



Article

An Intelligent Spam Detection Model Based on Artificial Immune System

Abdul Jabbar Saleh, Asif Karim, Bharanidharan Shanmugam , Sami Azam * ,
Krishnan Kannoorpatti, Mirjam Jonkman and Friso De Boer

College of Engineering, IT and Environment, Charles Darwin University, Casuarina, NT 0810, Australia; engrajabbar@live.com (A.J.S.); asif.karim@cdu.edu.au (A.K.); bharanidharan.shanmugam@cdu.edu.au (B.S.); krishnan.kannoorpatti@cdu.edu.au (K.K.); mirjam.jonkman@cdu.edu.au (M.J.); friso.deboer@cdu.edu.au (F.D.B.)

* Correspondence: sami.azam@cdu.edu.au

Received: 31 May 2019; Accepted: 9 June 2019; Published: 12 June 2019



Abstract: Spam emails, also known as non-self, are unsolicited commercial or malicious emails, sent to affect either a single individual or a corporation or a group of people. Besides advertising, these may contain links to phishing or malware hosting websites set up to steal confidential information. In this paper, a study of the effectiveness of using a Negative Selection Algorithm (NSA) for anomaly detection applied to spam filtering is presented. NSA has a high performance and a low false detection rate. The designed framework intelligently works through three detection phases to finally determine an email's legitimacy based on the knowledge gathered in the training phase. The system operates by elimination through Negative Selection similar to the functionality of T-cells' in biological systems. It has been observed that with the inclusion of more datasets, the performance continues to improve, resulting in a 6% increase of True Positive and True Negative detection rate while achieving an actual detection rate of spam and ham of 98.5%. The model has been further compared against similar studies, and the result shows that the proposed system results in an increase of 2 to 15% in the correct detection rate of spam and ham.

Keywords: spam; ham; phishing; anomaly detection; Negative Selection

1. Introduction

Email has been an extremely important medium of communication for quite some time now, allowing almost instant reachability to any part of the world with internet connectivity. As stated by Tschabitscher [1], almost 5 billion email accounts were actively in use in 2017, and this is expected to grow to over 5.5 billion by the end of 2019. Tschabitscher [1] also highlights the fact that though more than 270 billion emails are exchanged daily, approximately 57% of these are just spam emails [1]. There are a number of existing machine learning methods as well as techniques that closely resemble biological immune systems to filter spam or phishing emails but their performance has been a major concern. Most of the techniques manage to successfully thwart spam but the tradeoff is that they also block some of the non-spam emails, known as ham. This is a problem, as it could lead to the loss of important information for the user.

1.1. Common Threats

Users worldwide are constantly bombarded by various types of email attacks, such as email spoofing, phishing, and variants of phishing, such as spear phishing, clone phishing, whaling, covert redirect etc. Email spoofing often involves forging the email header (The *From* part) so that the message appears to have been sent from a legitimate user. Email spoofing is a ploy used in spam campaigns because people tend to open an email when they think it is sent by someone they recognize [2]. Email

phishing is a form of spoofing which deceives the user with messages which appear legitimate [3]. Malicious attackers have also attempted to hide the text behind images to defeat anti-spam programs. It is an obfuscation method by which the text of the message is stored as a JPEG or GIF image and displayed in the email. This prevents text-based spam filters from detecting and blocking spam messages.

1.2. Architecture of an Email

Emails are composed of headers and the body of the email. First there is the TCP/IP Header, containing source and destination IP address, then the SMTP envelop, holding the email transactional fields, and source and destination email addresses (but this part is not seen through to the email clients); and then the SMTP headers, where different visible email parts (accessed by the email clients to relate the information to the user), such as 'subject', 'from' and 'to' fields reside. As mentioned above, this is the section which gets tinkered with by the fraudsters, as the actual source and destination emails are kept in the SMTP envelop, which is not directly visible by the user. Finally, there is the body of the email—which is the email message, optionally containing links and attachments, which can be of a malicious nature. Bulk email is not only irritating for everyday email users but it also creates a major computer security problem that costs billions of dollars in productivity losses [4]. In addition, it is also the primary medium for phishing [5,6], as well as for the spread of malicious software, such as viruses and worms [4].

1.3. Complexities Caused by Spam Emails

As indicated above, spam emails can have multifarious problematic effects on individuals, organizations, and the population in general. Leung and Liang [7] noted that phishing alerts often bring about a substantial negative return on stocks. Other researchers have described the negative effects on companies whose legitimate email messages were considered spam by anti-spam systems [8]. Spam emails can cause significant reputational damage, as well as an individual's identity theft, through the installation of malicious attachments; the stolen information can later be used to harass the victims [9]. Botnets [10] can also spread through spam emails. High profile cases of companies which are victimized have become regular occurrences. In 2018, sensitive financial information of employees of a United States Bus Company had fallen into the hands of scammers, due to a phishing email scam [11]. Then, a recent whaling attack cost the French cinema chain 'Pathé', also in 2018, over USD 21 Million [12]. These are just a few of examples of the widespread problem of spam emails.

1.4. Shortcomings of Non-Automated Spam Filtration Methods

There are a number of non-automated spam identification frameworks available which do not rely on machine learning principles but these systems face considerable bottlenecks in tackling contemporary spam attack patterns and dynamism. In this section, we will briefly highlight some of the shortcomings in these systems.

The blacklisting of sender addresses has been a popular choice over the years. But this system alone (primary\lone defense against spam emails) has proven to be insufficient, as the spammers are adept at changing the sending address and the database update process is often slow [13].

The heuristic approach is another popular technique, where a set of rules is applied to incoming emails to mark them as spam or ham. Regular expressions are often used to construct the rule set. However, if the scammers manage to get access to the ruleset, they can quite conveniently prepare their messages beforehand to avoid the filtering system.

Other known frameworks are keyword matching, country-based filtering, and greylisting, to name a few, but these also suffer from limitations that modern spam gangs can easily exploit. Another key issue common to almost all of these frameworks is that with the increase in SPAM detection for these systems, the *False Positive* rate (FP) also increases dramatically, which leads to a poor overall performance.

1.5. The Benefits of Our Proposed Machine Learning Based Approach

Adoption of machine learning to detect the changing nature of the problem has been quite promising in recent times. Machine learning approaches are currently being used with newer technologies such as Blockchain [14,15] as well as with more traditional sectors [16]. Machine learning based filters are able to adapt themselves to the changing nature and behavior patterns of the spammers. As spammers always tend to introduce subtle changes and design new ways of spreading spam emails as widely as possible, this is a critical advantage. Static filters, which are not able to detect these subtle changes, eventually fail against even a slightly modified spam patterns.

Machine learning based techniques can be divided into two basic categories: Unsupervised and Supervised (the majority). Supervised learning models require a training set of samples which have been previously labelled. These methods perform better with the availability of more training instances but this requires a considerable amount of prior labelling work. In the gathering phase, as many emails as possible are usually collected. However the labelled training instances are infrequent and this slows down the 'sharpening', or, to put it in a simpler way, the 'becoming more intelligent' of the anti-spam systems.

