Improving Email Spam Detection Using Content Based Feature Engineering Approach

Wadi' Hijawi*†, Hossam Faris*, Ja'far Alqatawna*†, Ala' M. Al-Zoubi*†, Ibrahim Aljarah*
*King Abdullah II School for Information Technology
Business Information Technology Department
The University of Jordan, Amman, Jordan
†Jordan Information Security and Digital Forensics Research Group (JISDF)
Amman, Jordan
w.hijjawi@jisdf.org, {hossam.faris, j.alqatawna, i.aljarah}@ju.edu.jo, a.alzoubi@jisdf.org

***Abstract*—Recently, a wide range of Machine Learning (ML) algorithms have been proposed for building email spam detection models. However, the performance of ML methods highly depends on the extracted features. In this paper, we discuss the most influencing spam features reported in the literature. We also describe the development and implementation of an open source tool that provides a flexible way to extract a large number of features from any email corpus to produce cleansed dataset which can be used to train and test various classification algorithms. A total of 140 features are extracted from SpamAssassin email corpus using the developed tool. Extracted features are used to evaluate four popular ML classifiers and a better results are achieved in comparison with the results of a similar previous study.**

Keywords*—Spam Detection, Feature Extraction Tool, Spam Features, Data Mining, Machine learning.

I. INTRODUCTION

Few years ago, and as a result of opening spam email with a bad link, Aramco - the largest oil company in the world - was a victim of the worst cyberattack ever seen. Consequently, more than 35,000 hard desks in the company's computers were partially deleted or destroyed. With such security incidents, detecting spam email is still representing a challenge to deal with. Usually users do not want to receive spam emails as they waste their time, efforts and resources. More seriously, they may carry malicious contents in a form of attachments or URLs which may lead to dangerous security breaches. Despite the rise of social network platforms and instant messaging applications, email remains the most common mean of communication. According to Symantec, there were 190 billion emails sent each day in 2015 [1]. Moreover, statistics reported by Kaspersky lab showed that the percentage of spam in the global email traffic reached about 60 percent in the third quarter of 2016 [2] . With the ever increasing percentage of received spam messages, email has become a source for wasting time, efforts and resources.

Another serious aspect of spam, is the fact that it can be malicious. It is one of the popular methods used by cybercriminals to lure online victims [3]–[6]. Several deception methods are usually employed by malicious spammers including attachments with double file extension, tiny or obfuscated URLs that point to malicious websites or emotional stories which end up some sort financial fraud and identify theft.

Learning based classifiers are considered a common approach for spam detection in the literature [7]–[11]. The detection process relies on the assumption that spam content has a set of features which can be use to discriminate spam from the legitimate email. However, many factors contribute to the difficulty of accurate spam detection. These include the subjective nature of spam, concept drift, language issues, processing overhead and message delay, and the irregular cost of filtering errors [12]. This implies that a good detection approach should consider in its feature engineering stage the context in which the email is circulated, spammers' social engineering and obfuscation techniques, the language and the needed infrastructure.

In this paper, we construct a comprehensive and a representative collection of spam email features using a very powerful and flexible feature extraction tool developed specifically for processing large email corpus. The produced dataset is used to train and test various classification algorithms. A total of 140 features are extracted from SpamAssassin email corpus. Four popular classifiers are evaluated and when the results are compared with those obtained in an earlier study, a better results are obtained.

The rest of this paper is organized as follows. Section II reviews some of the related works, section III discusses common spam features reported in the literature. The development of the feature extraction tool is presented in section IV. Experiment and results are presented in section V, followed by the conclusion in section VI.

II. RELATED WORK

In this section we review four important related studies which represent the foundation of the current study. We analyzed these studies to construct a comprehensive set of spam features which will be discussed in section III.

978-1-5090-5969-0/17/$31.00 ©2017 IEEE

Tran et al. [13] proposed 58 features to predict spam emails that contain malicious attachments and URLs using only the content of the email's subject and body. They classified these features to four classes: Header features (six features), Text features (forty-six features), Attachment features (four features) and URL features (two features).They used two datasets (Habul DS and Botnet DS) to rank these features using the information entropy scores generated from the Random Forest classifier to find the set of the most importance features to detect spam emails with malicious attachments and the other set of the most importance features to detect spam emails with malicious URLs.

