International Journal of Computer Networks and Communications Security

VOL. 5, NO. 6, JUNE 2017, 115–125
Available online at: www.ijcncs.org
E-ISSN 2308-9830 (Online) / ISSN 2410-0595 (Print)

IHASS: Intelligent and Hybrid Anti-Spam System

Shubair Abdulla[1] and Altyeb Altaher[2]

[1] College of Education, Instructional and Learning Technology Depart, Sultan Qaboos University, Oman

[2] Faculty of Computing and Information Technology in Rabigh, King Abdulaziz University, Saudi Arabia

[1]shubair@squ.edu.om, [2]aaataha@kau.edu.sa

ABSTRACT

Spam emails are increasing continually and represent real threat to internet community. Email spammers use advanced software techniques to send millions of spam emails across the internet to distribute announcements of services and items to obtain commercial benefits. Spam detection became a challenge and most of the existing spam detection techniques are based on content-based filtering (CBF) which have many drawbacks. This paper aims to solve these CBF drawbacks by introducing designing, implementing, and evaluating a novel CBF approach for automatic detection of spam emails. The proposed approach is denoted as Intelligent and Hybrid Anti-Spam System. Its main theme is the neuro-fuzzy inference engine which has been applied successfully in a wide range of research. Initially, a normalization task is performed to remove any deceptive character from the text body. The preparation stage then examined the content of well-known datasets of spam emails to identify all the phrases that solely identify spam email. Our experiments show that IHASS could achieve very good accuracy level and works stably well.

Keywords: *Spam email detection, content-based filtering, neuro-fuzzy inference engine, kNN-based Evolving Neuro-Fuzzy Inference System.*

1 INTRODUCTION

E-mails are used in many aspects of our life including, education, banking, health and government services. Moreover, email is significantly potential for many businesses [1]. Spam is unsolicited e mail aims to distribute announcements of services and items to obtain commercial benefits, by requesting the user to click on a link and buy an item or service [2]. The spam e-mail is a recent serious problem, which threaten many useful electronic services, it represents 89–92% of the total number of the sent e-mails and it contributes potentially to the consumption of many resources including the internet infrastructure and the user's time [3].

The Spam and Phishing in Q2 of 2015 report [4] announced by Kaspersky Lab, shows a noticeable increase in the exploit of international events in spam e-mails in order to get personal important information and voluntary aids. International events that were mentioned include, the Olympic Games in Rio de Janeiro, earthquake that occurs in Nepal and the election of the president in Nigeria. Moreover, the Spam and Phishing in Q2 of 2015 report stated that 509,905 new masks of phishing URLs were inserted into the databases of the Kaspersky Lab during this period. According to MacAfee, the phishing email has become In 2015 the most serious attack among all the forms of malware, the continuously growing use of phishing as an attacking tool support the believe that phishing email will continue being the most potential access path for malware in the future [5].

Spammers use advanced software techniques to send millions of spam emails across the world by a mouse click and without any cost, this make the majority of email users facing the problems of the spam email [6]. The international economy is potentially affected by spam as shown in different studies. The researchers in [7] explained an analysis of the effect of phishing on the market value of international companies, which explained that phishing alerts cause a potential negative return on stock. Thus, there is an urgent need for effective and efficient anti-spam methods.

Most popular anti-spam methods are based on content-based filtering (CBF) [8, 9]. These methods employ email content features e.g. word frequencies to classify email into legitimate (ham) and illegitimate (spam) emails. Although these anti-spam solutions have presented high level of

classification accuracy, they present also some drawbacks [10]:

1. A CBF anti-spam approach tends to give weak performance for new, zero-day spam emails as it has been learned to only focus on predefined words that are indicate the presence of spam emails. Spammers may use different new words in their spam message.
2. Spammers may defraud the CBF approaches by inserting special characters between letters, i.e. "reply" becomes "rep#l$y". In this scenario, the word that indicates solely the presence of spam may be passed with low frequency.
3. A CBF anti-spam approach causes significant processing costs on computers, as it needs to examine the text body every email received. If an email server comes cross millions of emails, the anti-spam may cause system crash.

This paper is structured as follows: Section 2 presents the literature review; the research methodology is explained in section 3. Section 4 presents the experimental results. Section 5 discusses the experimental results. Finally, we conclude our paper in Section 6.

2 LITERATURE REVIEW

Efficient detection of spam email is an important issue, researchers have proposed different approaches for email spam detection; the proposed approaches are classified into two main categories [8, 11, 12]:

2.1 Content-Based Filtering (CBF)

Content-based filters are able to learn and distinguish between the spam and legitimate email using the email features and machine learning algorithm .CBF is used successfully in many anti-spam solutions [8, 13]. Recent research efforts focus on the improvement of learning algorithms and individual classifier [9] [14]. Support Vector Machine (SVM) has been used in [15] and [16] to for building the Spam filter. Androutsopoulos et. al. suggested a Naïve Bayesian based filter for spam detection. They show that the performance of the suggested filter is better than a keyword-based one [17].