In light of these difficulties, we propose a novel hybrid technique of a supervised approach with a modified machine learning technique inspired by the human immune system. It is called Negative Selection Algorithm (NSA) and uses different detectors in order to successfully detect spam emails with a high degree of confidence. The algorithm determines whether an email is spam by evaluating a probable spam word footprint scale as well as by checking a database containing blocked IP addresses and a database of frequently used confirmed spam keywords. We expect our novel approach of hybridization of Machine Learning principles along with NSA can have a significant impact on spam detection and filtration.

2. Related Work

Considerable work has already been done in this field, and due to the importance of the topic, new detection methods are regularly proposed. Nousseir et al. [17] proposed a character-based technique. This approach uses a multi-neural network classifier. Each neural network is trained on the basis of a normalized weight obtained from the ASCII value of the word characters. However, an attacker can camouflage the words, e.g., by writing the words using a slightly different spelling or by using visuals, to circumvent detection. This results in relatively low correct detection rates.

Aski et al. [18] developed a rule based framework, where 23 carefully selected features were identified from a personally accumulated spam dataset. Each of the criteria was then assigned a score. In order to label the email as spam or ham, the accumulated score was compared to a threshold value. Three machine learning principles, Multilayer Perceptron (MLP), Naïve Bayesian Classifier, and C4.5 Decision Tree Classifier were used, but the study was conducted on a limited database of just 750 spam and ham emails. An effective performance measure in terms of time and memory footprint has not yet been described.

Another study proposes that text mining in emails should be done at the term level [19]. The mining process starts by pre-processing the document collection and extracting the relevant terms from the documents. Each document is then represented as a set of terms and annotations characterizing the document. This method gives the number of occurrences of the terms. However, one of the drawbacks is that it cannot handle large texts.

Work has also been done to identify spamming accounts. The ErDOS (Early Detection of Spamming) system, uses an algorithm specifically designed for the early detection of spamming accounts. The detection approach implemented by ErDOS amalgamates content-based detection with features based on inter-account communication patterns [20]. This work could be extended in the future so that it may support the real-time signaling of a spammer's account.

In another study [21] a new hybrid model, combining standard Negative Selection Algorithms, and Differential Evolution, has been suggested. The proposed model has the unique feature of implementing

Differential Evolution in the random generation phase of NSA. The model also maximizes the generated detector distance while minimizing the overlap of detectors [21]. However, this work does not address the issues of image spamming and clickjacking.

Attackers often try to avert word based filtering systems by using character variations to disguise the word, such as spelling “mortgage” as “M*o*r*t*g*a*g*e”. Another common example is the word “Viagra” (‘Vjagra’, ‘Viagrra’, ‘V i a g r a’ or ‘vi<bre/>agra’). This approach limits the effectiveness of most content-based techniques. However, regular expressions (regex), generated manually, can be of great use in identifying messages obfuscated by spammers through varying patterns. In this context, a regular expression is a compact way to depict sets of words or sentences that follow a certain pattern [22]. This can then be locally integrated with a content-based filtering scheme. Ruano-Ordás et al. [22] used such a technique to filter spam messages. They developed a novel genetic programming algorithm, named DiscoverRegex, to automatically generate regular expressions for a given dataset. One potential improvement of this system would be to extract regular expressions from the full text of the messages instead of the currently adopted ‘contents of subject header’ only.

Clickjacking, also known as IFrame Overlay or UI redressing, is a type of attack in which a field or button is overlaid by malicious scripts or links that are not usually visible. The technique has become rather popular amongst the hacker community. It leads the users to clicking on links or buttons which they cannot see, usually because the color of the link is the same as the page background color. Three of the Alexa Top 10 web sites and nearly 70% of the major banking sites have no precautions in place against clickjacking attacks [23]. One readily implementable modification against clickjacking has been to offer a confirmation prompt to users when the target element has been clicked [24]. For confirmation, the user may be asked to mark a checkbox. Another way is to enter a correct CAPTCHA before clicking on any action button intended.

The ‘Phishing Email Detection System’ (PEDS) is a framework based on supervised and unsupervised techniques [25]. It also has elements of reinforcement learning. The system can adapt itself based on detected changes in the environment. The framework works well for Zero-Day phishing attacks but not for general advertisement spams. The engine of the system, the ‘Feature Evaluation and Reduction’ (FEaR) algorithm, dynamically selects and ranks the critical features from emails based on several environmental parameters. However, the selected features are somewhat unconventional and not really sufficient yet.

Zhu and Tan introduced a ‘Local Concentration (LC) based Feature Extraction’ technique for the development of their anti-spam model [26]. It is inspired by the ‘Biological Immune System (BIS)’. The LC approach is capable of determining position-correlated information from a message by converting each area of a message to a corresponding LC feature. The message content is divided using a fixed length sliding window. Feature engineering can help to derive new features from existing ones [27].

The spam filtering implementation discussed by Hayat et al. [28], is based on an improved version of the Naïve Bayes algorithm. It has a better performance than systems based on the standard version of the Naïve Bayes algorithm. The framework compares the content of the current batch of emails to that of earlier ones. If a major change is detected, the model automatically makes the necessary modifications. Once updated, the detection rate of spam improves considerably. An 8–9% increase in accuracy was achieved relative to a multinomial Naïve Bayes algorithm.

Support Vector Machines (SVMs) are widely used and have been proven quite efficient in designing anti-spam systems based on machine learning. The advantage of SVMs over other algorithms is that they can handle high dimensional feature sets which have many attributes [29]. SVMs can also transform non-linearly separable data into new linearly separable data by a procedure called the ‘kernel trick’ [30]. However, Alsmadi and Alhami [31] argued that a lower False Positive rate can be achieved using NGram based clustering and classification.

In his work, Idris [32] developed a new detector model with a set of matching rules using NSA. The system effectively matches self and non-self in order to improve the detector’s performance. The system has achieved an accuracy of 89.29% on a dataset of 4601 instances. However, the model only

uses word-based similarity through Euclidian distance for matching. Taking header information, such as the source IP, into consideration might further improve its performance.

3. Proposed Methodology

The model proposed in this study is trained by developing a memory of the past behavior of spam emails. It does not allow the same type of behavior in future incoming messages, as the model has been inoculated by the user against a particular behavior. This process is called Negative Selection. The design of the Negative Selection algorithm has been based on the self–nonself discrimination behavior of the mammalian acquired immune system [33] as shown in Figure 1.