Three classifiers (Naïve Bayes, Random Forest and Support Vector Machine) were used with the prior datasets to evaluate the features. They found that these features success for detecting spam emails with malicious attachments using the content of the email's subject and body with high AUC-PR and ACC scores but on the other hand, it does not match the expectations for detecting spam emails with malicious URLs using the content of the email's subject and body only.

Shams and Mercer [14] used four benchmark corpora (CSDMC2010, Spam Assassin, LingSpam, and Enron-Spam) with 40 features to propose a novel spam classification method. They grouped these features to three groups: Traditional features, Text features and Readability features. They used Borutal[1], a feature importance-measuring algorithm to rank the features and found that the Spam Words, Grammar Errors, Spelling Errors and Language Errors features are the most important features. Additionally, five classifiers (Random Forest, BAGGING, ADABOOSTM1, Support Vector Machine and Naïve Bayes) were used to identify the importance for each group of features; they used the traditional group as a baseline and calculated the ham misclassification rate (FPR) for each classifier. Then they compared these evaluation results with that calculated for each groups added to baseline and for all set of features and found that the optimal value for FRP was reached when using the all set of features. These five classifiers were used with the four-benchmark datasets and seven evaluation measures (Precision, Recall, F-score, Accuracy, FPR, FNR and AUC) were reported for each classifier and dataset. They found that the BAGGING classifier performs the best.

Alqatawna et al. [15] studied the effect of adding malicious features for improving the evaluation results for four classical spam emails detection classifiers:C4.5 decision trees, MLP, Naïve Bayes and Random forests. They used imbalance data dataset (Spam Assassin) with 90 features to develop the four models using the prior classifiers. These features were categorized to nine categories: Header based, Character based, Word based, Syntactic features, Structural features, Particular content (word/character), Size, Links related features and Attachment related features. Using three evaluation metrics (Accuracy, Precision and Recall), a comparison between the results before and after adding the malicious features reported that these features has significantly improved the ability of the classifiers to detect spam emails.

Al-Shboul et al. [16] proposed 35 features that were grouped to two groups: E-mail body features and Readability features. They reported the Accuracy, Precision, Recall, Hit rate and F-score for each features group and for all features set using Spam Assassin and CSDMC2010 datasets and using three classifiers: k-Nearest Neighbour, Decision Trees and Random Forest. The results showed that these features reported better results than other works with the same email corpora and with the same classifiers. Additionally, they reported the same evaluation measures after using the same datasets and combined the three classifiers in various voting schemes. They found that the results were better than the results obtained by the individual classifiers.

III. SPAM FEATURES

Based on the analysis of the spam features found in the related studies in the literature, we classify the features into three main categories: header features, payload (body) features, and attachment features. Figure 1 shows the hierarchy of the features categories. In the following subsections, we provide a description of each category and its related features.

A. Header features

The email header is an essential element of any email message, which consists of a set of important features that helps in email delivery. These features grouped into two classes, namely; email's metadata and subject. Table I shows twenty-five header features that are collected from the literature with brief descriptions.

B. Payload features

These features are categorized into three sub categories: Email body features, Readability features, and Lexical diversity features.

1) Email body features: The e-mail body contains unstructured data such as images, HTML tags, text, and other objects. This features set contains 59 features, which are described in Table II.

2) Readability features: Readability features represent the difficulty properties of reading a word, a sentence or a paragraph in the given email's body [14], [16]. Readability features are extracted based on the syllables — sequence of speech sounds — , which are used to distinguish between the simple and complex words. This set of features contains 23 features that measure the difficulty of reading the email's body. The readability features are discussed as follows:

- Number of simple words features (with and without stopwords), such as each word has at most two syllables.

