I. J. Computer Network and Information Security, **2018, 12, 1-10**
Published Online December 2018 in MECS (http://www.mecs-press.org/)
DOI: 10.5815/ijcnis.2018.12.01

An Email Modelling Approach for Neural Network Spam Filtering to Improve Score-based Anti-spam Systems

Yahya Alamlahi
IT Operations and Systems Manager at CACBank®, Open University Malaysia (OUM) – UST Centre, Sana'a, Yemen
E-mail: yahya.alamlahi@hotmail.com

Abdulrahman Muthana
Thamar University, Yemen
E-mail: ab.muthana@gmail.com

Received: 26 September 2018; Accepted: 18 October 2018; Published: 08 December 2018

Abstract—This research proposes a model for presenting email to Artificial Neural Network (ANN) to classify spam and legitimate emails. The proposed model based on selecting wise 13 fixed features relevant to spam emails combined with text features.

The experiment tests many scenarios to find out the best-suited combination of features representation. These scenarios show the effect of using term frequency (tf), term frequency-inverse document frequency (tf*idf), Level two (L2) normalization, and principal component analysis (PCA) for dimension reduction. Text features vectors are represented in the principal component space as a reduced form of the original features vectors. PCA reduction effect on ANN performance is also studied.

Among these tests, best-suited model that improves ANN classification and speeds up training is concluded and suggested. An idea of integrating ANN anti-spam filter into score-based anti-spam systems is also explained in this paper. XEAMS email gateway, the commercial anti-spam, already uses Naïve Bayes (NB) filter as one of its many techniques to identify spam email. The proposed approach influences filtering results by 7.5% closer to XEAMS anti-spam system results than NB filter does on real-life emails of Arabic and English messages.

Index Terms—Artificial Neural Networks, E-mail classification, Spam filtering, Machine learning, principal component analysis.

I. Introduction

Many techniques introduced to fight spam using machine-learning applications. Email messaging systems is being the most important means of communication in any organization and integrated into its business processes. This valuable service can be misused intentionally for sending emails in bulk, for advertising and marketing products, or even sending malware to others.

Statistical and weight-based anti-spam uses a scoring system for many features of email text to classify it. For example, it gives specific scores for Sender Policy Framework (SPF) record, Real time blacklist, sender features, content-based features, bag of word regression expression, Naïve Bayes filter, etc. then it evaluate total of scores by using specific weighting in order to be compared to threshold value by which it classify each email to legitimate, spam, possible spam. XEAMS (eXtended Email and Messaging System) gateway is an example that uses these technique of spam fighting. Xeams anti-spam system calculates summation (negative and positive values) of ranks (scores) resulting from around 24 different filters then compare it with predefined and configurable threshold to decide whether the email is spam, possible spam, or legitimate.

This paper focuses on implementing ANN email classification filter by selecting the best email model that suits ANN. Then the ANN spam filter performance is compared to the performance of Naïve Bayes filter in XEAMS gateway to justify using it as a one dependable filter that improve such type of these anti-spam systems.

Below definitions, illustrate any abbreviation used in the research:

Filter Accuracy (A): it is the percentage of all emails that are correctly categorized.

Spam Precision (P): it is the proportion of emails classified as spam that are truly spam.

Spam Recall (R): it is the proportion of spam e-mails that are classified as spam, i.e. the spam e-mail that the filter manages to block.

False Positive Spam: equivalent to false alarm. It is an email classified incorrectly as spam.

Contingency Table 1 easily explains the relation between them and give the mathematical definition.

Table 1. Contingency Table of Spam Classification

	Actual Spam messages	Actual legitimate messages
Classified as spam	True positive (*tp*)	False positive (*fp*)
Classified as legitimate	False negative (*fn*)	True negative (*tn*)

Above terms can be calculated by:

Filter **A**ccuracy: $A = \frac{tp+tn}{tp+fp+fn+tn}$, Spam **P**recession: $P = \frac{tp}{tp+fp}$, Spam **R**ecall: $R = \frac{tp}{tp+fn}$, and $F1$ is called *F measure* at $\beta = 1$ given by $F1 = \frac{2PR}{P+R}$.

Accuracy alone is not an appropriate measure for information retrieval problems. Same thing for P or R alone so that F measure introduced based on both R and P values. It is a single measure that trades off precision P versus recall R. it is the weighted harmonic mean of precision and recall (Manning, Raghavan, & Schütze, 2009[1]). The research uses F1 as F measure.

II. Literature Review

ANN filters like any algorithms in machine learning field begins with a vector representation of individual email message. Researchers studied many different representations of emails. Most of them uses "bag of words" representation in which each email represented by a vector of distinct terms. Length of the vector is the number of the distinct words in all the emails in the training data. Each element in the vector is given a different meaning from a researcher to another. Some implementation uses term frequency (tf), tf-idf, or a normalized form of these statistical representations.

