^2 f(\Delta t)dt - \int_a^b \Delta x_i(t)f(\Delta t)\,dt\right] + CR[\int_a^b \Delta x_i(t)f(\Delta t)\,dt]^2 \tag{30}$$

$$E(U_m) = (1-CR)x_m(t) + CR\int_a^b x_i(t)f(t)dt \tag{31}$$

$$E(\Delta U_m) = (1-CR)x_m(t) + CR\int_a^b x_i(\Delta t)f(\Delta t)\,dt \tag{32}$$

We further substitute Eqs. (29)–(32) into Eq. (28) to generate expectation equation of $E[(U_m + \Delta U_m)^2]$ which is the change in target vector at a given time t. x_m is the best selection during the mutation process, F is the mutation of selection without time t of x_i, x_j, x_k, $F(t)$ is the mutation of selection with time t change in $x_i\,(t)$, $x_j\,(t)$, $x_k(t)$, F^I is the change of mutation with respect to time t, Δx is the difference between the selected point and CR is the crossover mutation.

3.2.3. Computing the generated detectors in the spam space

From Eqs. (7) and (8) of the normalized non-spam space, the non-spam space is represented in Eq. (33) with radius Rs as:

$$S = \{X_i \,|\, i = 1,\ 2, \ldots, m; Rs = r\} \tag{33}$$

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

$$X_i = \{x_{i1}, x_{i2},\ x_{i3}, \ldots, x_{iN}\}, \quad i = 1,\ 2,\ 3, \ldots, m \tag{34}$$

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

$$d_j\ = (C_{j\ ,}\ R_j^d) \tag{35}$$

Eq. (35) denotes one detector d_j where $C_j = \{C_{j1},\ C_{j2}, C_{j3}, \ldots, C_{jN}\}$ is the detector centre, R_j is the detector radius. The Euclidean distance was used as the matching measurement. The distance between non-spam sample X_i and the detector d_j can be defined as

$$L(X_i,\ d_j\ ) = \sqrt{(x_{i1} - C_{j1})^2 + \cdots + (x_{iN} - C_{jN})^2} \tag{36}$$

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

$$\ltimes = L(X_i,\ d_j\ ) - Rs \tag{37}$$

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

$$R^d,\ j = \min\,(\ltimes)\ \text{if} \ltimes\ \le 0 \tag{38}$$

Also, if detector d_j matches any non-spam samples, 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 does not match any detector in the spam space, it is labelled as non-spam.

3.3. Computation of fitness function in the spam and non-spam space

One most important quality of spam and non-spam detector space is how distant the generated spam detector is from the non-spam space; this helps in improving the proposed model. We decided to employ the local outlier factor as a fitness function to maximize the distance between generated spam detectors and the non-spam space. An outlier can be defined as a data point that is not the same as the remaining data with respect to some measures. The technique will model the data point with the use of a stochastic distribution (Sajesh and Srinivasan, 2011) and the point is determined to be an outlier based on its relationship with the model. The outlier detection algorithm that was proposed as fitness function in this study to maximize the generated spam detector space 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 local neighbourhood.

- Let us assume k-distance (i) to be the distance of the generated detector (i) to the nearest neighbourhood (non-spam).
- Set of k-nearest neighbour (non-spam element) includes all spam detectors at this distance.
- Set S of k-nearest neighbour is denoted as $N_k(i)$.
- Here non-spam space$=S$.
- This distance is used to define the reach-ability distance.
- $\text{Reach}-\text{ability}-\text{distance}_k(i,s)=\ \max\{k-distance(s),\ d_{i,s}\}$
- The local reach-ability density (LRD) of r is defined as

$$lrd(i)=1/\left(\frac{\sum s\epsilon N_k(i) reachability-distance_k\ (i,s)}{|N_k(i)|}\right) \tag{39}$$

Eq. (39) is the quotient of the average reach-ability distance of the generated detector i from non-spam element. It is not the average reach-ability 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 the equation below:

$$LOF_{\text{K}}(i)=\frac{\sum s\in N_k(i)(lrd(s)/lrd(i))}{|N_k(i)|}=\frac{\sum s\in N_k(i)lrd\ (s)}{|N_K(i)|}/lrd(i) \tag{40}$$

Eq. (40) shows the average local reach-ability density of the neighbour divided by the candidate detectors own local reach-ability density. In this scenario, values of spam detector approximately 1 indicates that the detector is comparable to its neighbour (not an outlier) and value < 1 indicates a dense region (which is an inliers) while value > 1 indicates an outlier. The major idea of this technique is to assign to each detector the degree of being an outlier. The degree is called the local outlier factor (LOF) of the detector as shown in Eq. (40). The methodology of the computation of LOF for all detectors is explained in steps as follows:

Step 1: For each generated detector i compute k distance element in non-spam region (distance of k-nearest neighbour in non-spam space s) as shown in Eq. (41).
Step 2: Eq. (42) computes reach-ability distance for generated detector i with non-spam region as: $\text{Reach}-\text{dist.}(i)=\max\{k-distance(s),\ d_{i,s}\}$, when $d_{i,s}$ is the distance from detector i to non-spam space s.
Step 3: Computation of the local reach-ability density of generated detector i as inverse of the average reach-ability distance based on *Minpts* (minimum number of non-spam region) nearest neighbour of detector i in Eq. (43).
Step 4: Eq. (44) computes LOF of generated detector i as average of the ratios of the local reach-ability density of the neighbours in non-spam space divided by number of the objects own local reach-ability density.