[1] http://cran.r-project.org/web/packages/Boruta/index.html

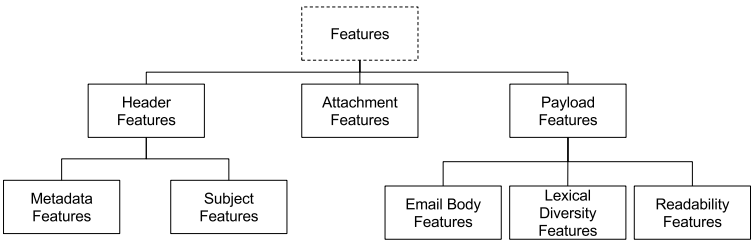

Fig. 1. Hierarchy of features groups.

TABLE I. THE HEADER FEATURES

ID	Feature Details	Type	Studies	ID	Feature Details	Type	Studies
1	Year	Metadata	[15]	26	Replay to MIL?	Metadata	[15]
2	Month	Metadata	[15]	27	Replay to Yahoo?	Metadata	[15]
3	Day	Metadata	[13] , [15]	28	Replay to AOL?	Metadata	[15]
4	Hour	Metadata	[13] , [15]	29	Replay to Gov?	Metadata	[15]
5	Minute	Metadata	[13] , [15]	30	X-Mailman-Version	Metadata	[15]
6	Second	Metadata	[13] , [15]	31	Exist Text/Plain?	Metadata	[15]
7	From Google?	Metadata	[15]	32	Exist Multipart/Mixed?	Metadata	[15]
8	From AOL?	Metadata	[15]	33	Exist Multipart/Alternative?	Metadata	[15]
9	From Gov?	Metadata	[15]	34	Number of characters.	Subject	[13]
10	From HTML?	Metadata	[15]	35	Number of capitalised words.	Subject	[13]
11	From MIL?	Metadata	[15]	36	Number of words in all uppercase.	Subject	[13]
12	From Yahoo?	Metadata	[15]	37	Number of words that are digits.	Subject	[13]
13	From Example?	Metadata	[15]	38	Number of words containing only letters.	Subject	[13]
14	To Hotmail?	Metadata	[15]	39	Number of words containing letters and number.	Subject	[13]
15	To Yahoo?	Metadata	[15]	40	Number of words that are single letters.	Subject	[13]
16	To Example?	Metadata	[15]	41	Number of words that are single digits.	Subject	[13]
17	To MSN?	Metadata	[15]	42	Number of words that are single characters.	Subject	[13]
18	To Localhost?	Metadata	[15]	43	Max ratio of uppercase letters to lowercase letters of each word.	Subject	[13]
19	To Google?	Metadata	[15]	44	Min of character diversity of each word.	Subject	[13]
20	To AOL?	Metadata	[15]	45	Max of ratio of uppercase letters to all characters of each word.	Subject	[13]
21	To Gov?	Metadata	[15]	46	Max of ratio of digit characters to all characters of each word.	Subject	[13]
22	To MIL?	Metadata	[15]	47	Max of ratio of non-alphanumeric characters to all characters of each word.	Subject	[13]
23	Count of "To" Email	Metadata	[13]	48	Max of the longest repeating character.	Subject	[13]
24	Replay to Google?	Metadata	[15]	49	Max of the character lengths of words.	Subject	[13]
25	Replay to Hotmail?	Metadata	[15]	-	-	-	-

- Number of complex words features (with and without stopwords), such as each word contains three or more syllables
- Word length features (with and without stopwords) based on the number of syllables, and the number of the words in the text.
- Fog Index (FI) features (with and without stopwords), which are the most popular readability measure that used estimate the years of education experience to understand the text at glance.
- Flesch Reading Ease Score (FRES) features (with and without stopwords), which are used to asses the textual difficulty.
- SMOG index features (with and without stopwords), which are used to measure the difficulty of the text writing.
- FORCAST index features (with and without stopwords), which are used to measure the reading skills of the text that contains high percentage of simple words.
- Flesch-Kincaid Readability Index (FKRI) features (with and without stopwords), which are similar to the previous features, but with different weighting factors.
- Simple Word FI features (with and without stopwords), which are similar to the Fog Index but with the simple words.
- Inverse FI features (with and without stopwords).
- SMOG-I feature, which is used to measure the difficulty of the text writing.
- Automated Readability Index (ARI), which are used to measure the understandability of a given text.
- Coleman-Liau Index (CLI), which is similar ARI, but with of syllables factor instead of characters factor.