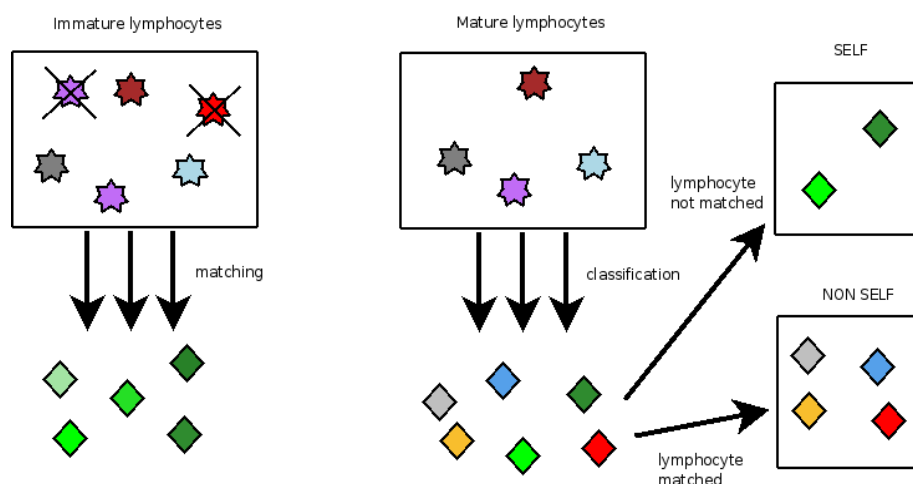


Figure 1. Self and non-self agents [34], Reproduced with permission from Heba Elshandidy, *Geniuses ONLY! Artificial Immune Systems – PART II*, published by Wordpress, 2011.

The principle behind the process via Negative Selection is the anticipation of unfamiliarity or the variation from what is familiar. This is also a key step in anomaly and change detection algorithms. The objective is achieved by developing a model of anomalies, changes, or unfamiliar (non-normal or non-self) data by generating patterns that do not correspond to or match an existing body of available (self or normal) patterns. The prepared non-normal model is then used to monitor existing natural data or new data streams by looking for matches to the non-normal patterns. The Negative Selection or Artificial Selection differentiates between normality and anomaly by having knowledge of self and non-self-behavior. This knowledge can be programmed into a system or developed through a series of learning processes known as training. The process of training can be carried out by ‘Learning’ from the contents of different self and non-self datasets.

Negative Selection Algorithm (NSA) Illustrated

Artificial Immune Systems (AIS) are highly distributed, computationally intelligent methods or systems based on the observation of the behavior and interaction of antigens and antibodies in a biological system. Negative Selection Algorithms (NSA), a sub-area of AIS, imitate the way a human body detects and disposes of harmful antigens. An antigen may be defined as a substance that causes the immune system to produce antibodies against it. An antigen may be a substance from the environment, such as chemicals, bacteria, viruses (non-self-antigen) or it may be produced in the body (self-antigen). The immune system does not recognize the substance, and is consequently trying to eliminate it [35].

The NSA is developed on the basis of the mechanism in the Thymus that induces a set of mature T-cells capable of binding or matching only with non-self-antigens. These T-cells are ‘trained’ in the maturing phase to not interfere with self-cells. The mature T-cells possess a repository of known patterns or of other self-cells which can be used when faced with antigens, or non-self. Remedial

action is taken if something out of ordinary enters into the system or if a break in the expected pattern is detected. In biological systems, antibodies are created to deal with unwanted antigens. The ‘binding’ results in the destruction of the non-self-antigen.

The NSA starts by producing a set of self-strings, S , that define the normal state of the system. The next step is to generate a set of detectors, D , capable of recognizing or binding with the complement of S only, that is S' (non-self). These detectors act in a similar way to mature T-cells which are capable of inducing antibodies soon as a match is found. Binding occurs with the antigen or in this case, spam keywords or blacklisted IPs. The core logic within the detectors works similar to the mechanism of biological antibodies. The algorithm can then be applied to new data in order to separate them into ‘self’ or ‘non-self’.

The model is trained with both spam and ham datasets to build the knowledge base required for intelligent operation. The detectors within react differently when faced with spam keywords (non-self-antigen) than with ham keywords (self-antigen). As mentioned above, the T-cells in the immune system also go through a maturing process to learn how to react when faced with self and non-self-antigens. The immune system has more than one frameworks in place to combat unwanted situations (e.g., the release of an immature T-Cell into the blood stream). The proposed system also follows such a pattern through the implementation of multiple detectors that act as a barrier against spam emails.

4. Design of the Framework

A model is prepared to be trained with self (non-spam or ham) datasets as well as with spam datasets. These datasets have previously been investigated and are labeled either as spam or non-spam. For training and building the spam and non-spam databases, the well-known Enron email datasets are used. The entire raw set can be found in the data repository of Carnegie Mellon University, USA. The six sets used for this study were downloaded from the Department of Informatics, Athens University of Economics and Business, Greece.

The downloaded six datasets are arranged in the following structure:

Each email is passed through the trainer model where it is processed to extract keywords and to categorize its contents. The email addresses are also extracted, as well as the source IP addresses of these emails. At the end of each process, statistics are presented with a list of words and their frequency in the current set of emails. This is recorded in separate databases which can be updated when more datasets become available in the future. It has been observed that more recently updated databases result in a higher detection rate of spam and a lesser number of false positives. Flagging a true spam message as spam is termed True Positive while marking a legitimate message as a spam message is known as False Positive. False positives result in the loss of legitimate emails which is a major concern. To improve the system’s performance and lower the false positive and false negative rate, the model needs to be trained with updated datasets whenever possible so that it is aware of the behavior of newly coming threats. In total 50,409 emails, a subset of Enron email corpus, have been used for training and testing purposes. As outlined in Table 1, 33,792 emails or just under 66%, have been used for training purposes. The remaining 17,157 emails or just over 34%, have been used as the test dataset.

Table 1. Enron email dataset.

Dataset	Number of Spam	Number of Ham
Enron ₁	1513	3735
Enron ₂	1496	4361
Enron ₃	1500	4012
Enron ₄	4500	1500
Enron ₅	3675	1500
Enron ₆	4500	1500
Total	17,184	16,608

4.1. Training Phase

In the training phase the model is trained with both ham and spam Enron datasets. In the following section each segment of the process has been outlined with an appropriate pseudocode.

- To begin with, the datasets need to be identified by the framework as either HAM or SPAM. *Corresponding pseudocode:*
 1. Input: SPAM and HAM database
 2. Define Spam and Ham as containers for SPAM (non-self) and HAM (self) categories of datasets.
- Once the datasets have been appropriately marked, the features of SPAM or HAM emails need to be identified. Words and keywords along with the frequency of their occurrence are stored in appropriate databases. In the case of spam emails, certain word-combinations are also stored.

Corresponding pseudocode:

Read SPAM or HAM datasets

- a. **For** input $i = 1$ to length of datasets:
 - b. **Read** email_(i)
 - c. Sample words and keywords
 - d. Record words along with their frequency
 - e. If a word or combination of consecutive words (a keyword) has appeared before, then update its frequency in the SPAM and HAM databases
 - f. **EndFor**
- Repeat step 1 (a–f) whenever new datasets are available to further train or update the database.