Size of the vector may become huge based on the diversity of the training set. Various techniques introduced to reduce its size. One of them is to remove stop words (words like if, of, and, etc.). Second technique is to reduce words to their root form by a process called stemming (so, for example, "went" and "gone" reduced to "go") (Madigan, 2005[2]). In Arabic language, different forms of letter can be reduced to one form (for example, hamza "ء", alef with upper hamza "أ", and alef with lower hamza "إ" all are reduced to alef without hamza "ا" so that words appears in one form) (Goweder, Rashed, Elbekaie, & Alhammi, 2008[3]). Furthermore, diacritics, connectors, suffixes and prefixes can be removed in a manner that does not distort the word especially in Arabic and dialects (Alamlahi & Ahmed, 2007[4]). Blanzieri & Bryl, 2008 [5] survey gives an overview of machine learning applications for spam filtering, and the ways of evaluation and comparison of different filtering methods. It states that the Naıve Bayes classifier occupies a special place. (Creech & Jiang, 2012[6]) uses a combined semantic and statistical approach to feature extraction. Nine statistical and two semantic features are extracted from the various emails. They are total word count, ratio of blank lines to total lines, number of hyperlinks, word count to punctuation ratio, number of compound words, ratio of capitals to lower case, number of words using numbers and punctuation marks to obfuscate spelling, number of repeated words, number of repeated punctuation events. The semantic features are semantic analysis against spam word use, and semantic analysis against bona fide word use. The first nine features, derived statistically, relating to the fundamental differences between spam emails and legitimate emails. It is similar to weight measure used in literature. In (Bansod, Mangrulkar , & Bhujade, 2015[7]), they assign weight to the spam words. Their system is based on spam document contains weights for each spam words. The weights are assigned in between 0 to1 for the spam words and 0 to -1 for the non-spam words.

Spam feature weighting, along with professional pre-processing such as regex other than simple stemming, and assigning weights to many algorithms decisions working concurrently at the same time are proven technique in commercial anti-spam like XEAMS mail gateway that eliminates up to 99% of junk email right out of the box (Synametrics Technologies, n.d. [8]).

(Clark, Koprinska, & Poon, 2003[9]) shows that stemmer and stop words improve the performance with only 0.7% and worsen it with 1% on encrypted corpus, therefore this approach uses stemmer and stop words to reduce data size and not to increase performance of the filter. The approach uses neural network in both training and classification processes whereas Goweder, Rashed, Elbekaie, & Alhammi [3] implement an anti-spam system that uses a Multi-Layer Perceptron (MLP) as a classifier and a Genetic Algorithm (GA) as a training algorithm. Their implementation has achieved accuracy of 94% to detect spam emails and 89% to detect legitimate emails.

In (Cui, Mondal, Shen, Cong, & Tan, 2005[10]), they propose a method based on extracting predefined semantics (from, to, CC, etc.) as well as variable length free-text fields (Subject and body). They also propose using Principal Component Analysis (PCA) as a pre-processor of ANN to reduce the data in terms of both size and dimensionality so that input data become more classifiable and faster convergence of training process used by ANN. Their extraction method is based on seven fixed features and text features. The seven fixed features are Attachment, Content type, Sender domain, FW, RE, To-Group, and CC. Text features for subject and body extracted using tf*idf method. Overall, they found that PCA improves classification from 85% to 93%.

Hence, the research proposes a model using wise selection of fixed features by make use of both Cui, Mondal, Shen, Cong, & Tan and Creech & Jiang approaches as well as new specific additional fixed features that make sense for differentiate spam and spammer behaviour. Spammers use misspelled words to fool antispam system so that exact matching of spam word does not help in identifying spam emails. Body and subject are free text fields in which the approach tests both simple tf and tf*idf representations. This representation produces large vector of features then reduced as possible by PCA to guarantee best results.

III. Methodology & Implementation

A. Data Collection

The research is based on two large corpora. Both are in eml file format. They are plain text without encryption. One is ready for use and available online for data mining and anti-spam researches. It is CSDMC2010 SPAM corpus (Unitec, 2010[11]). The second is a private corpus collected from CACBANK®. All rights are reserved for the organization and it is not permitted to be published. It gives a real-life sample of dataset that contains Arabic and English messages. The paper refers to it as CAC2016©.

CSDMC2010 SPAM corpus contains two parts: TRAINING, 4327 messages out of which there are 2949 non-spam messages (HAM) and 1378 spam messages (SPAM), all received from non-spam-trap sources. They are labelled with 1 stands for a HAM and 0 stands for a SPAM. TESTING, 4292 messages without known class labels. The experiment uses only 1700 emails of them.

CAC2016 corpus is collected within February 2016. It is for consecutive 29 days. Every email is already statistical ranked with Synametrics XEAMS Anti-spam gateway as well as assigned Naïve Bayes value and categorized in three classes; spam, good and possible spam. Good message means legitimate email (HAM), possible spam means the email is tend to be spam. In other words, possible spam means XEAMS cannot decide whether the email is spam or not. Corpus contains 51,820 emails in eml file format classified in three classes: 16,638 legitimate emails (HAM), 32,612 SPAM emails, and 2,570 possible spam messages (POSSIBLE SPAM). The experiment uses only 1576 consecutive emails from this corpus.

B. Spam Labels (Target Array for ANN)

Instead of representing spam labels by 1 and 0, labels are represented by pair of bits; [0 1] for spam email, and [1 0] for legitimate email. ANN for pattern recognition and classification are best working with such form of targets other than using one bit to represent two classes. The researcher tests both ideas to prove this. Hence, the proposed ANN has two neurons in output layer instead of one.