Assume p as the population of the generated detectors, S is the non-spam space and i is the ith detector in p (Fig. 6)

For each detector i we have

$$i\in p.\ \text{Max}(k-dist(s)) \tag{41}$$

$$/\ \text{Reach}-\text{ability}\ distance_p/{*}\ \max(dist_{i,s}) \tag{42}$$

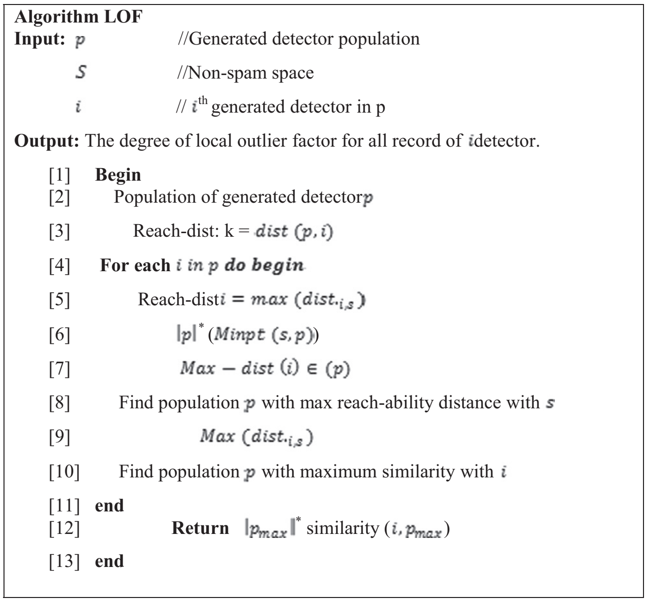
$$|p|^*(Minpt\ (s,\ i)) \tag{43}$$

$$|p|^*(similarity\ \ (i,\ p) \tag{44}$$

3.4. Overlapping in spam space (generated detector)

Overlapping exists during the generation of detectors in the spam space. This is as a result of detectors not spreading properly in the spam space. The detectors overlapped each other and affect the evaluation of the algorithm negatively. A fitness function is introduced to solve the problem of overlapping in spam space, the minimum and maximum distance between two overlapped detectors a and b in spam space are calculated. Each of the detectors is modelled and an algorithm is developed to eliminate or minimize overlapping. The overlapped detectors will be pulled apart from each other along the shortest distance to correct overlapping between them.

The overlapping distance is calculated by subtracting the distance between each overlapped detector along the shortest path and also doubling the radius. The distance that each detector is to travel is half the distance of the direction each detector needs to travel that same distance. The sum of the Y-distance is divided by the sum of X-distance and the arc tan of the result in the division is the angle that is used in the offset. Fig. 7 shows two



```
Algorithm LOF
Input: $p$        //Generated detector population
       $S$        //Non-spam space
       $i$        // $i^{th}$ generated detector in p
Output: The degree of local outlier factor for all record of $i$detector.
Begin
  Population of generated detector$p$
    Reach-dist: k = $dist\ (p,i)$
 For each $i$ in $p$ do begin
    Reach-dist$i = max\ (dist._{i,s})$
        $|p|^*(Minpt\ (s,p))$
        $Max - dist\ (i) \in (p)$
   Find population $p$ with max reach-ability distance with $s$
          $Max\ (dist._{i,s})$
   Find population $p$ with maximum similarity with $i$
end
       Return  $|p_{max}\|^*$ similarity ($i, p_{max}$)
end
```

Fig. 6. Algorithm of fitness function in spam and non-spam space.

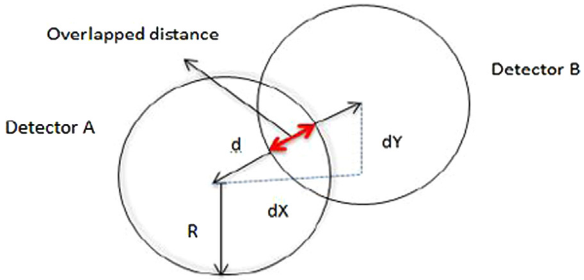


Fig. 7. Overlapped detectors. (For interpretation of the references to colour in this figure, the reader is referred to the web version of this article.)

overlapping detectors; distance between both overlapped detectors is shown as red. *dX* and *dY* represent the *X* distance and *Y* distance respectively with radius *R*.

Let us assume both detectors as *X* and *Y* coordinates respectively.

$$\text{Detector } A=(X_1,\ Y_1) \tag{45}$$

$$\text{Detector } B=(X_2,\ Y_2) \tag{46}$$

Calculating the distance between detectors coordinate *X* and *Y* in Eqs. (47) and (48) we have

$$(X-\text{axis});\ \ dX=X_1-X_2 \tag{47}$$

$$(Y-\text{axis});\ \ dY=Y_1-Y_2 \tag{48}$$

Calculating the distance between the two detectors *D* in Eq. (49) we have:

$$D=sqrt\ ((dX2)+(dY2)) \tag{49}$$

Calculating the distance between each overlapped detectors along the shortest path will also double the radius each overlapped detector needed to be moved as follows:

$$Dd=2R-D \tag{50}$$

The result of the algorithm above shows how the equations solve overlapping of detectors. The angle of offset is defined as follows:

$$\tan^{-1}(dY/dX) \tag{51}$$

where *d* represents distance and *D* represents detectors.

3.5. Flow of proposed model

A dataflow diagram for the proposed NSA–DE hybrid model was proposed in Fig. 8. This becomes imperative in order 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 the model in a sequential manner in lure of respect to standard practice.