3) Lexical diversity features: Lexical Diversity Features are extracted based on the vocabulary size in the text, where the word occurrences are counted using different constraints [13], [17], [18]. This set of features contains seven features as follows:

- Vocabulary Richness, which represents the number of distinct words in a text.
- Hapax legomena or V(1,N): represents the number of words occurring one time in the text with N words.
- Hapax dislegomena or V(2,N): represents the number of words occurring two times in the text with N words.
- Entropy measure, which asses the average amount of information in the text.
- YuleK, which is a text characteristic measure that is independent of text length.

TABLE II. THE EMAIL BODY FEATURES

ID	Feature Details	Studies	ID	Feature Details	Studies
1	Count of Spam Words	[14] , [15] , [16]	31	Number of question marks	[15] , [16]
2	Count of Function Words	[14] , [16]	32	Number of multiple question marks	[15]
3	Count of HTML Anchor	[13] , [14] , [15] , [16]	33	Number of exclamation marks	[15] , [16]
4	Count of Unique HTML Anchor	[13] , [15]	34	Number of multiple exclamation marks	[15]
5	Count of HTML Not Anchor	[14]	35	Number of colons	[15] , [16]
6	Count of HTML Image	[16]	36	Number of ellipsis	[15] , [16]
7	Count of HTML All Tags	[14] , [16]	37	Total number of sentences	[14] , [15] , [16]
8	Count of Alpha-numeric Words	[13] , [14] , [16]	38	Total number of paragraphs	[15]
9	TF-ISF	[14]	39	Average number of sentences per paragraph	[15]
10	TF•ISF without stopwords	[14]	40	Average number of words pre paragraph	[15]
11	Count of duplicate words.	[16]	41	Average number of character per paragraph	[15]
12	Minimum word length	[16]	42	Average number of word per sentences	[15]
13	Count of lowercase letters	[16]	43	Number of sentence begin with upper case	[15]
14	Longest sequence of adjacent capital letters	[15] , [16]	44	Number of sentence begin with lower case	[15]
15	Count of lines	[15] , [16]	45	Character frequency "$"	[15] , [16]
16	Total number of digit character	[15]	46	Number of capitalized words.	[13]
17	Total number of white space	[15]	47	Number of words in all uppercase.	[13]
18	Total number of upper case character	[15] , [16]	48	Number of words that are digits.	[13]
19	Total number of characters	[13] , [15]	49	Number of words containing only letters.	[13]
20	Total number of tabs	[15]	50	Number of words that are single letters.	[13]
21	Total number of special characters	[15]	51	Number of words that are single digits.	[13]
22	Total number of alpha characters	[15]	52	Number of words that are single characters.	[13]
23	Total number of words	[15] , [16]	53	Max ratio of uppercase letters to lowercase letters of each word.	[13]
24	Average word length	[15] , [16]	54	Min of character diversity of each word.	[13]
25	Words longer than 6 characters	[15]	55	Max of ratio of uppercase letters to all characters of each word.	[13]
26	Total number of words (1 - 3 Characters)	[15]	56	Max of ratio of digit characters to all characters of each word.	[13]
27	Number of single quotes	[15] , [16]	57	Max of ratio of non-alphanumeric characters to all characters of each word.	[13]
28	Number of commas	[15] , [16]	58	Max of the longest repeating character.	[13]
29	Number of periods	[15] , [16]	59	Max of the character lengths of words.	[13] , [16]
30	Number of semi-colons	[15] , [16]	-	-	-

- SichelS, which is the ratio of dislegomena to the number of distinct words.
- Honore, which is the ratio of hapax legomena (V (1,N)) to the vocabulary size N.

C. Attachment Features

We use two features related to attachments: Number of all attachment files in an email and Number of unique content types of attachment files in an email.

IV. Feature extraction tool

We implement an open source and flexible tool that can be used to extract the previously discussed features from any email corpus with emails saved in EML format[2]. EML file extension is one of the most common file extension used by email applications such as Outlook, Thunderbird and Gmail.

The tool splits the emails' into their main parts which are: "From", "To", "CC", "BCC", "Subject", "Text Body", and "HTML Body" and save each of them to an output file. Then the tool extracts the selected features from the corresponding email part and save them in another file in CSV format. If any error occurred during the extraction process, an error file will be generated.