C. The Statistical Representation for Emails

In order to present email to ANN, each email is represented by a vector of values. The challenge is to choose the best representation that suit ANN and improve classification process.

Two vectors represent each email. One for fixed features and the other for text features. The proposed 13 fixed features (denoted SF – structured features) extracted wisely from the email message fields. Unstructured features (free text fields which are subject and body fields of email message) -as popular- is represented by term frequency (TF) and term frequency-inverse document frequency (TF-IDF). Level 2 (L2) normalization of the vectors eliminates variance in value in each vector.

Then, horizontal joining fixed features (structured features) vector with either TF vector or TFIDF vector give the statistical representation for the email message. Possible combinations of vectors to represent each email message are denoted as below:

1. eml_sf-tf[]: structured features (SF) with term frequency representation (TF) of text.
2. eml_sf-tfl2[]: structured features (SF) with level 2 normalization of TF.
3. eml_sfl2-tfl2[]: level 2 normalization of SF with level 2 normalization of TF.
4. eml_sf-tfidf[]: horizontally joining SF vector with TFIDF vector.
5. eml_sf-tfidfl2[]: SF with L2 of TFIDF.
6. eml_sfl2-tfidfl2[]: L2 of SF vector with L2 of TFIDF.

Target vector is the actual classification of the sample in form of [0 1] for spam message and [1 0] for legitimate message. Hence, spam label vector is the target vector consisting of rows equal to number of samples and two columns represents 0 1 or 1 0.

Collection of samples are represented as two-dimension vector in which each row represents one sample and columns represent its features. Unique vocabs extracted from 1576 samples of CAC2016 corpus are 13095 terms, whereas 1700 samples of CSDMC2010 corpus produce 24955 terms. For example, the email *i* vector is represented as eml_sfl2-tfidfl2[doc_i][0..total_terms], and the corresponding target output is represented as spamLabel[doc_i][0..1]. Variable *i* is an integer between 0 to samples_count-1.

By evaluating which of these representations influence ANN performance, the research concludes the suggested best approach for email modelling. The candidate email representation is used to test PCA efficiency in reducing features space while keeping ANN performs well in email classification. The approach for ANN email classification is identified by selecting the best configuration leads to best results.

D. The 13 Fixed Features Representation

These features are extracted wisely from the email message fields by choosing features that make sense to email classes. Their numerical value is selected wisely too. As 0 denotes for SPAM and 1 for HAM, the lower numbers among evaluation set of values are assigned to the features that tend to be spam-like. For example, true value of empty TO feature is assigned 0 whereas country TLD feature existence is assigned 1 because its existence means it is less probable spam email. Table 2 lists the 13 fixed features (structured features) and their evaluation.

Table 2. The Structured Features and Evaluation

#	Description	Value	Evaluation
1	Empty TO	Boolean	true=0, false=1
2	Base64	Boolean	Content_Transfer_Encoding=base64? true=0, false=1
3	Multipart	Integer	Parts count
4	Attachment indicator	Integer	Email size in KB divided by 512, rounded.
5	country TLD	Boolean	In sender address. True=1, false=0
6	RE:/FW:	Boolean	Subject starts with RE or FW, true=1, false=0
7	img tag count	Integer	<img src tag count in html body
8	<a hRef tag count	Integer	<a hRef count in html body.
9	http count	Integer	http frequency in html body
10	@ count	Integer	Character '@' count in subject and html body
11	* Count	Integer	Character '*' Count in subject and html body
12	Language	Integer	Natural language detected 0,1,2
13	Spam words	Integer	Minimum Levenshtein distance between subject/body words and bag of spam words. Hinted from the fact that spammers use obfuscate spelling.

E. Experiment Steps

The experiment is performed in the following sequenced steps using both corpora:

Step 1. Building Emails model:

1. Read emails' files (eml format) using MimeKit (Stedfast, 2015[12]). It is an Open Source library for creating and parsing MIME, S/MIME and PGP messages on desktop and mobile platforms. It also supports parsing of Unix mbox files.
2. Build SF[ ] array for each email file (document) based on the evaluation ideas in table 2.
3. Remove stop words from subject and body of each email message. Then apply suitable stemmer based on language of the message. You can also change some variables data inside text such as telephone numbers, addresses, cities names, counties names, … etc. by a single meaningful word for each category to improve tf/tf*idf representation. For example, text like "Hi Sam, I am in Sana'a. I called you on 012323, but no response. Can you travel with me from Sana'a to Egypt? Please call me on my cell 7712345" should be converted to "Hi person, I am in CityName. I called you on PhoneNumber, but no response. Can you travel with me from CityName to CountryName? Please call me on my cell PhoneNumber". This experiment considers both Arabic and English stop words.
4. Apply language stemmer for each term. Suitable stemmer can be applied based on language identified for each email by NTextCat categorizer. As most of spams are in English, the researcher uses Centivus English Stemmer - the porter-stemmer algorithm (Porter, 2006[13]) to obtain the collection of base distinct words.
5. Build TF[ ] and TFIDF[ ] vectors from subject and body of all messages. The implementation is based on Kory Becker TF*IDF implementation (Becker, 2013[14]) with minor changes to handle language detection and fixing some runtime exceptions raised due to the large corpus being processed.
6. Normalize all vectors in L2-norm.
7. Writing sf, tf, sfl2, tfl2, tfidf, tfidfl2 vectors to csv files. All of them -except sf, sfl2 – will be used later to apply PCA on them using MATLAB then test dimensional reduction effect on ANN classification.
8. Writing emails models in csv files by making use of the combination of those vectors.
9. As those emails are already classified in spam/ham, build vector of spam labels to be target vector for training set of ANN. Spam labels vector also is written to csv file.