Though, the proposed hybridization model works at the random generation phase of negative selection algorithm, differential evolution is initialized at the detector generation phase of the algorithm. Several researchers have proved that the effectiveness of any computational algorithm depends on how effective data is represented for learning process (Gong et al., 2012). This is the main reason we choose to explore the hybridization of negative selection algorithm at detector generation phase. This makes the proposed model different from other models as our hybridization model will be implemented at the random generation of detector phase of negative selection algorithm. From Fig. 8, after the input data, the sample datasets are divided into training and testing sets. The training set was used as a prospective detector (candidate detector) by implementing differential evolution for generation of detector. Euclidean distance with a threshold value is further used to measure each generated detector before occupying the spam space. If the generated detector matches with the non-spam region, it is discarded but if it does 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. The maximization of distance with spam detector and non-spam element is also improved upon by the use of local outlier factor (LOF) as fitness function to maximize the distance of spam detector for good coverage. Distance between two overlapped detectors in the spam region is also calculated as fitness function to solve the problem of overlapping among detectors in spam space. The testing set is separated from the training set, the testing set attributes was used after proper coverage and maximization of the spam space for testing; at the end of the

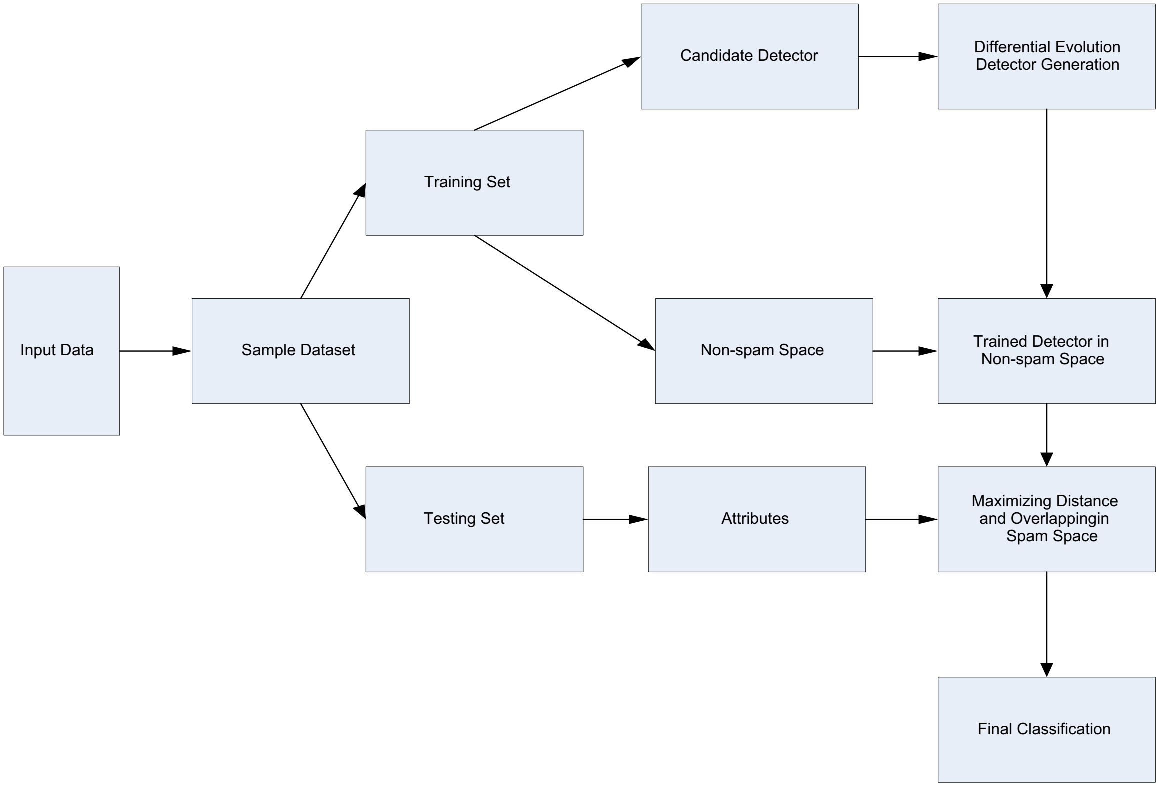

Input Data
Sample Dataset
Training Set
Testing Set
Candidate Detector
Differential Evolution Detector Generation
Non-spam Space
Trained Detector in Non-spam Space
Attributes
Maximizing Distance and Overlappingin Spam Space
Final Classification

Fig. 8. Dataflow diagram of proposed NSA–DE hybrid model.

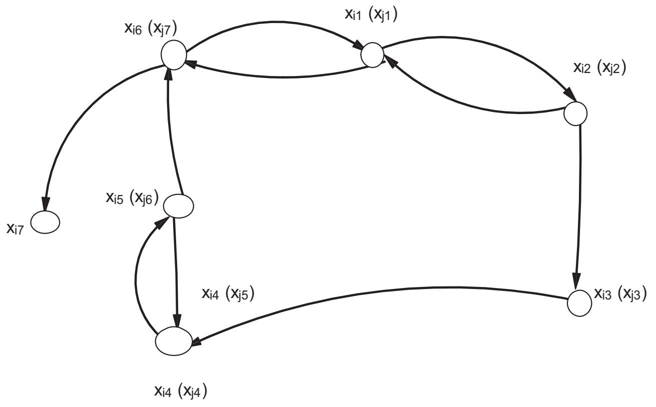


Fig. 9. Graph G_1.

testing, a final value is acquired by classification and final output generated for the hybrid scheme.