Corresponding pseudocode:

Read SPAM datasets

- a. **For** input $i = 1$ to length of datasets:
 - b. Select the words or combinations
 - c. Check Token Database for the selection at 1.b.
 - d. **If** found == TRUE
 - e. **Goto** step 1.a.
 - f. **Else**, frequency is greater than set threshold
 - g. copy the word to Token Database
 - h. **EndIf**
- i. **EndFor**

Figure 2 is a graphical illustration of the whole process.

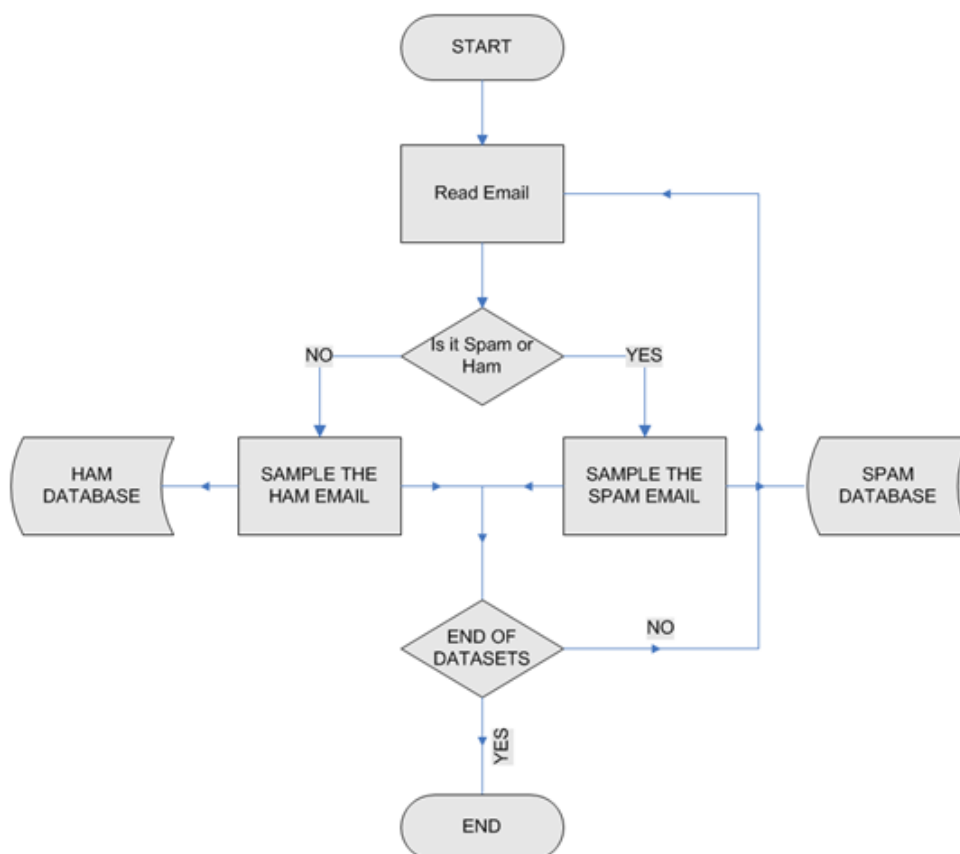


Figure 2. A flowchart of the overall design.

4.2. Enron Dataset Findings

Some of the keywords of interest which have been found after completing the training steps above for the Enron datasets and are presented in Table 2.

Table 2. A sample of spam keywords and their frequency in the Enron spam dataset.

No	Key	Frequency
1	Viagra	1177
2	Sex	333
3	xanax	262
4	vicodin	143
5	discreet	109
6	casino	86
7	teen	69
8	unsolicited	27
9	Removal instructions	20
10	babes	12
11	financial freedom	8
12	offer expires	5
13	make money	4
14	work from home	2
15	mortgage rates	1
16	free gift	1

All these words are spam keywords, tokens that spammers had used in their emails to sell their products or attract the user for various reasons. They are recorded in the token database and are

used to make a decision whether new emails are spam or ham. If the email passes this step, further processing is done to determine its authenticity as illustrated in the next section.

4.3. Structure of the Datasets

The Enron email dataset is one of the very few collections of organizational emails that are publicly available.

An Enron email message, found in the original corpus, is composed of the following headers in order (the header field in parenthesis is optional): "Message-ID", "Date", "From", ("To"), "Subject", ("Cc"), "Mime-Version", "Content-Type", "Content-Transfer-Encoding", ("Bcc"), "X-From", "X-To", "X-cc", "X-bcc", "X-Folder", "X-Origin", and "X-FileName". A blank line separates the email content. If the signature and quotation are available, they are also included. Header field names starting with an "X-" indicate that the field values are from the original email message. The field values of "From", "To", "Cc" and "Bcc" are converted to "X-From", "X-To", "X-Cc" and "X-Bcc" accordingly [36].

5. Detection Stages

Every email must pass through the following detection stages to be regarded as legitimate.

5.1. Blocked IP Database

The Blocked IP Database is a list of all blocked IP addresses. Any email originating from any of these IP addresses will immediately be marked as SPAM.

5.2. Token Database

As discussed above, the token database is manually updated and stores keywords which have already been flagged as spam or non-self. Any email containing even a single keyword from this database will be marked as spam. Some of the words or word combinations found in this database are *hidden assets*, *asian babes*, *for just \$xxx*, *accept credit cards*, *bank account*, *xanax*, *50% off*, *act now! don't hesitate!*, to mention a few. There are a number of word variations and different individual words that can be found in this database

5.3. SPAM and HAM Database

If an incoming email passes through the two detectors above, the SPAM and HAM databases, are used to confirm the email's legitimacy. Every email is scanned line by line to determine the status of each word and investigate whether they belong to the SPAM database or the HAM database.

If the majority of words in a line belong to the SPAM database that line is marked as a spam line, otherwise it is considered a HAM line. After all the lines have been processed, a decision is made based on how many SPAM lines have been found in the email. Currently a 30% threshold level is being used, i.e., if 30% of lines are spam lines then the email is marked as spam.

6. Email Scanning Algorithm

This section describes the actual algorithmic steps that have been developed to realize the framework.

- The email is preprocessed to be checked against the Token Database, the Blocked IP database, and finally the SPAM database.
- To reduce the processing time some of the helping verbs, such as am, is, are, was, were, be, being, been, do, did, didn't etc. have been removed.
- The whole email is then converted into lower case to reduce the complexity of checking the words in all their possible combinations of upper and lower case. For instance, one of the most common spam words 'viagra' can come in many varieties such as Viagra, vIagra, viaGRa, etc. In fact there could be 64 possible forms (2^6) of the word Viagra. A human can process all those different

forms and usually interprets them similarly but for the computer these are all different words. Converting to lower case reduces the processing overhead and improves the time efficiency.