The performance of CBF depends significantly on the used method for features selection. Mendez et. al. in [11] tested five features selection methods with four forms of Naive Bayes classifiers, the results illustrate the selection of the most suitable feature selection method is potentially important to enhance the accuracy of spam detection. Other machine Learning algorithms such as neural networks (ANN), support tree boosting schemes, rough sets (RS), random forests (RF), case-based reasoning (CBR) systems, artificial immune systems (AIS) and different types of the Naive Bayes (NB) algorithm have been used effectively to perform spam filtering [18, 19].

2.2 Collaborative Filtering (CF)

The similarity of characteristics is an important feature of spam email, which can be useful for detecting it effectively. A spam bulk mailing includes many copies of the same unique spam message, each copy sent to another recipient or group of recipients. The multiple copies from one bulk are changed a bit in order to avoid the similarity with each other. Spammers use obfuscation techniques in order to make collaborative spam detection is difficult process. Collaborative filtering depends on exchanging the spam messages information using the network infrastructure. Razor and Pyzor are Internet based groups that share Nilsimsa sums of spam message information Nilsimsa, 2011). The information about the servers which send spam messages through DNS service1 are shared by the DNSBL (DNS-based Black Lists) and DNSWL (DNS-based White Lists) [20].

Damiani et al. [21] proposed a digest-based collaborative spam filtering; the results show that the proposed filter achieved high detection rate and very low false-positives. Digest-based collaborative spam filter generates same digests out of same emails, and uses the generated digests to find out which emails are categorized in the same bulk. The digest queries are sent to a global web-based database to find out the queries which have been matched by the database. The global web-based database is used by Spam Assassin which is open source anti-spam software. Zhou et al. [22] proposed a collaborative spam filter based on peer-to-peer system. It produces multiple digests from each email as fixed length strings, randomly sampled from the email. The proposed filter uses the exact matching rather than similarity matching, which reduces its accuracy and make it easy to be evaded by spam obfuscation techniques. Content-based filter (CBF) and collaborative filter (CF) have some limitations. The performance of CBF is low for new users, as it needs huge number of examples. Also CF archives low performance when classifying new emails that were not rated before.

3 RESEARCH METHODOLOGY

The design and development of the Intelligent Hybrid Anti-Spam System (IHASS) is introduced in this section. Firstly, in the subsequent sections the methodologies of the research are overviewed. Then, the spam datasets employed in this research are described. It also explains the Most Frequent Maximum Phrase (MFMP) and the Term Frequency - Inverse Document Frequency (TF-IDF), the two algorithms that are used in both feature select and extraction and feature pattern construction.

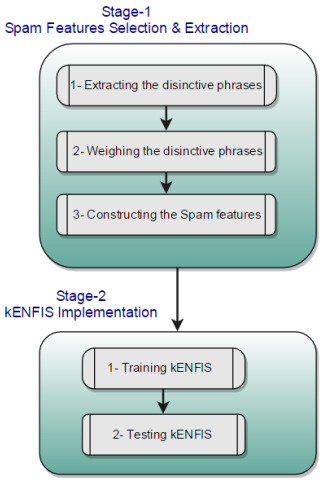

Fig. 1. Overview of IHASS building process

3.1 Overview

IHASS is a neuro-fuzzy inference classifier that automatically detects spam emails. In particular IHASS employs one of well-known neuro-fuzzy inference systems, kNN-based Evolving Neuro-Fuzzy Inference System (kENFIS) [23]. The producing of IHASS involved two phases: training (learning) phase and testing phase. The IHASS learning phase is trained with a set of email features labeled as spam or ham, producing a classifier that can distinguish between spam and ham emails. Prior to the learning phase, the spam features selection and extraction task is completed. This stage involves selecting the spam words, extracting the spam distinctive words, and constructing the spam features. Instead of estimating a database of words that distinguishes the spam emails and to increase the knowledgebase of the classifier, IHASS involved a new algorithm, the MFMP, which used to extract the most frequent phrases from three famous spam datasets. The output of MFMP forms basement for building the neuro-fuzzy inference engine. Figure 1 depicts the IHASS building process.

3.2 Datasets Used

To evaluate the IHASS system, we used three different datasets for spam and ham emails that are publicly available in www.csmining.org and are dedicated for testing spam filtering systems. The dataset are: CSDC2010-Spam, Ling-Spam, and Enron-Spam. The CSDC2010-Spam dataset consists of two parts, training set and testing set. We adopted the training set part which consists of 2949 ham samples and 1378 spam samples. Ling-Spam dataset consist of four directories. We used only the "bare" directory, in which there are 2412 ham samples and 481 spam samples. With regard to the Enron-Spam dataset, version 5 has been adopted. It consists of 1500 ham samples and 3675 spam samples [24-26].

Within the datasets, there is a great imbalance between the ham classes and spam classes. For example, the Ling-Spam dataset contains 2412 ham classes and 481 spam classes. This situation suggests that efficient feature reduction is very important. Furthermore, the total numbers over the datasets of ham classes and spam classes are quite big, so a class reduction process is necessary. To solve these two challenges, we adopted an efficient feature selection and extraction and we also reduced the number of classes. Table 1 shows the number of classes adopted from each dataset.