The tool is designed as a group of tab pages; each tab page represents a feature category and contains checkboxes with unique tags for selecting the desired features to extract. Additionally, a folder browser is used to select the corpus folder and a user-friendly progress bar is used to view the current status for the extraction process.

On the other hand, the source code for features extraction process is organized in one main class with a controller to call each selected feature's procedure, which will use each email object's attributes and methods to extract the desired features. The variables, packages and emails objects are initialized in separated classes. Figure 2 describes the implementation structure.

Different external libraries are utilized in the implementation which include: "Microsoft CDO for Windows 2000" COM Library to deal with each email file as an object and save the emails' parts into an output file. For features extraction development, we used the following libraries:

- HTML Agility Pack[3].
- IKVM.NET[4].
- Stanford.NLP.CoreNLP[5].
- RegularExpressions[6].
- LINQ (.NET Language-Integrated Query)[7].

The simplified design and implementation structure make the tool easy to use and enhances the ability to add more features to the tool later by adding a checkbox with a unique tag for the new feature, adding the extraction procedure in the main class and linking between this procedure's name and checkbox's tag in the controller. Additionally, passing parameters for each email as an object with a unique tag for each feature into the extraction procedures, this

[2]The source code of the proposed extraction tool is publicly available at the following website: https://github.com/WadeaHijjawi/EmailFeaturesExtraction

[3]External package to parse HTML. Available at:https://htmlagilitypack.codeplex.com/

[4]External package to enable Java and .NET interoperability. Available at:https://www.ikvm.net/

[5]External package to provides a set of natural language analysis tools. Available at:https://www.nuget.org/packages/Stanford.NLP.CoreNLP/

[6]Internal package to work with text such as search a pattern within text.

[7]Internal package to query and access data in objects such as XML documents, databases.

allows dealing with this procedures as independent problems. Moreover, using the internal and external packages reduces the source code lines and make the implementation easier to extend and modify.

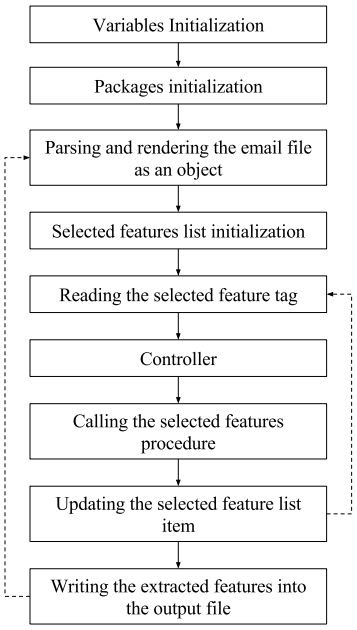

Fig. 2. The implementation structure.

V. EXPERIMENTS AND RESULTS

In our experiments we compare the performance of the four popular classifiers (i.e. J48 decision tree, Multilayer Perceptron (MLP), Naïve Bayes and Random Forest) when trained based on extracting all features described in this work with the results obtained in [15]. To make the comparison fair we use the same Email corpus (Spam Assassin Public) to extract the features. We have also unified all parameters settings for all classifiers and used same training and testing methodology which is a 5-folds cross-validation. It is important to note also that the investigated corpus has an imbalanced class distribution of 5051 Ham Emails and 1000 Spam Emails.

Three evaluation measures are used to evaluate the developed spam detection models: Accuracy rate, Precision and Recall to evaluate our results. These measures can be calculated based on the confusion matrix shown in Figure 3 as given in Equations 1, 2 and 3, respectively. Each classifier is applied five times and then the average and standard deviation for each evaluation measure is calculated. Table III summarizes these results. It can be noticed that the best classification results were obtained using the Random Forest classifier followed by the J48 decision tree. Overall, all classifiers except Naïve Bayes achieved very high evaluation results in terms of accuracy, precision and recall.

$$Accuracy = \frac{TP + TN}{TP + FN + FP + TN}. \quad (1)$$

		Predicted class	
		Positive	Negative
Actual class	Positive	True Positive (TP)	False Negative (FN)
	Negative	False Positive (FP)	True Negative (TN)