- Extra spaces are also removed along with some special characters to reduce extra checks as explained above, e.g., 'Viagra', 'Viagra', 'Viagra', 'Viagra', etc., they are all different words to the computer. Hence, removing leading and trailing spaces and lowering the case will reduce processing complexity.
- The email is evaluated using three different detectors. The first one identifies whether the source IP address matches to that of the blacklisted IP address or if anywhere in the content of the email body, such an IP address is mentioned.
- In the second detector, words within the email body is matched against a pre-designed confirmed spam token database, and if a word does get matched, then the email is flagged as spam.
- In the third detector, each line of the email is evaluated to determine if it contains any spam word from a SPAM dataset built from training the confirmed spam emails. If 30% of the lines of the email contains any spam word, then the email is flagged as spam email.

The following is a detailed look into the internals of each of the detectors through which an email is passed.

6.1. Detector # 1—Source Check

The email is checked against the Blocked IP database to check whether it is sent from a blacklisted IP address. Currently the chosen Blacklist database has 36,332 blacklisted IPs; downloaded from www.heise.de. It is an up-to-date collection of spam emitting IP addresses or suspicious sources. If an email is from any of these IPs, it is flagged as spam straightaway. The following algorithm takes a set of blacklisted IP as input and dissects the email header to determine if the source IP matches with any of the blacklisted IP, the algorithm also looks into the email body to check if any mentions of any blacklisted IP can be found. If such a match occurs, the email is filtered out and marked as spam.

Corresponding pseudocode:

1. Input: Blacklisted IP Address collection.
 2. Output: Email flagged as spam or ham.
 3. BEGIN:
 - a. **Input** email;
 - b. **Input** IPs;
 - c. **Let** L = Number of lines in email;
 - d. **For** I = 1 **to** L
 - e. W = Total words in each line;
 - f. **For** J = 1 **to** W
 - g. **If** match (word, IP list) == TRUE
 - h. Mark as spam;
 - i. **Else**
 - j. Mark as ham;
 - k. **EndIf**
 - l. **EndFor**
 - m. **EndFor**
- END

6.2. Detector # 2—Token Database

This phase starts off by checking each word of the email against the blacklisted Token Keys. If any word is found to be from the Token Keys, the email is flagged as spam without any further processing.

In other words the email is marked as spam if only a single word is found in the Token database. However this rule can be tuned and configured as needed, for example instead of a single word, we may tolerate up to three words before an email is labelled as spam. To do that, a loop can be included that will hold step f to k as the loop body, and will run thrice to check if three spam words can be detected from the entire body of the email, before finally labelling an email as spam or ham. It is also possible that it is not a single word which may be spam but a combination of words, e.g., 'free' is not a spam word but 'free credit' is a considered a spam key and '100% free' is a spam key without any doubt. To deal with this situation, the whole email is scanned for such combinations. These combinations are mapped directly to the Token Database and the whole email is searched for any occurrences.

Corresponding pseudocode:

1. Input: Token Database.
 2. Output: Email flagged as spam or ham.
 3. BEGIN:
 - a. **Input** email;
 - b. **Input** IPs;
 - c. **Let** L = Number of lines in email;
 - d. **For** I = 1 **to** L
 - e. W = Total words in each line;
 - f. **For** J = 1 **to** W
 - g. **If** match (word, Token Keys) == TRUE
 - h. Mark as spam;
 - i. **Else**
 - j. Mark as ham;
 - k. **EndIf**
 - l. **EndFor**
 - m. **EndFor**
- END

It is to be noted that although Detector 2 and 3 were programmed in one single algorithm, they have been presented in to different sections for the purpose of clarity.

6.3. Detector # 3—SPAM and HAM Database

It is not possible to have a complete list of current spam words or word combinations against which each email can be checked to determine its authenticity. As spammers keep changing the format and text frequently, the spam databases may not be able to keep pace. To deal with this problem it is necessary to frequently train SPAM and HAM databases with datasets of confirmed spam emails. Currently, as mentioned before, the "Enron" email datasets are being used for training the SPAM and HAM models of the developed algorithm. Spam emails from different origins can also be used to train the model.

Hence, in Detector # 3, each email is also checked against a trained model to make a decision whether it is a SPAM or non-SPAM email. The email is scanned line by line and if 30% of the lines are found to be of the pattern of spam emails then the email is marked as spam. The current threshold is 30% but it may be modified if it further improves the performance.

Corresponding pseudocode:

1. Input: SPAM Database;
2. Input: HAM Database.
3. BEGIN:
4. **Input** email;
5. **Let** L= Number of lines in the email
6. **For** K = 1 to L
 7. **Let** NW = Number of words in the current Line
 - a. **For** i = 1 to NW
 - b. **If** word(i) \in SPAM
 - c. Check ρ_{si} for word_(i) being a spam word;
 - d. **EndIf**;
 - e. **If** word(i) \in HAM
 - f. Check ρ_{hi} for word_(i) being a ham word;
 - g. **EndIf**;
 - h. **If** $\rho_{si} > \rho_{hi}$
 - i. word_(i) is a SPAM word.
 - j. **Else** word_(i) is a HAM word
 - k. **EndIf**
 - l. Record word_(i) and its status, i.e., SPAM or HAM
 - m. **EndFor**
 - n. Check how many words in current line are spam. If over 30% then the line is a SPAM line.
8. **EndFor**
9. Check how many line are found as SPAM lines in the whole email. If more than 30% lines are spam, then current email is a SPAM email otherwise a HAM email.

Here NW is the number of words in a line, ρ_s and ρ_h are probabilities which are approximated as the frequency of the word_(i) in the SPAM and HAM learning phase. Now, as can be seen from step **h** to **k**, if ρ_s is greater than ρ_h , the word is marked as spam. Otherwise, it is marked as ham. The status is also recorded for future reference. If a single word from a line is marked as spam in this fashion, the whole line is considered as a line containing spam words. All the lines of the email are checked for such occurrences and if at least 30% of the lines are marked as spam then the message is considered a spam email.

7. Results and Discussion

As can be seen from Figures 3 and 4, as well as Table 3, training the algorithm with more datasets improves the performance by reducing False Positives, *FP* and False Negatives, *FN*. A total of 17,157 emails from the Enron datasets were scanned, as shown in Table 3. With just Enron₁, 15,981 emails were detected accurately (True Positive, *TP* combined with True Negative, *TN*), while 1176 emails were identified wrongly. However, as can be seen from the same table, the subsequent addition of more datasets increased to proportion of emails that were correctly classified. After the addition of the final dataset, Enron₆, 16,912 emails were detected appropriately. In the context of this study, True Positive (*TP*) is the actual spam emails while True Negative (*TN*) is the actual ham emails.