3.5.1. Graphical representation of the model

The graph theory is implemented to model the detection generation system. We represent the computer system as graph G while the algorithm to be executed by G is represented as negative selection algorithm-differential evolution (NSA–DE) algorithm. G and NSA–DE are represented by means of a graph whose nodes (entity) are a representation of the computing facility. The paper present a graph model and algorithms for computing system for detector generation as represented in Fig. 9. The graph shows the computing facility of the computation and the interconnection between the facilities. The representation of the model is in the form of a facility graph. The nodes (entity) of the facility graph G represent the system facilities and the edges (Relationship) represent the access links between facilities. The facility is the hardware and software component of the system. The graph is an indication of the type of facilities that are accessed by other facilities. Fig. 9 shows the graph with the nodes (entity) x_{i1} access the nodes of x_{i2} and x_{i6}. The nodes x_{i2} of facility type x_{j2} and x_{i6} of facility type x_{j7} access the facility type x_{j1} of node x_{i1}. Similarly, the node x_{i5} with facility type x_{j6} access the facility type x_{j5} of node x_{i4}. Each x_i node as a facility type x_j access which represent both nodes (entity) and edges (relationship) respectively.

The NSA–DE model is executable by G_1 if NSA–DE model is isomorphic to a sub-graph of G_1.

Therefore, let us assume l labelled example $(x_{i1}, x_{j1})\ldots(x_{il}, x_{jl}) \in \mathbb{R}^d \times \{1, 0\}$ where x_i is the d-dimensional feature vectors associated with the ith host and x_j is the label 1 for spam and 0 for non-spam.

We have a set of weighted directed graph with nodes $x_{l+1}\ldots x_n$. Let x be the set of pairs (i, j) whenever node i is connected to node j, and let $k \geq 0$ be the weight of the link from x_i to x_j.

The individual population is created randomly. Each x_i consist of k nodes (k is a representation of the number of weights in the trained negative selection algorithm). Each (jth $[1, k]$) edges of node x_i have value of determined variable ranges from min_j to max_j. The values of $min_j = 0$ and $max_j = 1$ has proposed. The coding scheme of the weight in node x_i is connected to the edges.

A mutated individual v_i (vector) for each x_i individual in the population is based on the formulae

$$v_i = x_{r1} + F(x_{r2} - x_{r3}) \tag{52}$$

where $F \in (0.5, 1)$, and $r1,\ r2,\ r3,\ i \in [1,\ popsize]$ satisfy constraint:

$$r1 \neq r2 \neq r3 \neq i \tag{53}$$

Indices $r2$ and $r3$ points at the individual that are choosing randomly from the population while index $r1$ points at the population of the individual that is best with the lowest value of training error function $ERR(.)$.

All the x_i individual are crossed-over with mutated v_i individual. Due to the cross-over operation u_i individual is created. The operation is as follows:

For nodes $x_i = (x_{i1,}\ x_{i2},\ x_{i3}\ldots x_{ij})$ and $v_i = (v_{i1,}\ v_{i2},\ v_{i3}\ldots v_{ij})$; for each edges $j \in [1; k]$ of the node x_i, a randomly generated number ran_j ranging from [0, 1] is presented with the following rules:

if $rand_j < CR$ Then

$u_{ij} = v_{ij}$

Use

$$u_{ij} = x_{ij} \tag{54}$$

where $CR \in [0; 1]$

The selection of the nodes to new population is performed based on the rules below:

if $ERR\ (u_i) < ERR\ (x_i)$ then;

Replace x_i by u_i in the new population

Else

Leave x_i in the new population (55)

It then checks if the algorithm as reached the number of generation required. If number of generation reached, the algorithm is stopped and the result stored in node x_{r1} is returned; else the algorithm jumped back to start the mutation process.

Each generation from differential evolution is used to train the negative selection algorithm.

The weighted sum of the differential evolution is taken as a model of negative selection algorithm edges. The weighted sum ws_j of jth edges is defined as

$$ws_j = \sum_{i=1}^{p} w,\ i,\ ju_i \tag{56}$$

where p is the number of input in jth edges, w_{ij} is the value of weight representing the connection between jth edges and its ith input node. u_i is the value occurring on ith input.

The classic model of negative selection algorithm includes the matching measurement with Euclidean distance and the measure of the threshold value.

The distance between non-spam sample X_i and the weighted sum ws_j can be defined as

$$L(X_i,\ ws_j\ ) = \sqrt{(x_{i1} - C_{j1})^2 + \cdots + (x_{iN} - C_{jN})^2} \tag{57}$$

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

$$\ltimes = L(X_i,\ ws_j) - Rs \tag{58}$$

The detector sw_j fails to match the non-spam sample X_i if $\ltimes > 0$, therefore if sw_j does not match any non-spam sample, it will be retained in the detector set.

There is a 1–1 mapping from the nodes of x_i into the nodes of sw_j. This is an indication that sw_j is made up of all the facilities and connections between facilities required by x_i. x_i can be inserted in graph G, if G_1 and G_2 represent a system and an algorithm, G_2 is then executable by G_1. G_2 is represented in Fig. 10.

The isomorphism is as follows:

$(y_1, y_2) \rightarrow (x_{i1}, x_{i2})$

$(y_1, y_2) \rightarrow (x_{i6}, x_{i1})$

A k-detector D in a graph G, is the implementation of any k-nodes (entity) $\{x_{i1}, x_{i2},\ x_{i3},\ x_{i4}, \ldots, x_{i7}\}$ from G. All edges (relationship) connected to the nodes (entity) are also implemented. As a result of this, the graph will be denoted as G^D. This can be represented as $D = \{x_{i1}, x_{i2},\ x_{i3},\ x_{i4}, \ldots, x_{i7}\}$.

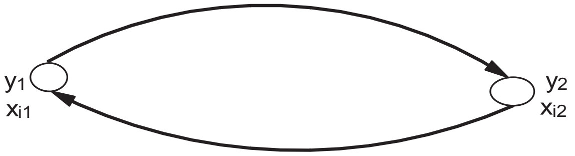


Fig. 10. Graph G_2.