Step 2. Presenting Models to ANN in MATLAB:

1. Start neural network tool in MATLAB using nnstart command.
2. Choose classification and pattern recognition type of ANN.
3. Use 70% of samples as training set, and 15% for validation set, and remaining 15% of samples for test set.
4. Train a NN with various neurons configuration using models eml_sf, eml_sfl2, eml_sf-tf, eml_sf-tfl2, eml_sfl2-tfl2, eml_sf-tfidf, eml_sf-tfidfl2, eml_sfl2-tfidfl2.
5. Write down performance result of ANN for each models.
6. Based on result, suggest best model suited for ANN that influence classification accuracy. That is, discussing the following:

   i. L2 normalization effect.
   ii. Using the structured features SF[] alone as an input model for ANN. How ANN performs in this case. It shows SF selection efficiency.
   iii. TF or TF*IDF do best in ANN spam classification.

7. Choose the best model that based on TF or TF*IDF from the previous step and reduce data size by using principal component analysis PCA function on it. Apply reduced data as input for ANN and compare results. Conclude how much PCA reduces text features vector in percentage to original data and affects ANN classification.

Step 3. As the experiment tests various models and configuration, conclude the best approach for modeling message for ANN to perform email spam filtering in acceptable accuracy.

Step 4. Compare survive approach concluded from step 3 with XEAMS scores and naïve Bayes (NB) classification. Conclude how ANN with this modelling influences any anti-spam gateway filter:

1. Collect suitable test sample from “possible spam” emails of CAC2016 corpus. These emails have boundary scores of XEAMS so they represent good test sample. The experiment uses 1285 samples of them.
2. Extract all NB scores and XEAMS scores for the test sample.
3. Read each email from the test sample and extract its features according to the model used in the trained ANN and its vector of features. That means vector of vocabulary should be maintained from the training phase and evaluate them for each email of the test sample.
4. Present each email as vector of values to the trained ANN and record probability of being SPAM in order to be compared to the values of NB scores.
5. Calculate deviation of NB and ANN outputs from XEAMS scores. The closer one to XEAMS scores is the better filter.

F. *ANN Implementation*

A two-layer feed-forward network, with sigmoid hidden and softmax output neurons (feedforward networks that can be trained to classify inputs according to target classes), can classify vectors arbitrarily well, given enough neurons in its hidden layer. The network is trained with scaled conjugate gradient backpropagation (a network training function that updates weight and bias values according to the scaled conjugate gradient method).

Spam labels are represented by a pair of bits so two neurons are required in the output layer. The number of neurons in input layer equals to the total extracted features from the corpus under study. Many researches have been made in evaluating the number of neurons in the hidden layer but still none of them was successful in finding the accurate result. This research is one of them too. For each test scenario, different number of neurons in hidden layer is tested to check best-suited hidden layer size that gives the best result. The proposed ANN is shown in Fig. 1. Number of neurons in hidden layers and input size vary for each scenario from the shown on the figure. The study uses MATLAB R2015a with the following ANN algorithms configuration: Random Data Division, Scaled Conjugate Gradient Training, Cross-Entropy Performance Calculation Method, MEX Calculations, double data-type for vectors’ elements.

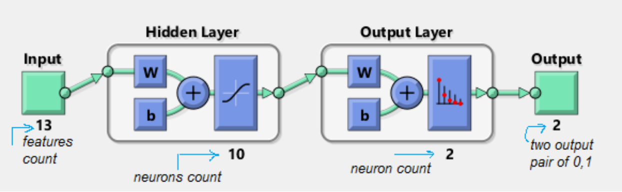


Fig.1. ANN Diagram, Numbers Differ for each test Scenario, MATLAB Depiction

IV. Results

A. *Normalization Effect*

By using fixed number of neurons (let it is 20), any one corpus (Let it CSDM2010), and one text features modeling (Let it is term frequency), test various models with and without L2. Comparing ANN performance for models like eml_SF-TF, eml_SF-TFL2, eml_SFL2-TFL2 gives an indicator of L2 efficiency and usage. Table 3 shows the results.