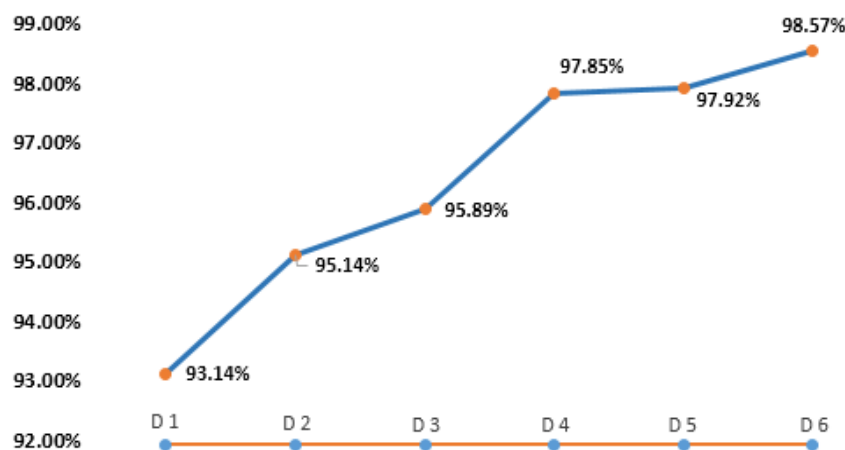


Figure 3. Progressive increase in the correct detection rate.

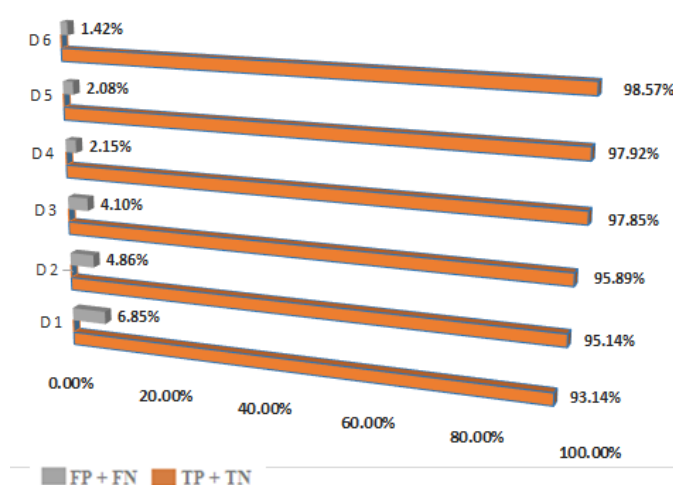


Figure 4. Comparison between (False Positive, FP + False Negative, FN) and (True Positive, TP + True Negative, TN) percentages.

Table 3. Progressing enhancement with the introduction of additional datasets

Datasets	Trained by Number of Datasets	Total Email Scanned: 17,157			
		TP + TN	FP + FN	TP + TN Percentage	FP + FN Percentage
Dataset ₁	<i>Enron</i> ₁	15,981	1176	93.14	6.85
Dataset ₂	<i>Enron</i> ₁ + <i>Enron</i> ₂	16,323	834	95.14	4.86
Dataset ₃	<i>Enron</i> ₁ + <i>Enron</i> ₂ + <i>Enron</i> ₃	16,453	704	95.89	4.10
Dataset ₄	<i>Enron</i> ₁ + <i>Enron</i> ₂ + <i>Enron</i> ₃ + <i>Enron</i> ₄	16,789	368	97.85	2.15
Dataset ₅	<i>Enron</i> ₁ + <i>Enron</i> ₂ + <i>Enron</i> ₃ + <i>Enron</i> ₄ + <i>Enron</i> ₅	16,801	357	97.92	2.08
Dataset ₆	<i>Enron</i> ₁ + <i>Enron</i> ₂ + <i>Enron</i> ₃ + <i>Enron</i> ₄ + <i>Enron</i> ₅ + <i>Enron</i> ₆	16,912	245	98.57	1.42

As shown in Figure 3, the model had a correct detection rate (True Positive + True Negative) of 93.14% during the initial run using only one dataset. After training with all six sets, this soared to 98.57%. With the introduction of the second dataset, D2, the improvement already become noticeable. D3 enhanced the correct detection rate slightly, but dataset 4 still had a major impact and the True rate rose to 97.85%. Adding dataset 5 resulted in a marginal improvement of 0.07% but the inclusion of the final dataset, Enron₆, or D6 in the context of the figure, increased the detection rate to 98.57%. It is

expected that with the addition of more datasets in the training phase, the framework will become even more accurate.

Figure 3 presents both true and false percentages. This framework results in a considerably combined lower False Positive (FP) and False Negative (FN) rate than many other proposed frameworks. Besides being able to detect common spam words, the mechanism is also able to catch double or triple word spam phrases as well as word obfuscation. It can also be observed from Figure 4 that while the combined True Positive (TP) and True Negative (TN) percentages increased gradually, the combined False Negative and False Positive percentages declined.

Figure 5 graphically illustrates the relative contribution of each of the detectors in correctly classifying an email as spam or ham starting from Enron₁ to Enron₆. The first detector checked the source IP address and its presence in the email body or header. The second one matched email body words to that of a pre-defined spam token dataset, while the final detector evaluated whether at least 30% of the lines of the email body contained any spam words. As can be seen, with the addition of more datasets, the relative contribution of detector 1 and 2 increased, while the opposite was true for detector 3. This signifies that with the increased training of the model, most of the spam emails were caught at the first two levels of detection.

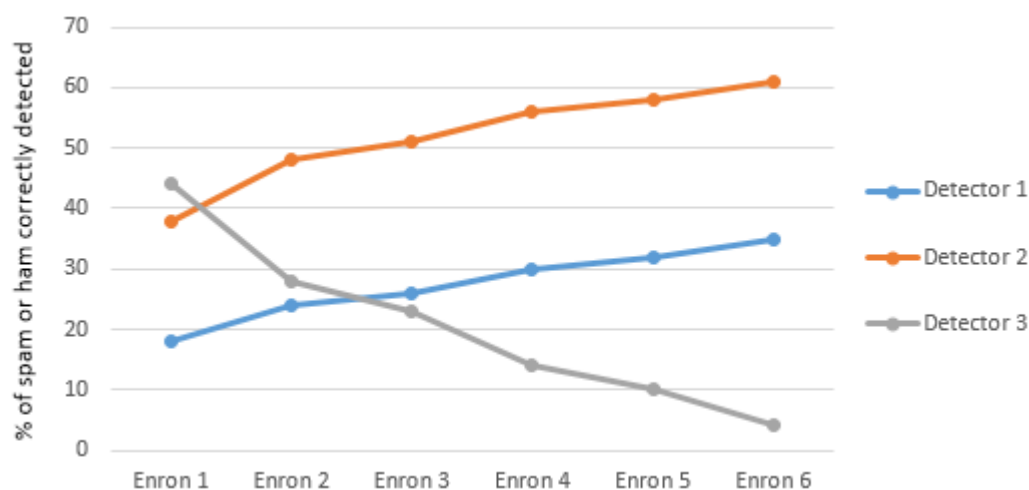


Figure 5. The relative contribution of each of the detectors in identifying spam or ham (True Positive and Negative) upon addition of more datasets.