Fig. 3. Confusion matrix

$$Precision = \frac{TP}{TP + FP}. \quad (2)$$

$$Recall = \frac{TP}{TP + FN}. \quad (3)$$

Finally, the obtained results based on the implemented features in the proposed tool are compared with the results reported in [15]. In the latter work, there are two groups of features the first includes malicious related features which we will refer to as "$Group1$" while the second is a group of spam features but without the malicious related features, this group is referred to as "$Group2$". Figures 4, 5, 6 and 7 depicts the comparison between the evaluation results obtained on the implemented features in this work and $Group1$ and $Group2$ features. According to the figures, it can be noticed that our implemented set of features have improved the performance of the classifiers specially in the case of J48, Random Forests and MLP.

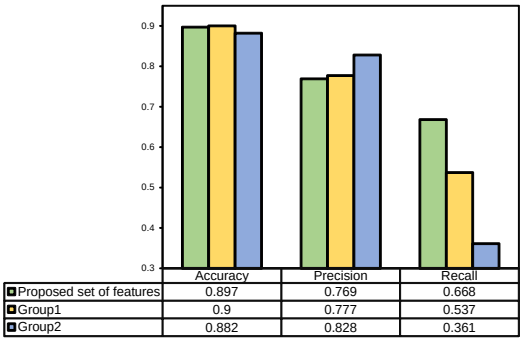

Fig. 4. Naïve Bayes results

VI. CONCLUSION

In this work, we discussed the development of spam detection models based on a comprehensive list of the most common spam features. The study also presents the development of an open-source simple tool for extracting email features. The goal of the tool is to ease the task of processing email corpus and extract large number of representative features. In this work, 140 features were extracted. Our experimental results showed that these features improved spam detection rates based on different popular machine

TABLE III. EVALUATION RESULTS OF PROPOSED FEATURES

6061 instances		J48	Random Forest	MLP	Naïve Bayes
Accuracy		98.4%(±0.073)	99.3%(±0.121)	97.6%(±0.307)	89.7%(±0.033)
Ham (5051)(83%)	Precision	0.99±0.001	0.992±0.001	0.979±0.002	0.923±0.0007
	Recall	0.991±0.0005	0.999±0.0008	0.991±0.002	0.956±0.0008
Spam (1000)(17%)	Precision	0.957±0.003	0.996±0.002	0.956±0.012	0.769±0.085
	Recall	0.95±0.005	0.963±0.006	0.896±0.014	0.668±0.161

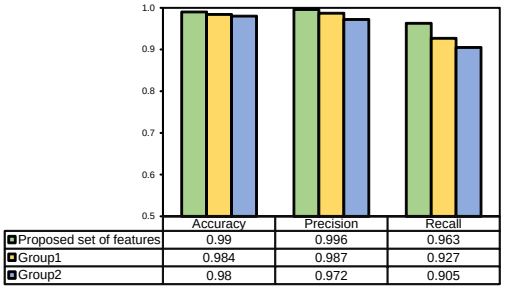

Fig. 5. Random Forests results

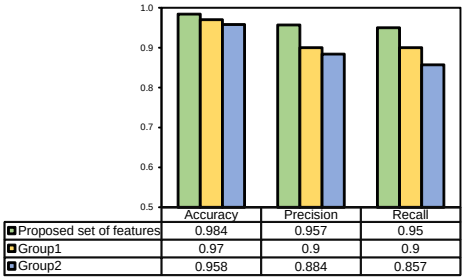

Fig. 6. J48 decision tree results

learning algorithms. For future work, more features are planned to be added to the extraction tool. Moreover, the influence of spam features can be studied based on different spam corpora.

REFERENCES

[1] Symantec, *2016 Internet Security Threat Report*, 2016 (accessed May 20, 2017).

[2] Kaspersky, *Spam and phishing in Q3 2016*, 2016 (accessed May 20, 2017).

[3] J. Alqatawna, A. Madain, A. M. Al-Zoubi, and R. Al-Sayyed, "Online social networks security: Threats, attacks, and future directions," in *Social Media Shaping e-Publishing and Academia*, pp. 121–132, Springer, 2017.