Table 1: Datasets used

	Samples of datasets			**Adopted samples**		
Dataset	**Spam**	**Ham**	**Total**	**Spam**	**Ham**	**Total**
CSDC2010	2949	1378	4327	200	200	400
Ling	2412	481	2893	180	180	360
Enron	1500	3675	5175	140	140	280

3.3 Spam Features Selection and Extraction

Generally, the feature selection and extraction stage is very crucial for building and testing a classifier. In most cases, the spasm bodies include a great deal of data that might be irrelevant, and has little or no effect on the classification operation quality and accuracy. Such data simply increases the size of the model and the time and resources that are needed for building a classifier. The main goal of spam features selection and extraction stage is to select the relevant spam attributes and to combine the spam attributes into a set of features. In particular, it examines the content of all spam emails in the datasets that are explained in the previous section to select the phrases that solely distinct spam emails and produce the feature patterns. The feature patterns will be the basis on

which the classification operation will be performed.

A: Extracting the Distinctive Phrases

The CBF anti-spam approaches have presented some drawbacks that cause decrease in the level of classification accuracy. These systems are learned often to only focus on a list of predefined words within the spam body. Also they pass illegitimate letters that contain fake words, such as "rep#l$y" word, a fake word of "reply". To overcome the CBF drawbacks, we propose MFMP algorithm. The MFMP algorithm is mainly designed to identify precisely the phrases which are solely distinct the spam emails. In particular, MFMP algorithm maximizes the knowledgebase of IHASS engine through investigating three different and rich datasets, about 5530 spam emails, and selecting the distinctive phrases that the IHASS will depend on in making a classification decision. Moreover, MFMP algorithm is able to extract the most frequent maximum strings over spam emails and ignore characters other than alphabets. Figure 2 depicts the MFMP algorithm.

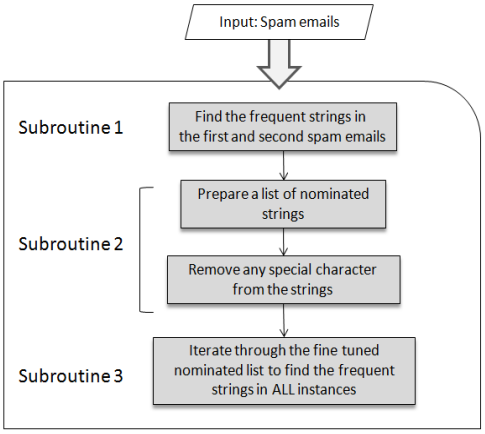

Fig. 2. The MFMP algorithm

Mainly the MFMP consists of three subroutines:

1. **Subroutine 1**: to find frequent phrases in the first spam email and the second spam email in the dataset.
2. **Subroutine 2**: to remove the special characters from the phrases in the list of candidate phrases.
3. **Subroutine 3**: to find the most frequent phrases in all spam emails of the list phrases produced.

Subroutine 1:

We can refer to the spam emails in the dataset, about 5530 emails, as (E_1, E_2, .. E_n). This subroutine aims to nominate a list of strings that are frequent in the first two instances in the dataset: E_1 and E_2. The subroutine compares E_1 with E_2, any string taken from E_1 exists in E_2 will be added to the list. The comparison is done using N-Length parameter. The N-Length parameter represents the initial length of the string in E_1 to search for in E_2. Initially, the value of N-Length is set to 12. The value of 12 is selected based on two rules: (i) the larger the N-Length parameter value, the fewer the number of loops and (ii) a large value of N-Length parameter could miss a valid small strings. Therefore, the value of 12 is selected as it reduces the trade-off between these two rules. Starting from the first character, the subroutine takes 12 characters form E_1 and searches E_2 for any matching. If there is a matching, it stores the 12 characters into the list and tries to find any additional matching. If there are additional matching characters, the subroutine will add the matching characters to the stored characters and adjust the length accordingly. Figure 3 gives an example of the initial length and the adjusted length according to the additional matching characters. The bytes "Approved Fun" is the first matching, but "d Valid" are additional matching. So the length is adjusted to 19 and the "bank loan agreement" will be nominated.

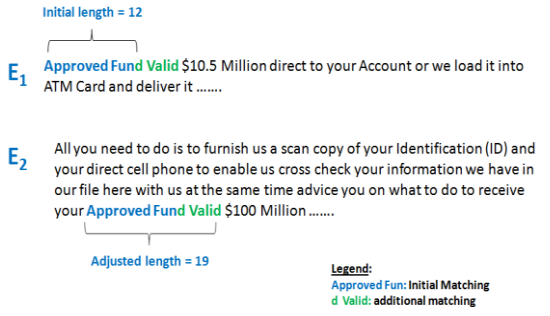

Fig. 3. Initial Length and Adjusted Length

In the next loop, the subroutine will take 12 bytes starting from the 2nd byte in case of no matching resulted from the previous loop; otherwise it will start from 9th byte and so on in the remaining loops until it scans all bytes of E_1. At the end, subroutine 1 produces a list of nominated strings of different lengths.

Subroutine 2

The main goal of this subroutine is to fine-tune the list of nominations. Any string contains special characters will be excluded from the nomination list.