Figure 6 demonstrates the result of benchmarking the proposed framework against similar studies done by other researchers [21,32,37,38] in terms of correct detection rate of spam and ham. Idris [32] experimented with a standalone NSA model, while swarm intelligence has been included with NSA to develop a filtering system by Idris et al. [38]. In the other two studies [21,37], standard NSA was used in conjunction with Differential Evolution and Fruit Fly Optimization algorithms were utilized respectively. It can be observed that our proposed model attains a higher correct detection rate of spam and ham compared to other similar frameworks.

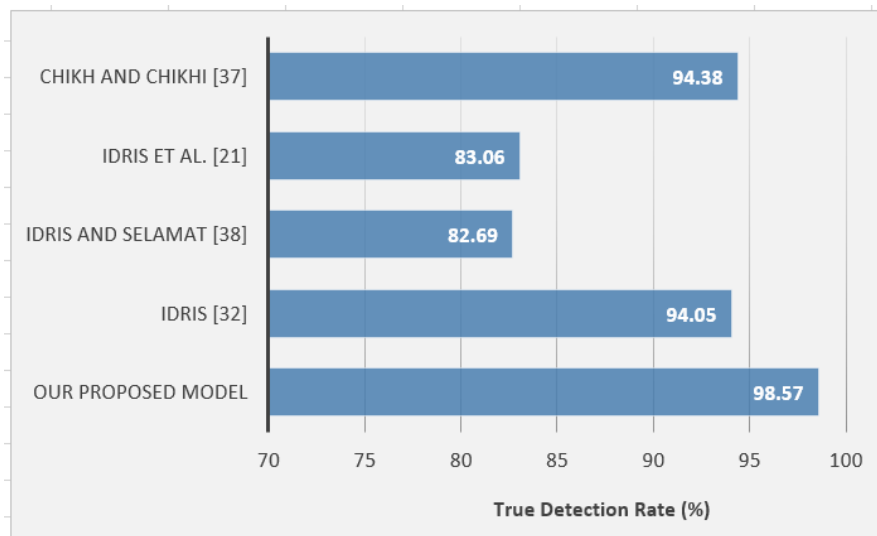


Figure 6. A benchmark comparison with similar studies.

8. Future Endeavors

Future versions of this spam filtering system could include a comparison of a Naïve Bayes algorithm and the proposed model. The solution could be fine-tuned and optimized even further with the inclusion of additional machine learning principles based on deep learning methods. If ρ_s and ρ_h are the approximate frequency of the word_(i) in SPAM and HAM as used in Section 6.3, the probability that word_(i) would be spam can then be calculated using Naïve Bayes theorem as:

$$\rho_i = \frac{(\rho_{si})}{(\rho_{si} + \rho_{hi})} \quad (1)$$

By following the Paul Graham's formula [39], the overall probability that a message or an email is spam or ham can be calculated by combining the probabilities as:

$$\rho = \frac{(\rho_1 \rho_2 \dots \rho_N)}{\rho_1 \rho_2 \dots \rho_N + (1 - \rho_1) + (1 - \rho_2) \dots + (1 - \rho_N)} \quad (2)$$

where ρ denotes the probability that the suspect message is spam. If this is less than the threshold, the email is marked as ham, otherwise it is marked as spam.

9. Conclusions

Spam is a serious issue that is not just annoying to the end-users, but also financially damaging and a security risk. Machine learning algorithms and processes have been found to be quite effective. The work presented in this paper, based on anomaly detection systems and machine learning principles, demonstrates that the inclusion of more databases considerably increases the correct detection rate, from 93.14% based on one dataset to almost 98.57% after including the last dataset (Enron₆). The amalgamation of IP based filtering with a Negative Selection Algorithm in a supervised approach is a novel approach.

We have also compared our proposed system to other systems using NSA and found that our proposed system outperforms other applications of NSA in terms of actual spam and ham detection.

Author Contributions: Conceptualization, A.J.S.; methodology, A.J.S., S.A., and B.S.; validation, A.K., F.D.B., K.K., and S.A.; writing—original draft preparation, A.J.S.; writing—review and editing, A.K., S.A., M.J., and B.S.; visualization, A.K.

Funding: This research received no external funding.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Tschabitscher, H. How Many Emails Are Sent Every Day. 2015. Available online: <https://www.lifewire.com> (accessed on 11 June 2019).
2. Gupta, S.; Pilli, E.S.; Mishra, P.; Pundir, S.; Joshi, R.C. Forensic Analysis of Email Address Spoofing. In Proceedings of the 5th International Conference on Confluence 2014: NGIT Summit, Noida, India, 25–26 September 2014; pp. 898–904.
3. Smadi, S.; Aslam, N.; Zhang, L. Detection of Phishing Emails Using Data Mining Algorithms. In Proceedings of the 9th International Conference on Software, Knowledge, Information Management and Applications, Kathmandu, Nepal, 15–17 December 2015; p. 4.
4. Bratko, A.; Filipic, B.; Cormack, G.; Lynam, T.; Zupan, B. Spam filtering using statistical data compression models. *J. Mach. Learn. Res.* **2006**, *7*, 2673–2698.
5. Jagatic, T.; Johnson, N.; Jakobsson, M.; Menczer, F. Social Phishing. *Commun. ACM* **2007**, *50*, 94–99. [CrossRef]
6. Shan, T.L.; Narayana, G.; Shanmugam, B.; Azam, S.; Yeo, K.C.; Kannoorpatti, K. Heuristic Systematic Model Based Guidelines for Phishing Victims. In Proceedings of the IEEE Annual India Conference, Bangalore, India, 16–18 December 2016; pp. 1–6.
7. Leung, C.; Liang, Z. An Analysis of the Impact of Phishing and Anti-Phishing Related Announcements on Market Value of Global Firms. Master's Thesis, HKU, Pok Fu Lam, Hong Kong, 2009.
8. Raad, N.; Alam, G.; Zaidan, B.; Zaidan, A. Impact of spam advertisement through e-mail: A study to assess the influence of the anti-spam on the email marketing. *Afr. J. Bus. Manag.* **2010**, *4*, 2362–2367.
9. Al-Sharif, S.; Iqbal, F.; Baker, T.; Khattack, A. White-Hat Hacking Framework for Promoting Security Awareness. In Proceedings of the 8th IFIP International Conference on New Technologies, Mobility and Security (NTMS), Larnaca, Cyprus, 21–23 November 2016.
10. Ghafir, I.; Prenosil, V.; Hammoudeh, M.; Baker, T.; Jabbar, S.; Khalid, S.; Jaf, S. BotDet: A System for Real Time Botnet Command and Control Traffic Detection. *IEEE Access* **2018**, *6*, 38947–38958. [CrossRef]
11. Foley, C. ABC Bus Companies, Inc.—Cyber Incident Notification. 2018. Available online: <https://www.doj.nh.gov/consumer/security-breaches/documents/abc-bus-20180302.pdf> (accessed on 24 May 2019).
12. French Cinema Chain Fires Dutch Executives Over CEO Fraud. Available online: <https://www.bankinfosecurity.com/blogs/french-cinema-chain-fires-dutch-executives-over-ceo-fraud-p-2681> (accessed on 25 May 2019).
13. Laorden, C.; Ugarte-Pedrero, X.; Santos, I.; Sanz, B.; Nieves, J.; Bringas, P.G. Study on the effectiveness of anomaly detection for spam filtering. *Inf. Sci.* **2014**, *277*, 421–444. [CrossRef]
14. Khan, M.I.; Faisal, F.; Azam, S.; Karim, A.; Shanmugam, B.; Boer, F.D. Using Blockchain Technology for File Synchronization. In Proceedings of the IOP Conference Series: Materials Science and Engineering, Moscow, Russia, 15–16 November 2018; IOP: London, UK, 2019. in press.
15. Vokerla, R.R.; Shanmugam, B.; Azam, S.; Karim, A.; Boer, F.D.; Jonkman, M.; Faisal, F. An Overview of Blockchain Applications and Attacks. In Proceedings of the International Conference on Vision Towards Emerging Trends in Communication and Networking (ViTECoN), Tamil Nadu, India, 30–31 March 2019.
16. Hoon, K.S.; Yeo, K.C.; Azam, S.; Shanmugam, B.; Boer, F.D. Critical Review of Machine Learning Approaches to Apply Big Data Analytics in DDoS Forensics. In Proceedings of the 2018 International Conference on Computer Communication and Informatics (ICCCI), Coimbatore, India, 4–6 January 2018.
17. Nosseir, A.; Khaled, N.; Islam, T. Intelligent Word-Based Spam Filter Detection Using Multi-Neural Networks. *Int. J. Comput. Sci. Issues* **2013**, *10*, 17–21.
18. Aski, A.S.; Sourati, N.K. Proposed efficient algorithm to filter spam using machine learning techniques. *Pac. Sci. Rev. A Nat. Sci. Eng.* **2016**, *18*, 145–149. [CrossRef]
19. Feldman, R.; Fresko, M.; Kinar, Y.; Lindell, Y.; Liphstat, O.; Rajman, M.; Schler, Y.; Zamir, O. Text Mining at the Term Level. In *Principles of Data Mining and Knowledge Discovery*; Springer: Berlin/Heidelberg, Germany, 1998; pp. 65–73.
20. Cohen, Y.; Gordon, D.; Hendler, D. Early detection of spamming accounts in large-Scale service provider networks. *Knowl. Based Syst.* **2018**, *142*, 241–255. [CrossRef]