Table 3. L2-norm Effect Comparison with TF

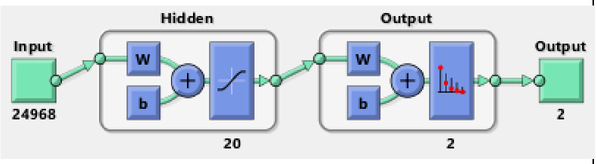
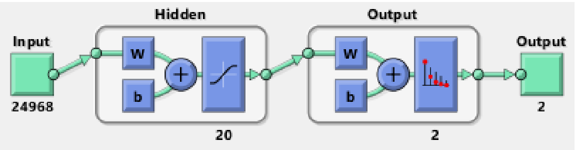
L2-norm effect comparison with TF			
 Input Hidden Output Output W b W b 24968 20 2 2			
Hidden Layer Neurons: ***20***			
Corpus: ***CSDM2010***			
Text Features Model: *Term Frequency (TF)*			
Samples number: ***1700***, training 70%, Validation 15%, Test 15%			
Features count: ***24968***			
%E: misclassification percentage			
Email model	eml_SF-TF	eml_SF-TFL2	eml_SFL2-TFL2
Epoch#	238	49, 20, 205	**37**, 89, 74
Time MM:SS	05:37	01:10, -, 04:52	00:49, 02:06, 01:44
Training %E	**2.69%**	6.97%, -, 8.40%	31.68%, 14.37%, 12.44%
Validation %E	7.06%	11.76%, -, **4.71%**	28.24%, 17.65%, 17.25%
Test %E	**2.35%**	11.37%, -, 4.71%	31.76%, 12.16%, 14,12%
Overall Accuracy	96.7%	91.6%, -, **98.5%**	68.8%, 85.5%, 86.6%
Overall Precession (P)	92.3%	80.5%, -, **96.8%**	-, -, 88.5%
Overall Recall (R)	97.5%	96.6%, -, **98.5%**	-, -, 65.5%
Overall F1	94.83%	87.82%, -, **97.64%**	-, -, 75.28%
Results: eml_SF-TFL2 gives best result, so L2 is good to use with Text Features only. Multivalued field above is for repeated test.			

Table 4. L2-norm Effect Comparison with TF*IDF

L2-norm effect comparison with TF*IDF			
 Hidden Output Input W b W b Output 24968 20 2 2			
Hidden Layer Neurons: ***20***			
Corpus: ***CSDM2010***			
Text Features Model: *Term Freq.* inverse Document Freq. (TF*IDF)*			
Samples number: ***1700***, training 70%, Validation 15%, Test 15%			
Features count: ***24968***			
%E: misclassification percentage			
Email model	eml_SF-TFIDF	eml_SF-TFIDFL2	eml_SFL2-TFIDFL2
Epoch#	59, 256	75	72
Time MM:SS	01:27, 06:02	01:46	01:40
Training %E	3.87%, **2.02%**	5.88%	5.13%
Validation %E	7.84%, 6.67%	**2.75%**	8.24%
Test %E	8.63%, 6.67%	**5.49%**	9.02%
Overall A	94.8%, 96.6%	**98.4%**	93.8%
Overall P	88.4%, 92.6%	**97%**	88.3%
Overall R	96%, 96.8%	**97.7%**	92.5%
Overall F1	92.04%, 94.65%	**97.35%**	90.35%

B. TF or TF-IDF?

To compare tf*idf model efficiency against tf model, also use the same configuration and test the three models of tf*idf as shown in table 4.

By comparing results on table 3 and table 4, obviously tf*idf model is better than TF in increasing ANN accuracy and lowering epochs. L2 normalization also improves tf*idf model. Candidate email modeling is eml_SF-tfidfL2 for any further test scenarios. Second candidate can be eml_SF-TFL2.

C. Fixed Features Efficiency

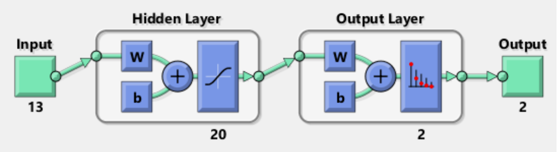
Comparing the inclusion of fixed features (eml_SF-TFIDFL2) to the use of text features only (eml_TFIDFL2) on table 5 shows that the 13 fixed features improves performance in terms of precession (P), accuracy (A) and F1 measure by 7.48% to 12.3% at the clearest difference (at maximum). Recall results is better without using the fixed features by 3.9% at maximum!

Using SF alone gives good precession results but bad accuracy, recall and F1 measure results. SF in L2 normalization improves ANN performance in all aspects: accuracy (A), precession (P), recall (R) and F1 measure as shown in table 6.

Table 5. ANN Performance With/Without Structured Features

	F1 measure		Recall		Precision		Accuracy		No. of Epochs	
neurons	*with SF*	*without SF*	*with SF*	*without SF*	*with SF*	*without SF*	*with SF*	*without SF*	*with SF*	*without SF*
10	94.30	94.84	94.3	98.2	94.3	91.7	92.5	93	63	74
20	92.24	93.03	98.4	98.3	86.8	88.3	89.1	90.3	170	264
30	90.90	88.29	96.9	99.1	85.6	79.6	87.3	82.8	169	47
40	93.42	92.17	97.7	98	89.5	87	90.9	89.1	181	187
50	87.00	89.12	97.4	98.3	78.6	81.5	80.9	84.3	80	98
60	94.14	94.24	98.2	98.9	90.4	90	91.9	92.1	230	341
80	93.87	86.39	97.5	96.5	90.5	78.2	91.6	80.1	168	70
100	94.53	93.88	97.1	97.3	92.1	90.7	92.6	91.7	225	113