Subroutine 3

Subroutine 3 finalizes the extraction process. It takes the fine-tuned list of nominated strings and identifies the frequent strings. In order to extract the continuous strings and obfuscated strings that split in arbitrary places within the letter text, this subroutine iterates through the list of strings. At each loop, a substring will be identified. Then the system orders the strings starting from the most frequent maximum string and down to the most frequent minimum substring. Ultimately, the system will collect all the parts of distinctive string, either continuous or split, in one string. It is worth mentioning that the greater the number of instances captured, the highest the accuracy of the signature generated. Ultimately, the step of extracting the distinctive words resulted in 188 phrases from the datasets.

B: Weighing the Distinctive Phrases

When the number of extracted features is large, some of which may be redundant or irrelevant, and this introduces several problems such as misleading the detection. Weighing the distinctive words is a process of identifying the level of significance of each phrase to the spam email. A phrase provides low level IDF of significance will be eliminated from the list. For this purpose, we used the most often adopted term weight vector, the TF- [19] to weigh the 188 phrases that have been extracted. The TF-IDF is a statistical measure used to calculate how significant a distinctive phrase is to the spam email. Phrase significance is defined as follows [27]:

$$w_{ij} = tf_{ij} \cdot log \frac{n}{df_i} \dots\dots\dots\dots (1)$$

Where:

- w_{ij}: weight of the *ith* term in the *jth* document,
- tf_{ij}: the number of occurrences of the *ith* term in the *jth* document,
- df_i: the number of spam emails in which the *ith* term occurs, and
- *n*: the total number of documents in the dataset.

Table 2 shows examples of phrases with their TF-IDF on the two datasets. The first column shows the serial number of the email in the CSDC2010-Spam and Ling-Spam datasets, the first 10 spam emails are selected.

C: Constructing Spam Features

Upon identifying the most frequent phrases of spam emails, which could be called spam distinctive phrases, we started the process of feature pattern (FP) construction. This step aimed mainly at preparing the features extracted for implementing kENFIS and at isolating spam and ham emails perfectly by selecting the most significant phrases. By completing this step, the number of phrases is reduced from 188 to 24 phrases, which represent the most significant phrases. This reduction is very important to reduce the overload on the PC systems when executing the classifier. The phrases will highly contribute to the process of construction the spam FPs. That is any email contains one or more than one distinctive phrase will be considered as spam. In order to detect the spam emails that include a distinctive phrase distributed in different places within the email bode, the most frequent phrases in an email must be compared against the predefined distinctive phrases. Therefore, we computed the similarity of the most frequent phrases with the predefined benchmarking spam distinctive phrases. The percentage-of-difference (PD) between them is computed based on the Levenshtein distance (LD), which is used to compute the number of characters that must be replaced, inserted or deleted to obtain the matching [28]. The formula used is:

$$PD = \frac{LD_{s,\bar{s}}}{Len(s)} \times 100 \dots\dots\dots\dots (2)$$

Where $LD_{s,\bar{s}}$: Levenshtein distance, S: the benchmarking distinctive phrase, $\bar{S}$: the most frequent phrase generated by MFMP, and $Len(s)$: length of benchmarking distinctive phrase.

Table 2: Examples of phrases and TF-IDF

Email	Phrase	TF-IDF
Dataset: CSD2010		
1	money-making systems	0.38508
2	money/credit card information	0.27600
3	Join free and Make $1	0.12022
4	change your membership status	0.43998
5	want to see that porno email	0.49552
Dataset: Ling		
1	your phone	0.31443
2	financial independence	0.11011
3	thousands of dollars	0.21113
4	fast financial relief	0.45735
5	best offering for you	0.19291

The FP construction step for each ham and spam email involved the following tasks:

1. Running the MFMP to extract the most frequent phrases in the email body,
2. Computing the similarity between the most frequent phrase and the 24 benchmarking distinctive phrases, the *PD*, and
3. Constructing the FPs from 25 columns. The first column shows the email class, i.e. 0 for ham emails and 1 for spam emails, and the rest columns show the features of the email, the PD values. They represent the measurement of similarity between the most frequent phrase in the email sample and the benchmarking distinctive phrase.

3.4 Implementation of kENFIS

The features selection and extraction operation has revealed that the classes, ham emails and spam emails are not linearly separable in the feature space. That is, one distinctive string or a part of it can exist in two emails that belong to two different classes. This overlapping classification problem calls for the use of fuzzy sets to classify the feature space, so that each email sample may belong to two or more classes with different degrees of membership. Each email sample will be associated to an appropriate class membership value. Due to this fuzzy nature, we adopted the neural networks and the fuzzy clustering to create neuro-fuzzy inference system to establish an appropriate membership. We selected kENFIS as classifier. kENFIS partitions the feature space into clusters by using an enhanced version of kNN classification method, kNN-based evolving fuzzy clustering method (kEFCM). The evolving operation is performed by incremental supervised learning. kENFIS builds up the knowledgebase by integrating the simplicity of k-nearest neighbors (kNN) algorithm with the accuracy of least-square method (LSM) [23].