21. Idris, I.; Selamat, A.; Omatu, S. Hybrid email spam detection model with negative selection algorithm and differential evolution. *Eng. Appl. Artif. Intell.* **2014**, *28*, 97–110. [[CrossRef](#)]
22. Ruano-Ordás, D.; Fdez-Riverola, F.; Méndez, J.R. Using evolutionary computation for discovering spam patterns from e-mail samples. *Inf. Proc. Manag.* **2018**, *54*, 303–317. [[CrossRef](#)]
23. Akhawe, D.; He, W.; Li, Z.; Moazzezi, R.; Song, D. Clickjacking Revisited: A Perceptual View of UI Security. In *Sergey Bratus & Felix*; Lindner, F.X., Ed.; ‘WOOT’, USENIX Association: Berkeley, CA, USA, 2014.
24. Dipti, Y.; Pawade, D.; Lahigude, A.; Reja, D. Review Report on Security Breaches Using Keylogger and Clickjacking. *Int. J. Adv. Found. Res. Comput.* **2015**, *2*, 55–59.
25. Smadi, S.; Aslam, N.; Zhang, L. Detection of online phishing email using dynamic evolving neural network based on reinforcement learning. *Dec. Support Syst.* **2018**, *107*, 88–102. [[CrossRef](#)]
26. Zhu, Y.; Tan, Y. Extracting Discriminative Information from E-Mail for Spam Detection Inspired by Immune System. In Proceedings of the IEEE Congress on Evolutionary Computation, Barcelona, Spain, 18–23 July 2010.
27. Fahim, M.; Baker, T.; Khattak, A.; Shah, B.; Aleem, S.; Chow, F. Context Mining of Sedentary Behaviour for Promoting Self-Awareness Using a Smartphone. *Sensors* **2018**, *18*, 874. [[CrossRef](#)] [[PubMed](#)]
28. Hayat, M.Z.; Basiri, J.; Seyedhossein, L.; Shakery, A. Content-Based Concept Drift Detection for Email Spam Filtering. In Proceedings of the 2010 5th International Symposium on Telecommunications, Tehran, Iran, 4–6 December 2010. [[CrossRef](#)]
29. Byun, H.; Lee, S.W. Applications of Support Vector Machines for Pattern Recognition: A Survey. In *Pattern Recognition with Support Vector Machines*; Lecture Notes in Computer Science; Lee, S.W., Verri, A., Eds.; Springer: Berlin/Heidelberg, Germany, 2002; Volume 2388.
30. Nizamani, S.; Memon, N.; Glasdam, M.; Nguyen, D.D. Detection of fraudulent emails by employing advanced feature abundance. *Egypt. Inf. J.* **2014**, *15*, 169–174. [[CrossRef](#)]
31. Alsmadi, I.; Alhami, I. Clustering and classification of email contents. *J. King Saud Univ. Comput. Inf. Sci.* **2015**, *27*, 46–57. [[CrossRef](#)]
32. Idris, I. Model and Algorithm in Artificial Immune System for Spam Detection. *Int. J. Artif. Intell. Appl.* **2012**, *3*, 83–94. [[CrossRef](#)]
33. Brownlee, J. Clever Algorithms: Nature-inspired Programming Recipes. In *Immune Algorithms*; LuLu.com: Morrisville, NC, USA, 2012; pp. 270–284.
34. Graham, P. A Plan for Spam. 2002. Available online: www.paulgraham.com/Spam.html (accessed on 21 February 2019).
35. Elshandidy, H. Available online: <https://helshandidy.files.wordpress.com/2011/04/negativeselection1.png> (accessed on 11 June 2019).
36. Wanli, M.; Tran, D.; Sharma, D. A Novel Spam Email Detection System Based on Negative Selection. In Proceedings of the Fourth International Conference on Computer Sciences and Convergence Information Technology, Seoul, Korea, 24–26 November 2009; pp. 987–992.
37. Chikh, R.; Chikhi, S. Clustered negative selection algorithm and fruit fly optimization for email spam detection. *J. Ambient Intell. Humaniz. Comput.* **2017**, *10*, 143–152. [[CrossRef](#)]
38. Selamat, I.I.A. A Swarm Negative Selection Algorithm for Email Spam Detection. *J. Comput. Eng. Inf. Tech.* **2015**, *4*, 2.
39. Zhou, Y.; Goldberg, M.; Ismail, M.; Wallace, W. Strategies for Cleaning Organizational Emails with an Application to Enron Email Dataset. In Proceedings of the 5th Conference North American Association for Computational Social and Organizational Sciences, Pittsburgh, PA, USA, 7–9 June 2007.