Therefore the basic concepts from this definition of detector generation model are:

1. The implementation of a graph G for detector generation with respect to NSA–DE and detectors D, if NSA–DE is executable by G^D.
2. G is a detector generator with respect to a set of algorithm (NSA–DE) $\{A_1, A_2, A_3, A_4, \ldots, A_P\}$ and a set of generated detector $\{D_1, D_2, D_3, D_4, \ldots, D_q\}$, if A_i is executable by G^{Dj} for all i and j where $1 \le i \le p$.

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 hybrid 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 validation of 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 the course of the section.

4.1. Spam base dataset analysis

The corpus bench mark is obtained from spam base dataset which is an acquisition from email spam messages. In acquiring this email spam messages, 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 Hopkins et al. (1999). Acquisition of this corpus is already pre-processed, unlike most corpora 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 are the percentage of manifestation of the special characters ";", "(", "[", "!", "$" and "#". Some other 3 features are a representation of various measures of manifestation of capital letters that exist in the text of the messages. Lastly, is the class label in the corpus; it gives the condition of an instance to be spam or non-spam with 1 and 0 representation. Spam base dataset is one of the best test beds that performs well (Koprinska, 2007) during learning and evaluation techniques.

4.2. Criteria for performance evaluation

Different measures are used to evaluate the accuracy and performance of NSA and NSA–DE model. To evaluate and compare performance and accuracy of both models, statistical quality measures used in machine learning and data mining journals were used (Biggio et al., 2011). They are Sensitivity (SN), Specificity (SP), Positive prediction value (PPV), Accuracy (ACC), Negative prediction value (NPV), Correlation coefficient (CC) and F-measure ($F1$). See Biggio et al. (2011) for more detailed mathematical formulae; though, they are briefly discussed below. Table 1 shows the performance evaluation for all listed criteria of 1000 generated detectors with threshold value of 0.4.

Table 1
At 1000 generated detectors with threshold value of 0.4, gives summary and comparison of results in percentage for NSA and NSA–DE model.

Model	*ACC*	*MCC*	*F1*	*SN*	*PPV*	*SP*	*NPV*
NSA	68.86	36.06	36.01	22.24	94.53	99.16	66.24
NSA–DE	80.66	60.08	69.76	56.62	90.86	96.30	77.35

Note: ACC=accuracy, CC=correlation coefficient, $F1$=F measure, SN=sensitivity, PPV=Positive prediction value, SP=Specificity and NPV=Negative prediction value.

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

$$SN = \frac{TP}{TP+FN} \tag{59}$$

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

$$SP = \frac{TN}{TN+FP} \tag{60}$$

(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

$$PPV = \frac{TP}{TP+FP} \tag{61}$$

(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.

$$NPV = \frac{TN}{FN+TN} \tag{62}$$

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

$$Acc = \frac{TP+TN}{TP+TN+FN+FP} \tag{63}$$

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

$$CC = \frac{[(TP)(TN)-(FP)(FN)]}{\sqrt{(TP+FN)(TP+FP)(TN+FP)(TN+FN)}} \tag{64}$$

(vii) F-measure ($F1$): It is a measure that combines both positive predictive value and sensitivity. The positive predictive value and sensitivity are evenly weighted.

$$F1 = 2 \times \frac{\text{Positive prdictive value} \times \text{Sensitivity}}{\text{Positive prdictive value} + \text{Sensitivity}} \tag{65}$$

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

$$T=\frac{\overline{X}_I-\overline{X}_2}{\sqrt{(S_{1(2/n_1)}+S_{2(2/n_2)})}} \tag{66}$$

The evaluation equations listed from (i) to (viii) above, 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–DE hybrid model is implemented by the division of the dataset using a stratified sample approach with 70% training set and 30% testing set for investigating the performance of the new model on an unseen data. The training set is 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 the same platform. The evaluation of both NSA model and its hybrid are implemented with threshold values of between 0.1 and 1 while the number of generated detectors is between 100 and 8000. The different threshold value and number of detectors generated have tremendous impact on the final output measure.

5. Experimental results and discussion

Performance comparison between negative selection algorithm model and proposed hybrid model using validation of an unseen data is summarized in Figs. 11–13. The result of hybridized NSA–DE model out performs the NSA model. The proposed hybrid model shows an improved accuracy when compared with the standard model which performs poorly by all measuring standards. It is clear that the hybrid model is better than the individual models due to the good forecasting scheme used in the evaluation of the model.

Though, performance evaluation criteria of the two models was listed in Section 4.2, graphical representation of accuracy, $F1$ measure and negative prediction values were represented in the paper as shown in Figs. 11–13 respectively. Accuracy measures the percentage of sample that is correctly classified. It can be observed that the proposed hybrid model performs better than negative selection algorithm model with average accuracy of the standard negative selection algorithm at 65.147%, the hybrid negative selection algorithm and differential evolution model is at 69.383%. At 7000 detectors with threshold value of 0.4, accuracy for negative selection algorithm is 68.863% while hybrid negative selection algorithm and differential evolution is at 83.056%. Also, $F1$ measures combine both positive prediction value and sensitivity by weighing both evenly. At 7000 detectors and threshold value of 0.4 positive predictive values for negative selection algorithm is 94.531% while hybrid negative selection algorithm and differential evolution is 87.081%; sensitivity with the same generated detectors and threshold values for negative selection algorithm is 22.243% while hybrid negative selection algorithm and differential evolution is at 66.912%. Overall $F1$ measure for negative selection algorithm is 36.012% while hybrid negative selection algorithm and differential evolution is 75.676%. Average $F1$ measure for negative selection algorithm is 22.094% while hybrid negative selection algorithm and differential evolution is 38.944%.