Typically, a classifier needs to be learned (trained) first on a finite training set. The training phase produces a trained model of the classifier. More training data gives better model generalization. The trained model then has to be tested on a different test data to estimate its classification accuracy. Also, there is a need for criteria to assess the classifier performance experimentally, e.g., error rate, accuracy. The kENFIS training and testing phases are explained in the following section along with the criteria that have been used for the assessing the performance.

4 EXPERIMENTAL RESULTS

We perform experiments to demonstrate the effectiveness of IHASS system in identifying spam emails. The first issue that we took in account is the over-fitting. The over-fitting issue is appeared when IHASS is trained only by the training set to achieve the optimal performance. Consequently, IHASS performance is superior for the training set but may drop dramatically for the testing set [29]. For the purpose of avoiding the over-fitting issue, a 10-fold Cross Validation is implemented for IHASS training. A 10-fold CV firstly partitions the training set into 10 equal-sized subsets and uses 9 subsets for training and the remaining subset for validation to evaluate the accuracy of the CV, and this process repeats for 10 times (folds). Since the subset used for validation in each fold is unknown to IHASS, it is a good benchmark for IHASS after training and thus avoids over-fitting. Table 3 describes the folds for each dataset. In addition to avoiding over-fitting problem, the 10-fold CV technique is used for parameter tuning based on each of the training datasets. The kENFIS system uses one tuning parameter, the number of nearest neighbors. The number of nearest neighbors highly influences the overall accuracy of the classification. Implementing the 10-fold CV technique allowed selecting the best value for this tuning parameter. Four values have been tested: 5, 7, 9, and 11. The value of 7 has been adopted for the nearest neighbors as this value gave the highest accuracy during the test.

Table 3: 10-Folds description

	Training samples			Testing samples		
Dataset	**Spam**	**Ham**	**Total**	**Spam**	**Ham**	**Total**
CSDC2010	20	20	40	180	180	360
Ling	18	18	36	162	162	324
Enron	14	14	28	126	126	252

Two types of measurements have been conducted: measurement of IHASS quality and measurement of IHASS spam detection. We considered the root mean square error (RMSE) and the non-dimensional error index (NDEI) as error performance measures and hence, to give IHASS quality indications.

RMSE is expressed as:

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(Z_i - \acute{Z}_i)^2} \dots\dots\dots\dots (3)$$

And NDEI is calculated as:

$$NDEI = \frac{RMSE}{stdev(Z)} \dots\dots\dots\dots (4)$$

Where n is the total number of samples, Z_i is the desired output, $\acute{Z}_i$ the IHASS output, and *stdev(Z)* is the standard deviation of the desired output.

Tables 4 & 5 summarized the RMSE and NDEI values for the training and testing sets respectively in terms of mean, maximum, minimum, and standard deviation. These statistical measures are calculated for the RMSE and NDEI over the 10-fold CV. The RMSE and NDEI plots of IHASS for the 10-fold CV are shown in Figures 4 and 5 respectively.

Since the main goal of IHASS is to detect spams, we measured the accuracy of IHASS in classifying emails into spams and hams. For each experiment, we registered some important metrics: true positive (tpv), false negative (fnv), true negative (tnv), and false positive (fpv). Assuming that the ham email is positive state and the spam email is negative state, then theses metrics can be defined as follows:

1. tpv: ham and the IHASS classified it ham.
2. fnv: ham but the IHASS classified it spam.
3. tnv: spam and the IHASS classified it spam.
4. fpv: spam but the IHASS classified it ham.

The accuracy of IHASS system, which us the percentage of emails that are classified correctly, is calculated as follows:

$$Accuracy = \frac{(tpv + tnv)}{n} \dots\dots\dots\dots (5)$$

Table 6 summarizes the results of the 10-fold experiments conducted, where the k value represents the value of the IHASS tuning parameter, the number of nearest neighbors. For this kind of study, measuring the tpv, fnv, tnv, fpv, and the accuracy is not enough as the ultimate goal of the testing is not only to determine the number of all emails that are classified correctly, but also the number of spams that are classified correctly. Hence, we calculated other statistical measurements, the precision and the recall. The precision is the percentage of classified emails that are spam and the recall is the percentage of spams that are classified correctly. An accurate algorithm should achieve better precision and maintain maximum recall [30]. The recall and the precision are calculated using following formulas:

$$Precision = \frac{tnv}{(tnv + fnv)} \dots\dots\dots\dots (6)$$

$$Recall = \frac{tnv}{(tnv + fpv)} \dots\dots\dots\dots (7)$$

Table 7 shows the values of precision and recall calculated for each folds. The PR curves have been cited in acquiring a clear informative picture about the performance of binary decision classifiers [31]. For a classifier that performs well, the curve is in the upper-right portion of the PR space. When the curve is plotted away from the upper-right portion, which means the values are far from 1, this indicates a poor performance. By plotting the PR curves, Figure 6 shows the performance of IHASS over the datasets.

5 DISCUSSION

Tables 4 and 5 show the RMSE and NDEI values for the training and testing sets. These two measurements indicate that the IHASS performs at good level of quality. Another indication can be revealed from the tables, the slight difference in IHASS training and testing errors over the three datasets. This fact is noticeably shown by the mean of RAMSE and NDEI values over the datasets.