Table 6. SF Model Alone Efficiency

Structured Features Alone efficiency		
 Hidden Layer Output Layer Input W b W b Output 13 20 2 2		
Hidden Layer Neurons: ***20***		
Corpus: ***CSDM2010***		
Features Model: *My Structured Features Alone*		
Samples number: ***1700***, training 70%, Validation 15%, Test 15%		
Features count: ***13***		
%E: misclassification percentage		
Email model	eml_SF	eml_SFL2
Epoch#	90	48
Time MM:SS	00:02	00:01
Training %E	16.39%	9.16%
Validation %E	18.04%	8.24%
Test %E	15.69%	10.20%
Overall A	83.5%	90.8%
Overall P	**93.7%**	**96.8%**
Overall R	50.4%	73%
Overall F1	65.54%	**83.23%**

At this stage of the experiment, two message models perform well in ANN classification. They are eml_SF-TFL2 and eml_SF-TFIDFL2. Due to high number of iterations required with the first model, the other one (eml_SF-TFIDFL2) is the best. All subsequent test cases are performed using the model eml_SF-TFIDFL2 of corpus.

D. Hidden Layer's Neurons Effect

For CSDMC2010 corpus (24968 input features), ANN with hidden layer of 10 to 50 neurons reaches the best classification accuracy. For Corpus CAC2016 (13108 input features), ANN with almost hidden layer of size less than 100 neurons reaches the best classification performance. The experiment shows that large number of neurons in hidden layer is not required to improve ANN. Hence, this research recommends finding out the suitable hidden layer size that improves overall results.

E. Principal Component Analysis (PCA) Effect

PCA reduces the features numbers while keeping the most important ones without worsen ANN classification. PCA is applied only on text features because they are huge according to corpus size. The experiment uses the representation of text features vectors in the principal component space as a reduced form of text features vectors. This test case shows how much PCA reduces features numbers and retests ANN with the new features space.

The experiment tests PCA effect using various ANN with different hidden layer size on both corpora. Table 7 presents results achieved at 20, 40 and 60 neurons in hidden layer. It is clear that ANN training is faster with PCA due to smaller input size and lower number of epochs. Intensive tests of PCA effect on CSDMC2010 corpus shows that using PCA does not always improve ANN classification. With PCA reduction, ANN works better on CAC2016 corpus in terms of all metrics.

Table 7. PCA Effect

Neurons	Corpus	Comp.	Epoch	A %	P %	R %	F1 %
60	CSDMC2010	*No PCA*	143	91.40	97.50	74.30	84.33
		With PCA	14	93.60	95.10	83.80	**89.09**
	CAC2016	*No PCA*	230	91.9	90.4	*98.2*	94.14
		With PCA	44	95.8	96.1	97.6	**96.84**
40	CSDMC2010	*No PCA*	332	94.00	98.60	81.90	89.48
		With PCA	24	93.90	95.50	84.50	**89.66**
	CAC2016	*No PCA*	181	90.90	89.50	97.70	93.42
		With PCA	111	94.30	94.00	97.50	**95.72**
20	CSDMC2010	*No PCA*	75	98.40	97.00	97.70	**97.35**
		With PCA	17	92.40	90.20	84.90	87.47
	CAC2016	*No PCA*	170	89.10	86.80	98.40	92.24
		With PCA	80	95.30	96.00	96.60	**96.45**

PCA reduces 13095 terms of CAC2016 corpus to 1575 terms. PCA also shows that the first 1061 features of reduced space of CAC2016 corpus explains 99.9999% of the original space. Testing ANN with this part of the reduced space gives how much ANN performance may be degraded with this part of space (reducing the reduced space by 33.19%). Table 8 compares ANN performance on the full space, reduced space, part of the reduced space.

Table 8. Reduce the Reduced Space of PCA Effect on ANN

Corpus CAC2016				
email model:	*SF and TFIDFL2 for Text features*			
99.99999% explained:	1061			
new reduction ratio:	91.90%			
reduced from PCA:	33.19%			
text features used	***All (13095)***	***1575***	***1061***	
hidden Layer Neurons	60			degradation ratio from 1575 to 1061 ↓
Epoch	230	44	36	
time MM:SS	6:31	00:07	00:06	
Training %E	6.79%	1.45%	1.45%	
Validation %E	11.02%	10.17%	11.02%	
Test %E	11.02%	11.02%	11.80%	
Accuracy (A) %	91.90	**95.80**	94.70	**1.15%**
Precession (P) %	90.40	**96.10**	95.20	**0.94%**
Recall (R) %	**98.20**	97.60	96.70	**0.92%**
F1 %	94.14	**96.84**	95.94	**0.93%**

Once PCA application improves ANN performance, using most explaining components of reduced features vector is also possible without noticeable degradation of ANN performance (around 1% only).

F. The Candidate ANN Approach

From the above experiment scenarios, suggested approach for ANN is to use:

1. Vector of the 13 fixed Features (SF vector).
2. Reduced text features gained by PCA on tf*idf in Level 2 norm representation of subject and body of emails.
3. Model the email message as eml_SF-PCA(TFIDFL2). It is a horizontal concatenation of SF vector with reduced vector given by PCA of TFIDFL2 vector or even by the most explaining components of PCA space.
4. Select best-suited number of neurons in hidden layer by examining various number of neurons effect. Usually, it is between 10 and 100 neurons. For CAC2016, 60 neurons is suggested.