From the results reported, it could be noted easily that hybrid NSA–DE model performed better in all ramifications. This proves consistency of the quality of measure used in every respect. Fig. 13 shows the negative prediction value with 7000 detectors and threshold value of 0.4 for negative selection algorithm at 66.241% while hybrid negative selection algorithm and differential evolution is 81.308%. The average negative prediction values for negative selection algorithm is 63.872% while hybrid negative selection algorithm and differential evolution is 68.059%. The NSA model performs very low when compared with the hybrid model. The improvement is on a very big scale and it shows the relevance of differential evolution 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 gives summary of result obtained with 1000 generated detectors and threshold value of 0.4. The table shows the improvement of hybrid negative selection algorithm based on detector generation and maximization of the coverage area.

5.1. Statistical t-test

The p value (probability) is used to determine if the population mean differs 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 and hybrid negative selection algorithm-differential evolution at 0.05 alpha levels. This shows that there is a mutual unity between negative selection algorithm and hybrid negative selection algorithm-differential evolution among their variables. This is corroborated by the mean of each of the negative selection algorithm and hybrid negative selection algorithm-differential evolution ranging between 65.1477 and 70.4763 for accuracy and also the standard deviation indicated that there is a deviation between 0.98 and 1.89. Other evaluation measure analysis is represented in Table 2.

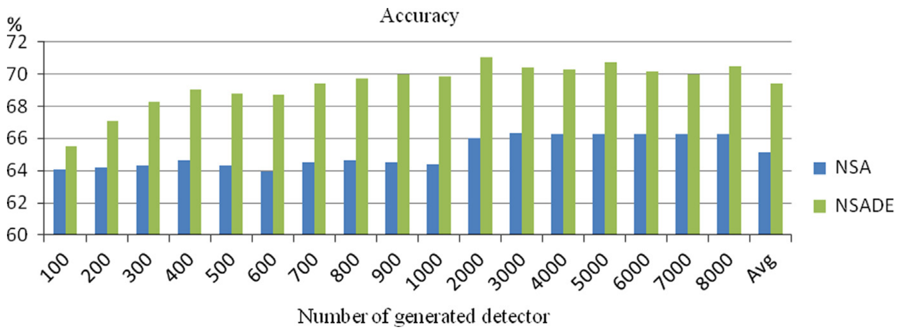


Fig. 11. Accuracy of negative selection algorithm and hybrid negative selection algorithm with differential evolution.

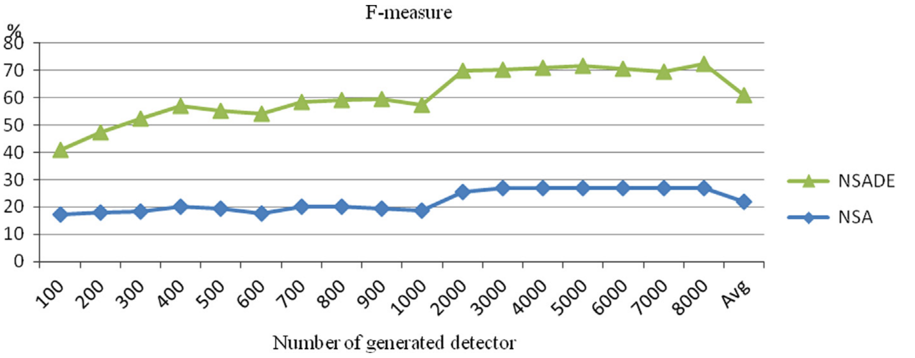


Fig. 12. *F*-measure of negative selection algorithm and hybrid negative selection algorithm with differential evolution.

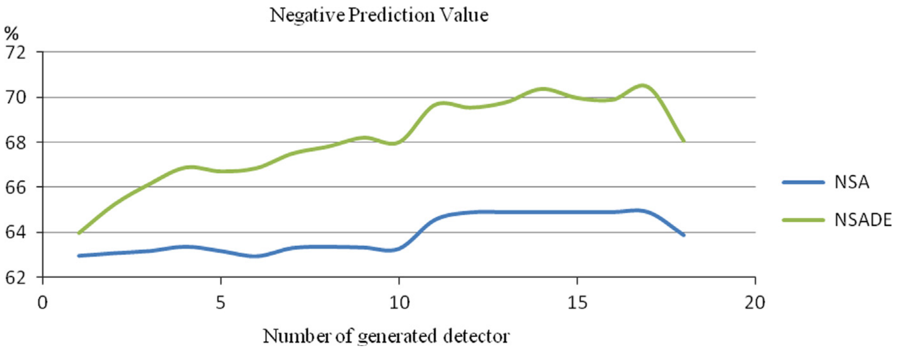


Fig. 13. Negative predictive value of negative selection algorithm and hybrid negative selection algorithm with differential evolution.