IHASS system uses one tuning parameter, k which holds the number of nearest neighbors. It is very likely that a fixed value of k will cause a bias on different datasets. Therefore, we used different values of k for the datasets, rather than a fixed value across all datasets. Regardless of the online classification, in most cases including our case, the cross-validation is the best way to optimize the value of k. Table 6 shows the performance of IHASS in terms of accuracy using 4 values of k 5, 7, 9, and 11 across all datasets. We dedicated the first four experiments (folds) as trails for optimizing the value of k. In the fifth experiment, we used the values of k=7, k=7 and k=5 for the CSDC2010, ling, and Enron datasets respectively, as these values achieved the highest accuracy. However, on average, the accuracy of IHASS with the used value of k is slightly higher than the accuracies of IHASS with other values of k. Table 8 shows the differences of IHASS accuracies using k values for all datasets.

From this table, we conclude that IHASS has a low sensitivity to the number of nearest neighbors parameter, which a merit of IHASS as it confirms an easy way of tuning the system.

From Table 7, the averages of precision and recall are equal for both CSDC2010 and Enron datasets, and their values are 0.75 and 0.80 respectively. While the averages of precision and recall for the Ling dataset is 0.85 and 0.86 respectively. Intuitively, the total performance of IHASS could be considered low because the system covers only 75%, 86%, and 80% of the spam emails in the CSDC2010, Ling, and Enron datasets. As the main goal of IHASS is to classify the spam and ham emails, this implies a care of both precision

and recall measurements. F1-measure, which is defined as a harmonic mean of the precision and recall, is calculated to convey the balance between the precision and recall. The average values of F1-measure for the CSDC2010, Ling, and Enron datasets are 75%, 86%, and 80%.

Table 4: RMSE and NDEI statistical measurements for the training sets

	RMSE			NDEI		
	CSDC2010	**Ling**	**Enron**	**CSDC2010**	**Ling**	**Enron**
Mean	0.173	0.161	0.175	0.336	0.340	0.359
Min	0.103	0.115	0.119	0.302	0.301	0.322
Max	0.249	0.200	0.240	0.368	0.388	0.395
STDEV	0.055	0.025	0.042	0.022	0.029	0.026

Table: 5 RMSE and NDEI statistical measurements for the testing sets

	RMSE			NDEI		
	CSDC2010	Ling	Enron	CSDC2010	Ling	Enron
Mean	0.128	0.120	0.120	0.316	0.298	0.291
Min	0.108	0.100	0.100	0.269	0.257	0.254
Max	0.150	0.147	0.148	0.348	0.350	0.331
STDEV	0.013	0.016	0.019	0.026	0.033	0.029