[4] J. Alqatawna, A. Hadi, M. Al-Zwairi, and M. Khader, "A preliminary analysis of drive-by email attacks in educational institutes," in *Cybersecurity and Cyberforensics Conference (CCC)*, pp. 65–69, IEEE, 2016.

[5] R. A. Halaseh and J. Alqatawna, "Analyzing cybercrimes strategies: The case of phishing attack," in *Cybersecurity and Cyberforensics Conference (CCC)*, pp. 82–88, IEEE, 2016.

[6] A. Zaid, J. Alqatawna, and A. Huneiti, "A proposed model for malicious spam detection in email systems of educational institutes," in *Cybersecurity and Cyberforensics Conference (CCC)*, pp. 60–64, IEEE, 2016.

[7] A. M. Al-Zoubi, J. Alqatawna, and H. Faris, "Spam profile detection in social networks based on public features," in *2017 8th International Conference on Information and Communication Systems (ICICS)*, pp. 130–135, April 2017.

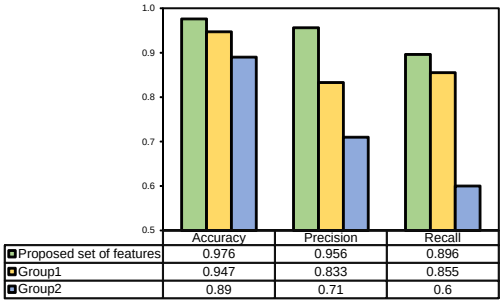

Fig. 7. MLP results.

[8] T. S. Guzella and W. M. Caminhas, "A review of machine learning approaches to spam filtering," *Expert Systems with Applications*, vol. 36, no. 7, pp. 10206 – 10222, 2009.

[9] A. Rodan, H. Faris, and J. Alqatawna, "Optimizing feedforward neural networks using biogeography based optimization for e-mail spam identification," *International Journal of Communications, Network and System Sciences*, vol. 9, no. 1, p. 19, 2016.

[10] H. Faris, I. Aljarah, and J. Alqatawna, "Optimizing feedforward neural networks using krill herd algorithm for e-mail spam detection," in *Applied Electrical Engineering and Computing Technologies (AEECT), 2015 IEEE Jordan Conference on , vol., no., pp.1-5*, pp. 1–5, IEEE, 2015.

[11] H. Faris, I. Aljarah, and B. Al-Shboul, "A hybrid approach based on particle swarm optimization and random forests for e-mail spam filtering," in *International Conference on Computational Collective Intelligence*, pp. 498–508, Springer, 2016.

[12] 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 Syst. Appl.*, vol. 39, pp. 12487–12500, Nov. 2012.

[13] K.-N. Tran, M. Alazab, and R. Broadhurst, "Towards a feature rich model for predicting spam emails containing malicious attachments and urls," in *Eleventh Australasian Data Mining Conference Canberra, ACT*, vol. 146, 2013.

[14] R. Shams and R. E. Mercer, "Supervised classification of spam emails with natural language stylometry," *Neural Computing and Applications*, vol. 27, no. 8, pp. 2315–2331, 2016.

[15] J. Alqatawna, H. Faris, K. Jaradat, M. Al-Zewairi, and A. Omar, "Improving knowledge based spam detection methods: The effect of malicious related features in imbalance data distribution," *International Journal of Communications, Network and System Sciences*, vol. 8, no. 5, pp. 118–129, 2015.

[16] B. A. Al-Shboul, H. Hakh, H. Faris, I. Aljarah, and H. Al-sawalqah, "Voting-based classification for e-mail spam detection," *Journal of ICT Research and Applications*, vol. 10, no. 1, pp. 29–42, 2016.

[17] F. J. Tweedie and R. H. Baayen, "How variable may a constant be? measures of lexical richness in perspective," *Computers and the Humanities*, vol. 32, no. 5, pp. 323–352, 1998.

[18] W. H. Choi, "Finding appropriate lexical diversity measurements for small-size corpus," in *Applied Mechanics and Materials*, vol. 121, pp. 1244–1248, Trans Tech Publ, 2012.