Table 2
t-test for negative selection algorithm (NSA) and hybrid negative selection algorithm-differential evolution (NSA–DE)

Measure	Algorithm	*t*	Df	Mean	SD	Sig	Comment
			$(n-1)$			(2-tailed)	
ACC	NSA	273.003	16	65.1477	0.9840	0.000	Higher
	NSA–DE	203.56	16	69.3828	1.4053	0.000	correlation
F1	NSA	22.385	16	22.0938	4.0694	0.000	Higher
	NSA–DE	27.304	16	38.9436	5.8809	0.000	correlation
PPV	NSA	229.009	16	85.0243	1.5308	0.000	Higher
	NSA–DE	93.457	16	81.7766	3.6078	0.000	correlation
CC	NSA	40.210	16	23.2112	2.3800	0.000	Higher
	NSA–DE	49.363	16	33.6978	2.8146	0.000	correlation
SN	NSA	17.166	16	13.6344	3.2748	0.000	Higher
	NSA–DE	17.468	16	28.6430	6.7610	0.000	correlation
SP	NSA	778.802	16	98.6269	0.5221	0.000	Higher
	NSA–DE	166.871	16	95.8612	2.3686	0.000	correlation
NPV	NSA	309.986	16	63.8717	0.8496	0.000	Higher
	NSA–PSO	145.310	16	68.0587	1.9311	0.000	correlation

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

Therefore, there is significant correlation between the mean of negative selection algorithm and hybrid negative selection algorithm-differential evolution. This also shows a high level of accuracy between them.

6. Conclusion and recommendations

A new hybrid model that combines negative selection algorithm (NSA) and differential evolution (DE) has been proposed and implemented. The uniqueness of this model is that the DE is implemented at the random generation phase of NSA; also the generated detector distance was maximized and overlapping of detectors was also minimized. The detector generation phase of NSA and detector coverage area determines how robust and effective an algorithm will perform. DE implementation improved detector generation phase of NSA while local outlier factor (LOF) was used as fitness function to maximize the distance of generated detectors and distance between overlapped detectors are calculated as fitness function to resolve overlapping between two detectors. The proposed hybrid serves as a better replacement to NSA model. Spam base dataset was used to investigate the performance of NSA model against hybrid NSA–DE model. Performance and accuracy investigation as shown that the proposed hybrid model was able to detect email spam better than the NSA model. Validation of the proposed framework has been carried out with published dataset and an in-depth comparative study of the proposed hybrid model and the standard model has been carried out in order to show the improvement of the proposed hybrid model over the NSA model. Statistical *t*-test was also used to determine the correlation between negative selection algorithm and hybrid negative selection algorithm-differential evolution in this paper. In totality, the empirical report has shown the superiority of the proposed NSA–DE hybrid model over the NSA model. The proposed hybrid system will be useful in other applications as negative selection algorithm solves vast number of complex problems. Based on the results presented, this research should be viewed as an improvement in the field of computational intelligence. As future work, we will propose a parallel hybridization of two evolutionary algorithms to perform 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

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, vol. 5132. Springer, Berlin, Heidelberg, pp. 36–47.

Balthrop, J., Forrest, S., Glickman, M.R., 2002. Revisiting LISYS: parameters and normal behavior. In: Proceedings of the 2002 Congress on Evolutionary Computing: 2002, pp.1045–1050.

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., Barra, T., Ferreira, H., Knidel, H., Castro, L., Zuben, F., 2006. An immunological filter for spam. In: Bersini, H., Carneiro, J. (Eds.), Artificial Immune Systems, vol. 4163. Springer, Berlin, Heidelberg, pp. 446–458.

Biggio, B., Fumera, G., Pillai, I., Roli, F., 2011. A survey and experimental evaluation of image spam filtering techniques. Pattern Recognition Lett. 32 (10), 1436–1446.

Carpinter, J, Hunt, R, 2006. Tightening the net: a review of current and next generation spam filtering tools. Comput. Secur. 25 (8), 566–578.

Cormack, G., Lynam, T., 2007. TREC Public Spam Corpus. 〈http://plguwaterlooca/∼gvcormac/treccorpus07/〉 (cited 15.01.09).

Cormack, G, Smucker, M, Clarke, C, 2011. Efficient and effective spam filtering and re-ranking for large web datasets. Inform. Retr. 14 (5), 441–465.

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

Forrest, S., Perelson, A.S., 1994. Self nonself discrimination in computer.

Golovko, V., Bezobrazov, S., Kachurka, P., Vaitsekhovich, L., 2010. Neural network and artificial immune systems for malware and network intrusion detection. In: Advances in Machine Learning II. Koronacki, J., Wierzchon, S.T., Ras, Z.W., Kacprzyk, J. (Eds.), Springer Berlin Heidelberg, vol. 263, pp. 485–513.

Gong, M., Zhang, J., Ma, J., Jiao, L., 2012. An efficient negative selection algorithm with further training for anomaly detection. Knowledge-Based Syst. 30 (0), 185–191.

Gonzalez, F., Gomez, J., Madhavi, K., Dipankar, D., 2003. An evolutionary approach to generate fuzzy anomaly (attack) signatures. In: Information Assurance Workshop, 2003 IEEE Systems, Man and Cybernetics Society, 18–20 June 2003, pp. 251–259.

Guangchen, R., Ying, T., 2007. Intelligent detection approaches for spam. In: Third International Conference on Natural Computation, 2007 ICNC 2007, 24–27 August 2007, pp. 672–676.

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

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.

Mark Hopkins, Erik Reeber, George Forman, Jaap Suermondt, 1999. Spam Base Dataset. Hewlett-Packard Labs. 〈http://archiveicsuciedu/ml/datasets/Spambase〉.

Hu, Y., Guo, C., Ngai, E.W.T., Liu, M., Chen, S., 2010. A scalable intelligent non-content-based spam-filtering framework. Expert Syst. Appl. 37 (12), 8557–8565.

Khilwani, N., Prakash, A., Shankar, R., Tiwari, M.K., 2008. Fast clonal algorithm. Eng. Appl. Artif. Intell. 21 (1), 106–128.

Knorr, E.M., Ng, R.T., 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., San Francisco, CA, USA; pp. 392–403.

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

Gong, M., Zhang, J., Ma, J., Jiao, L., 2012. An efficient negative selection algorithm with further training for anomaly detection. Knowl.-Based Syst. 30, 185–191.