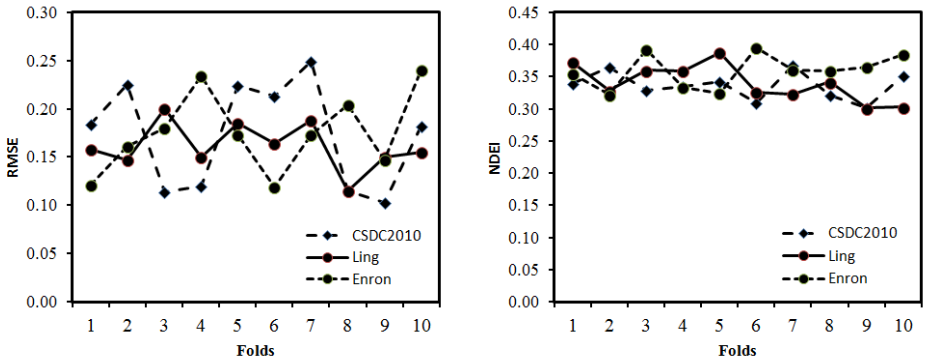

Fig. 4. RMSE & NDEI measurements for IHASS training sets

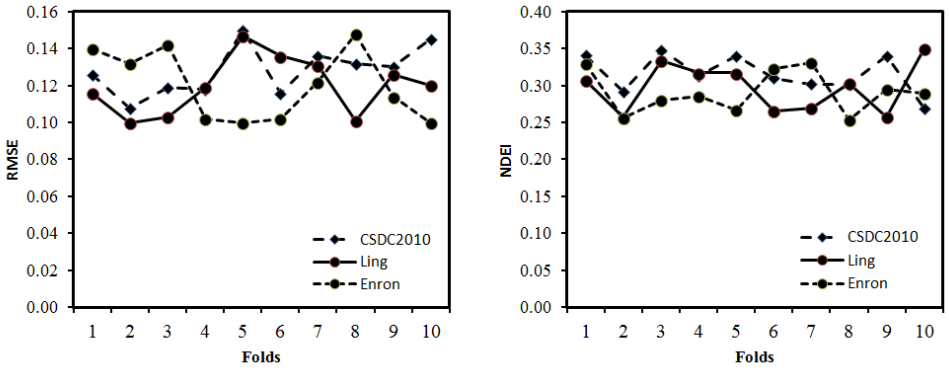

Fig. 5. NDEI measurements for IHASS testing sets

Table 6: IHASS testing results

	CSDC2010						Ling						Enron					
fold	k	tpv	fnv	tnv	fpv	accu	k	tpv	fnv	tnv	fpv	accu	k	tpv	fnv	tnv	fpv	accu
1	5	120	60	122	58	0.67	5	133	29	133	29	0.82	5	101	25	103	23	0.81
2	7	138	42	140	40	0.77	7	143	19	142	20	0.88	7	100	26	98	28	0.79
3	9	127	53	119	61	0.69	9	131	31	132	30	0.81	9	100	26	98	28	0.79
4	11	123	57	126	54	0.68	11	128	34	126	36	0.78	11	99	27	96	30	0.77
5	7	140	40	139	41	0.78	7	140	22	144	18	0.88	5	101	25	101	25	0.80
6	7	141	39	139	41	0.78	7	141	21	143	19	0.88	5	100	26	101	25	0.80
7	7	140	40	140	40	0.78	7	140	22	142	20	0.87	5	100	26	101	25	0.80
8	7	143	37	140	40	0.79	7	140	22	144	18	0.88	5	100	26	103	23	0.81
9	7	141	39	138	42	0.78	7	141	21	143	19	0.88	5	101	25	103	23	0.81
10	7	142	38	139	41	0.78	7	140	22	143	19	0.87	5	101	25	102	24	0.81

Table 7: Precision and recall

	CSDC2010		Ling		Enron	
fold	Precision	Recall	Precision	Recall	Precision	Recall
1	0.67	0.68	0.82	0.82	0.80	0.82
2	0.77	0.78	0.88	0.88	0.79	0.78
3	0.69	0.66	0.81	0.81	0.79	0.78
4	0.69	0.70	0.79	0.78	0.78	0.76
5	0.78	0.77	0.87	0.89	0.80	0.80
6	0.78	0.77	0.87	0.88	0.80	0.80
7	0.78	0.78	0.87	0.88	0.80	0.80
8	0.79	0.78	0.87	0.89	0.80	0.82
9	0.78	0.77	0.87	0.88	0.80	0.82
10	0.79	0.77	0.87	0.88	0.80	0.81

Table 8: Differences of IHASS accuracy in terms of k values

	CSDC2010		Ling		Enron	
Accuracy of	k = 7	k = 9	k = 7	k = 5	k = 5	k = 7
	0.77	0.69	0.88	0.82	0.81	0.79
Differences	8%		6%		2%	
Accuracy of	k = 7	k = 11	k = 7	k = 9	k = 5	k = 9
	0.77	0.68	0.88	0.81	0.81	0.79
Differences	8%		7%		2%	
Accuracy of	k = 7	k = 5	k = 7	k = 11	k = 5	k = 11
	0.77	0.67	0.88	0.78	0.81	0.77
Differences	10%		10%		4%	

6 CONCLUSION

Spam email is a serious problem nowadays and the solution to this problem is a challenge. This paper presents a novel Intelligent Hybrid Anti-Spam System (IHASS) for spam email detection, to overcome the drawbacks of the CBF techniques for spam email detection. The main characteristic of IHASS is the neuro-fuzzy inference engine, which has been applied successfully in a wide range of research. It uses same neuro-fuzzy inference engines for spam email detection. The neuro-fuzzy inference engine of IHASS is well trained to improve the detection accuracy of spam email. A new algorithm called Most Frequent Maximum Phrase (MFMP) is used to extract the most frequent phrases from the spam emails in the dataset. MFMP aims at maximizing the knowledgebase of neuro-fuzzy inference engine by increasing the number of phrases that the classifier will depend on in taking a classification decision. The experimental results show that the proposed IHASS is stable and detects the spam email with high accuracy level.

7 REFERENCES

[1] R. Islam and J. Abawajy, "A multi-tier phishing detection and filtering approach," Journal of Network and Computer Applications, vol. 36, pp. 324-335, 2013.

[2] Y. Zhang, S. Wang, P. Phillips, and G. Ji, "Binary PSO with mutation operator for feature selection using decision tree applied to spam detection," Knowledge-Based Systems, vol. 64, pp. 22-31, 2014.

[3] MAAWG, "Email metrics program: The network operators' perspective. Report #10 – Third and fourth quarter 2008," S. Francisco CA, USA.2009.

[4] T. K. Lab. (2015, October 28). Kaspersky Lab Spam and Phishing in Q2 2015 Report: .

[5] O. R. W. Paper, "Best Practices for Dealing with Phishing and Next-Generation Malware," Washington - USA April 2015.

[6] G. V. Cormack, M. D. Smucker, and C. L. Clarke, "Efficient and effective spam filtering and re-ranking for large web datasets," Information retrieval, vol. 14, pp. 441-465, 2011.

[7] J. Carpinter and R. Hunt, "Tightening the net: A review of current and next generation spam filtering tools," Computers & security, vol. 25, pp. 566-578, 2006.

[8] S. Garriss, M. Kaminsky, M. J. Freedman, B. Karp, D. Mazières, and H. Yu, "RE: Reliable Email," in Proceedings of the 3rd conference on Networked Systems Design and Implementation (NSDI), San Jose, CA, USA: , 2006, pp. 22-22.