G. Approach Efficiency in Score-based (weight-based) Anti-spam

The trained model is tested on an unclassified test sample. The test sample contains 1285 unclassified emails marked as possible spam by XEAMS. Both Naïve Bayes (NB) and NN classifications are compared using this sample. The ANN is trained until validation stops at 103 iteration. Accuracy measures of this ANN was P=92.4%, R=95.7% and F1=94.02%. The ANN outputs two values per email message. They are HAM and SPAM probability. If they are close to [1 0], the message is HAM, and if they are close to [0 1], the message is SPAM. By comparing these values with NB and XEAMS scores, ANN efficiency in score-based anti-spam is concluded. Table 9 illustrates the comparison.

Therefore, it is possible to compare NB result to NN result using XEAMS score as a reference, which is a sum of many spam criteria. Deviation of both NB and NN regarding XEAMS results calculated for a sample of 1285 emails as shown in table 9. It proves that this method of using ANN influences score-based anti-spam system by 7.5% better than Naïve Bayes although NB result is used as part of the referential values. It is promising result because the ANN used in this test has only 94.02 for F1 measure.

Table 9. ANN vs. NB on Unclassified Test Sample

#	Xeams Certainty %	NB Certainty SPAM %	NN Certainty SPAM %	(-XE+NB)^2	(-XE+NN)^2
1 2	. .	. .	. .	. .	. .
. .	. .	. .	. .	. .	. .
1283	100	98.57143	71.84855	2.040816327	792.5040928
1284	100	98.57143	65.29693	2.040816327	1204.302799
1285	100	98.57143	54.41904	2.040816327	2077.623824
Variance=				2249.960509	1925.229319
Population deviation=				47.43374863	43.8774352
NN lower deviation than NB by %=					7.497432803

V. Main Finding

- Selected fixed features improves ANN classification. The selected 13 fixed features is not enough alone for ANN to classify emails in acceptable results of F1 and recall, although it gives high classification precision. Using fixed features along with text features improves ANN classification in terms of accuracy, precision and F1 measure. Recall rate is better without using fixed features. The research recommends presenting the email to ANN in form of a vector of both fixed features and text features.
- L2-norm of vector of features in preprocessing data is important to improve ANN and speed up training.
- Either term frequency (tf) or term frequency-inverse document frequency (tf*idf) can be used to represent text features to be presented to ANN. tf*idf with L2-norm gives best result and speeds up ANN training more than tf. Drawback of tf*idf is that its calculations are more complex than tf calculations.
- Principal Component Analysis (PCA) can be used to reduce features space and speed up training phase of ANN while keeping excellent performance in most cases. Using PCA does not always improve ANN classification Once PCA application improves ANN performance, using most explaining components of reduced features vector is also possible without noticeable degradation of ANN performance.
- It is still difficult to specify the required number of neurons in hidden layer of ANN that can best perform email classification. The study recommends examining various hidden layer size until reach the accepted ANN results. The experiment shows that the number is between 10 to 100 neurons for the corpora under study.
- Most of score-based anti-spam systems use Naïve-Bayes score to improve their decision in classifying spam emails in addition to other criteria scores of spam features. Like Naïve-Bayes score, ANN decision can be assigned a score and used to improve the anti-spam systems. The research proves that the ANN decision is closer to score-based anti-spam result than Naïve-Bayes decision by 7.5%.

VI. Conclusion

According to the result of the experiment, the proposed modeling approach to represent email for ANN can be summarized in the below steps:

1. Extract fixed features into statistical representation as vector of numbers described in table 2.
2. Data preprocessing: From subject and body of email message, remove stop words then apply suitable stemmer based on language of the message. Change some variable data inside text such as telephone numbers, addresses, cities names, counties names, … etc. by a single meaningful word for each category.
3. Build vector of tf*idf using the preprocessed data of subject and body of emails.
4. Calculate L2-norm (Euclidean norm) of tf*idf vector. Assume its name tfidfL2 vector.
5. Use PCA algorithm to reduce tfidfL2 vector. Let its name as pca(tfidfL2) vector.
6. Concatenate vectors SF and pca(tfidfL2) horizontally to make one vector of double-type numbers, it is the email model that can be used as an input to ANN. assume its name eml_SF-pcaTFIDFL2 vector.
7. Build target vector of spam labels of the emails in form of pair of bits. Assume its name SpamLabels vector.

8. Use ANN of a two-layer feed-forward network type, with sigmoid hidden and softmax output neurons. Two neurons are in the output layer. Train the ANN using eml_SF-pcaTFIDFL2 vector as an input and SpamLabels vector as a target. Use different number of neurons in hidden layer until finding the best-suited neurons number. If the performance is still not accepted, eliminate PCA usage and use lower number of neurons in hidden layer.

ANN classification does not work from the first day deployed within any score-based anti-spam system. Real-life example in this research shows that it takes only two days to get enough samples for training. It depends on volume of received emails to the organization under study.