Massey, B., Thomure, M., Budrevich, R., Long, S., 2003. Learning spam: simple techniques for freely-available software. In: Proceedings of the annual conference on USENIX Annual Technical Conference. (ATEC '03). USENIX Association, Berkeley, CA, USA, p. 13.

Oda, T., White, T., 2003. Increasing the accuracy of a spam-detecting artificial immune system. In: Proceedings of the 2003 Congress on Evolutionary Computation, 2003 CEC '03, 8–12 December 2003, vol. 391, pp. 390–396.

Oda, T., White, T., 2005. Immunity from Spam: an analysis of an artificial immune system for junk email detection. In: Jacob, C., Pilat, M., Bentley, P., Timmis, J. (Eds.), Artificial Immune Systems, vol. 3627. Springer, Berlin, Heidelberg, pp. 276–289.

Oda, T., White, T., 2003. Developing an immunity to Spam. In: Cantú-Paz, E., Foster, J., Deb, K., Davis, L., Roy, R., O'Reilly, U.-M., Beyer, H.-G., Standish, R., Kendall, G., Wilson, S., et al. (Eds.), Genetic and Evolutionary Computation — GECCO, vol. 2723. Springer, Berlin, Heidelberg, pp. 231–242.

Pérez-Diaz, N., Ruano-Ordás, D., Fdez-Riverola, F., Méndez, J.R., 2012. SDAI: an integral evaluation methodology for content-based spam filtering models. Expert Syst. Appl. 39 (16), 12487–12500.

Prabhakar, R., Basavaraju, M., 2010. A novel method of spam mail detection using text based clustering approach. Int. J. Comput. Appl. 5 (4), 15–25.

Prakash, A., Khilwani, N., Tiwari, M.K., Cohen, Y., 2008. Modified immune algorithm for job selection and operation allocation problem in flexible manufacturing systems. Adv. Eng. Softw. 39 (3), 219–232.

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

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.

Schryen, G, 2007. Anti-spam legislation: an analysis of laws and their effectiveness. Inform. Commun. Technol. Law 16 (1), 17–32.

Selamat, A., Olatunji, S.O., Abdul Raheem, A.A., 2012. A hybrid model through the fusion of type-2 fuzzy logic systems and sensitivity-based linear learning method for modeling PVT properties of crude oil systems. Adv. Fuzzy Syst. 2012, 19.

Sirisanyalak, B., Sornil, O., 2007. An artificial immunity-based spam detection system. In: IEEE Congress on evolutionary computation, 2007 CEC 2007, 25–28 September 2007, pp. 3392–3398.

Wamli, M., Dat, T., Dharmendra, S., 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, C., Zhao, Y., 2008. A new fault detection method based on artificial immune systems. Asia-Pac. J. Chem. Eng. 3 (6), 706–711.

Wanli, M., Dat, T., Sharma, D., 2010. A practical study on shape space and its occupancy in negative selection. In: 2010 IEEE Congress on Evolutionary Computation (CEC), 18–23 July 2010, pp. 1–7.

Wenqing, Z., Zili, Z., 2005. An email classification model based on rough set theory. In: Proceedings of the 2005 International Conference on Active Media Technology, 2005 (AMT 2005), 19–21 May 2005, pp. 403–408.

Wie, W., Zhang, P.T., Tan, Y., He, X., 2011. An immune local concentration based virus detection approach. J. Zhejiang Univ. – Sci. C 12 (6), 443–454.

Xin Y., Fengbin Z., Liang X., Dawei W., 2010. Optimization of self set and detector generation base on real-value negative selection algorithm. In: 2010 International Conference on Computer and Communication Technologies in Agriculture Engineering (CCTAE), 12–13 June 2010, pp. 12–15.

Yevseyeva, I., Basto-Fernandes, V., Ruano-Ordás, D., Méndez, J.R., 2013. Optimising anti-spam filters with evolutionary algorithms. Expert Syst. Appl. 40 (10), 4010–4021.

Yildiz, A.R., 2013a. A new hybrid differential evolution algorithm for the selection of optimal machining parameters in milling operations. Appl. Soft Comput. 13 (3), 1561–1566.

Yıldız, A.R., 2009. An effective hybrid immune-hill climbing optimization approach for solving design and manufacturing optimization problems in industry. J. Mater. Process. Technol. 209 (6), 2773–2780.

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

Yildiz, A.R., 2013c. Hybrid Taguchi-differential evolution algorithm for optimization of multi-pass turning operations. Appl. Soft Comput. 13 (3), 1433–1439.

Yildiz, A.R., 2013d. A new hybrid artificial bee colony algorithm for robust optimal design and manufacturing. Appl. Soft Comput. 13 (5), 2906–2912.

Yuebing C., Chao F., Quan Z., Chaojing T., 2010. Negative selection algorithm with variable-sized *r*-contiguous matching rule. In: 2010 IEEE International Conference on Progress in Informatics and Computing (PIC), 10–12 December 2010, pp. 150–154.

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

Zhang, P.T., Wang, W., Tan, Y., 2010. A malware detection model based on a negative selection algorithm with penalty factor. Sci. China Inform. Sci. 53 (12), 2461–2471.

Zhang, Y., Wu, L., Xia, F., Liu, X., 2010. Immunity-based model for malicious code detection. In: Lecture Notes in Computer Science, vol. 6215. Changsha, pp. 399–406.

csmining.org/index.php/ling-spam-datasets (cited: available from: csminingorg/indexphp/ling-spam-datasets).

spamAssassin: The Apache SpamAssassin Project, 〈http://spamassassin.apache.org/〉 (cited: Available from: 〈http://spamassassinapacheorg〉).