[9] T. S. Guzella and W. M. Caminhas, "A review of machine learning approaches to spam filtering," Expert Systems with Applications, vol. 36, pp. 10206-10222, 2009.

[10] M. Chawla and S. S. Chouhan, "A Survey of Phishing Attack Techniques," International Journal of Computer Applications, vol. 93, 2014.

[11] J. R. Méndez, I. Cid, D. Glez-Peña, M. Rocha, and F. Fdez-Riverola, "A comparative impact study of attribute selection techniques on naive Bayes spam filters," in Advances in Data Mining. Medical Applications, E-Commerce, Marketing, and Theoretical Aspects, ed: Springer, 2008, pp. 213-227.

[12] F. Fdez-Riverola, E. L. Iglesias, F. Díaz, J. R. Méndez, and J. M. Corchado, "Applying lazy learning algorithms to tackle concept drift in spam filtering," Expert Systems with Applications, vol. 33, pp. 36-48, 2007.

[13] R. B. Basnet and A. H. Sung, "Classifying phishing emails using confidence-weighted linear classifiers," in International Conference on Information Security and Artificial Intelligence (ISAI), 2010, pp. 108-112.

[14] M.-w. Chang, W.-t. Yih, and C. Meek, "Partitioned logistic regression for spam filtering," in Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining, 2008, pp. 97-105.

[15] H. Drucker, D. Wu, and V. N. Vapnik, "Support vector machines for spam categorization," Neural Networks, IEEE Transactions on, vol. 10, pp. 1048-1054, 1999.

[16] J. Moon, T. Shon, J. Seo, J. Kim, and J. Seo, "An approach for spam e-mail detection with support vector machine and n-gram indexing," in Computer and Information Sciences-ISCIS 2004, ed: Springer, 2004, pp. 351-362.

[17] I. Androutsopoulos, J. Koutsias, K. V. Chandrinos, and C. D. Spyropoulos, "An experimental comparison of naive Bayesian and keyword-based anti-spam filtering with personal e-mail messages," in Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval, 2000, pp. 160-167.

[18] N. Pérez-Díaz, D. Ruano-Ordás, F. Fdez-Riverola, and J. R. Méndez, "SDAI: An integral evaluation methodology for content-based spam filtering models," Expert Systems with Applications, vol. 39, pp. 12487-12500, 2012.

[19] C.-C. Lai, "An empirical study of three machine learning methods for spam filtering," Knowledge-Based Systems, vol. 20, pp. 249-254, 2007.

[20] J. Levine, "RFC 5782: DNS Blacklists and Whitelists," Technical report, The Internet Engineering Task Force2010.

[21] D. De Capitani, E. Damiani, S. De Vimercati, P. Capitani, and P. Samarati, "An open digest-based technique for spam detection," in Proceedings of International Workshop on Security in Parallel and Distributed Systems, 2004.

[22] F. Zhou, L. Zhuang, B. Y. Zhao, L. Huang, A. D. Joseph, and J. Kubiatowicz, "Approximate object location and spam filtering on peer-to-peer systems," in Middleware 2003, 2003, pp. 1-20.

[23] A. Shubair, S. Ramadassb, and A. A. Altyebb, "kENFIS: kNN-based evolving neuro-fuzzy inference system for computer worms detection," Journal of Intelligent & Fuzzy Systems: Applications in Engineering and Technology, vol. 26, pp. 1893-1908, 2014.

[24] V. Metsis, I. Androutsopoulos, and G. Paliouras, "Spam filtering with naive bayes-

which naive bayes?," in CEAS, 2006, pp. 27-28.

[25] I. Androutsopoulos, J. Koutsias, K. V. Chandrinos, G. Paliouras, and C. D. Spyropoulos, "An evaluation of naive bayesian anti-spam filtering," arXiv preprint cs/0006013, 2000.

[26] J. Yang, Y. Liu, Z. Liu, X. Zhu, and X. Zhang, "A new feature selection algorithm based on binomial hypothesis testing for spam filtering," Knowledge-Based Systems, vol. 24, pp. 904-914, 2011.

[27] E. Blanzieri and A. Bryl, "A survey of learning-based techniques of email spam filtering," Artificial Intelligence Review, vol. 29, pp. 63-92, 2008.

[28] L. Yujian and L. Bo, "A normalized Levenshtein distance metric," Pattern Analysis and Machine Intelligence, IEEE Transactions on, vol. 29, pp. 1091-1095, 2007.

[29] P. Domingos, "A few useful things to know about machine learning," Communications of the ACM, vol. 55, pp. 78-87, 2012.

[30] R. Bunescu, R. Ge, R. J. Kate, E. M. Marcotte, R. J. Mooney, A. K. Ramani, et al., "Comparative experiments on learning information extractors for proteins and their interactions," Artificial intelligence in medicine, vol. 33, pp. 139-155, 2005.

[31] J. Davis and M. Goadrich, "The relationship between Precision-Recall and ROC curves," in Proceedings of the 23rd international conference on Machine learning, 2006, pp. 233-240.