This approach can support any score-based anti-spam system and improve its classification. Like Naïve-Bayes filtering, it is not recommend using ANN as single anti-spam system. Spammers strive to make their spam emails to look like legitimate emails that is difficult to machine or even to human to classify them. therefore, other techniques are used together to fight spam such as public Real-time Black Listing (RBL), SPF (Sender Policy Framework) checks, HELO message, and reverse DNS availability checks.

ACKNOWLEDGMENT

We would like to acknowledge CACBank® of YEMEN. CACBANK provides access to valuable information and systems to process them according to the research scenarios. Without this support, the research could not have been successfully conducted.

REFERENCES

[1] Manning, C. D., Raghavan, P., & Schütze, H. (2009). *An Introduction to Information Retrieval*. England: Cambridge University Press.

[2] Madigan, D. (2005). Statistics and The War on Spam. In D. Nolan, R. Peck, G. Casella, G. W. Cobb, & R. Hoerl, *Statistics: A Guide to the Unknown* (pp. 135-147). Duxbury Press.

[3] Goweder, A. M., Rashed, T., Elbekaie, A. S., & Alhammi, H. A. (2008). An Anti-Spam System Using Artificial Neural Networks and Genetic Algorithms. *ACIT'2008 - the 2008 International Arab Conference on Information Technology*, (pp. 1-8).

[4] Alamlahi, Y., & Ahmed, F. (2007). Sana'ani Dialect to Modern Standard Arabic: Rule-based Direct Machine Translation. In H. R. Arabnia, D. de la Fuente, E. B. Kozerenko, & J. A. Olivas (Ed.), ICAI'11 - The 2011 *International Conference on Artificial Intelligence. II*, pp. 892-895. Las Vegas Nevada: CSREA Press. Retrieved from http://worldcomp-proceedings.com/proc/proc2011/icai.html

[5] Blanzieri, E., & Bryl, A. (2008, March). A Survey of Learning-Based Techniques of Email Spam Filtering. *Journal Artificial Intelligence Review, 29*(1), 63-92.

[6] Creech, G., & Jiang, F. (2012). Semantics Based Multi-layered Networks for Spam Email Detection. *NUMERICAL ANALYSIS AND APPLIED MATHEMATICS ICNAAM 2012: International Conference of Numerical Analysis and Applied Mathematics. 1479*, pp. 1518-1523. Kos, Greece: AIP Publishing. doi:10.1063/1.4756452

[7] Bansod, R., Mangrulkar , R. S., & Bhujade, V. G. (2015). Spam Classification using Artificial Neural Network with Weight Measures. *International Journal of Advanced Computer Technology (IJACT), 4*(6), 68-72.

[8] Synametrics Technologies. (n.d.). *Xeams Web - Main Page*. Retrieved June 10, 2016, from XEAMS official web site: http://www.xeams.com/

[9] Clark, J., Koprinska, I., & Poon, J. (2003). A Neural Network Based Approach to Automated E-mail Classification. *null* (p. 702). IEEE.

[10] Cui, B., Mondal, A., Shen, J., Cong, G., & Tan, K.-L. (2005). On Effective E-mail Classification via Neural Networks. In K. V. Andersen, J. Debenham, & R. Wagner (Ed.), *16th International Conference, DEXA 2005* (pp. 85-94). Copenhagen, Denmark: Springer Berlin Heidelberg. doi:10.1007/11546924_9.

[11] Unitec. (2010, May 27). *Spam email datasets*. (Unitec) Retrieved June 16, 2016, from CSMining Group: http://csmining.org/index.php/spam-email-datasets-.html

[12] Stedfast, J. (2015). *MimeKit*. Retrieved June 23, 2016, from MimeKit: http://www.mimekit.net/

[13] Porter, M. (2006, January). *The Porter Stemming Algorithm*. Retrieved from Martin Porter's Home Page on tartarus.org: http://snowball.tartarus.org/algorithms/english/stemmer.html

[14] Becker, K. (2013, September 13). *TF*IDF in C# .NET for Machine Learning - Term Frequency Inverse Document Frequency*. Retrieved from Primary Objects, Software Development, Programming, AI: http://www.primaryobjects.com/2013/09/13/tf-idf-in-c-net-for-machine-learning-term-frequency-inverse-document-frequency/

Authors' Profiles

Yahya M. Alamlahi received his B.Sc. in Computer Science from Sana'a University, Yemen, and Master of Information Technology from Open University Malaysia, UST Center. His research area includes natural language processing, neural network and artificial intelligence in general. He is ITIL certified and a certified engineer in storage and virtualization from EMC and VMware (EMCSAe, VCP). Currently, he is IT Operations and Systems manager at CACBank® in Yemen.

Abdulrahman A. Muthana received his B.Sc. in Computer Science from Mosul University, Iraq, M.Sc. in Computer Applications from Bangalore University, India and PhD in Information Security from University Putra Malaysia. His research areas includes information security, smartphone security, network security, software security. He is now an Assistant Professor in Faculty of Computer Science and Information Systems, Thamar University, Yemen.

How to cite this paper: Yahya Alamlahi, Abdulrahman Muthana,"An Email Modelling Approach for Neural Network Spam Filtering to Improve Score-based Anti-spam Systems", International Journal of Computer Network and Information Security(IJCNIS), Vol.10, No.12, pp.1-10, 2018.DOI: 10.5815/ijcnis.2018.12.01