OPEN ACCESS
Indo-JC
Indonesia Journal of Computing
ISSN 2460-9056
socj.telkomuniversity.ac.id/indojc

Ind. Journal on Computing
Vol. 4, Issue. 1, March 2019. pp. 109-118
doi:10.21108/indojc.2019.4.1.284

# Study on the Effect of Pre-processing Methods for Spam Email Detection

Fariska Zakhralativa Ruskanda [#1]

*# Department of Informatics, Widyatama University*
*Jl. Cikutra 204A, Bandung, Indonesia*

[1] fariska.zr@widyatama.ac.id

## Abstract

The use of email as a communication technology is now increasingly being exploited. Along with its progress, email spam problem becomes quite disturbing to email user. The resulting negative impacts make effective spam email detection techniques indispensable. A spam email detection algorithm or spam classifier will work effectively if supported by proper pre-processing steps (noise removal, stop words removal, stemming, lemmatization, term frequency). Most studies spam classifier do not carry out a more detailed study of the effect of the combination of preprocessing methods on the classification results. This research studies the effect of pre-processing steps on the performance of supervised spam classifier algorithms. Experiments were conducted on two widely used supervised spam classifier algorithms: Naïve Bayes and Support Vector Machine. The evaluation is performed on the Ling-spam corpus dataset - collection of total 962 spam and ham messages from linguistic mailing lists that are publicly available - and uses evaluation metrics: accuracy. The experimental results show that different combination of pre-processing methods give different effects to different classifier. The results of this study can be used to improve accuracy when using spam classifiers.

**Keywords:** spam email detection, pre-processing, supervised classifier

## Abstrak

Email sebagai teknologi komunikasi kini semakin banyak digunakan. Seiring dengan perkembangannya, masalah email spam menjadi sangat mengganggu pengguna email. Dampak negatif yang dihasilkan membuat teknik pendeteksian email spam yang efektif sangat diperlukan. Algoritma pendeteksian email spam atau pengklasifikasi spam akan bekerja secara efektif jika didukung oleh langkah-langkah praproses yang tepat (penghapusan *noise*, penghilangan *stop word*, *stemming*, *lemmatization*, frekuensi kata). Kebanyakan penelitian pada pengklasifikasi spam tidak melakukan penelaahan lebih rinci terhadap pengaruh kombinasi metode praproses pada hasil klasifikasi. Penelitian ini mempelajari efek langkah-langkah praproses pada kinerja algoritma pengklasifikasi spam yang diawasi. Eksperimen dilakukan pada dua algoritma pengklasifikasi spam yang banyak digunakan: Naïve Bayes dan Support Vector Machine. Evaluasi dilakukan pada dataset korpus Ling-spam, yaitu koleksi 962 email spam dan ham dari milis linguistik yang tersedia untuk publik, dan menggunakan metrik evaluasi berupa akurasi. Hasil eksperimen menunjukkan bahwa langkah-langkah praproses yang berbeda memberikan efek yang berbeda untuk pengklasifikasi yang berbeda. Hasil studi ini dapat digunakan untuk meningkatkan akurasi pada saat menggunakan pengklasifikasi spam.

**Kata Kunci:** deteksi spam email, praproses, pengklasifikasi terawasi

## I. Introduction

THE digital age and expansion of the World Wide Web (WWW) has resulted in a flood of communications and information on the internet. Anyone can communicate and share any information and anytime easily and quickly. One of the media is via email. As one of the innovations in communication, email makes people can communicate quickly and easily. However, along with its development, some email related challenges arose, such as spam email, email spoofing, email bombing, phishing email, and others. Among these things, one of the most annoying is spam email. Spam is an email containing crafted information that is usually sent to many people by an unknown party, without their consent [1]. The purpose

Received on Jan 2019. Accepted on March 2019

of spam is usually to advertise a product, promotion, bait for a fraud scheme, or computer malware that aims to infiltrate the receiving computer [1]. Spreading of spam gives some negative consequences. Among them are cyber-crime, network resource consumption, human resource consumption, lost email, etc.

Spam filter is an automated classification of emails that recognizes spam and legitimate (ham) email [2]. It utilizes spam classifier that usually uses one or both approaches: rule-based (hand-crafted) or machine learning. A lot of research is conducted on both approaches, however, there is still little attention discussing the pre-processing method that became the initial stage before the email classification process was performed. As known, to detect spam emails, some standard Natural Language Processing (NLP) pre-processing steps need to be done, such as noise removal, stop words removal, stemming, lemmatization, term frequency. It is necessary to prepare the emails to be ready for analysis. These pre-processing steps can affect the overall performance of the detection algorithm. The varied combination of pre-processing methods used in many spam detection studies motivates us to conduct further investigations that other researchers have not yet done.

The study of the effect of using various combinations of pre-processing steps on some spam detection algorithms is proposed in this paper. We selected two spam detection classifiers that represented different approaches: Naïve Bayes and Support Vector Machine. In this study, we try to answer some research questions as follows:

1) Which combination of pre-processing methods should be used to provide accurate classification results?
2) Is the combination of the pre-processing method used is appropriate for all classifiers? Or does each classifier require a different combination of pre-processing methods? What caused this?

To evaluate the performance of the detection algorithm after the pre-processing step applied, we used Ling-spam corpus – a publicly available collection of total messages from linguistic mailing lists. This is a balanced spam email dataset – the condition where equal instances for both classes: spam and ham. The organization of this paper is as follows: Section two discusses related research. The third section explains some pre-processing methods. Section four discusses the experiments and discussions. The final section deals with conclusions.

## II. Literature Review

The approach method for spam detection generally consists of two types: hand-crafted classifier and machine learning classifier [1]. Almost all spam filters currently use either or both of these approaches. The purpose of both is the same - classify emails to spam or ham (non-spam).

Among the hand-crafted classifier approaches are human classifier, ad hoc classifier, rule-based filtering, whitelists, blacklists, and collaborative spam filtering [1]. Human classifier uses humans as the party that identifies email spam, one of them through a summary line containing the sender's name, subject, and delivery time. Yerazunis conducted email groupings and produced a disagreement rate of 0.16% [3]. However, this method has weakness in term of processing time and labor cost. The second approach is the ad hoc classifier. This technique requires the existence of a single user or system administrator tasked with grouping spam email into a block or quarantine based on special criteria [1]. The problem with this technique is the existence of some rules / criteria that often generate false positives, such as errors in the “from” rule.

The other hand-crafted approach used automatic mechanism is SpamAssassin[1], the open-source spam filter which is the rule-based classifier pioneer. Rules used are hand-crafted rules expressed by using a special formal notation, and are designed to be used on a variety of systems. The word list approach is used in whitelists and blacklists. Whitelist is a list of senders whose emails must be received, regardless of content. The problem of this way is when email spoofing occurs [4]. Blacklist is a list of senders, domains, or IP addresses that are considered as spam senders. Examples of systems using this are the Real-time Blackhole List (RBL) for the Mail Abuse Prevention System (MAPS)[2]. This method provides effective results to filter spam, but also presents new challenges. Spam is sent from many sources, and the success of a blacklist

[1] http://spamassassin.apache.org
[2] http://www.mail-abuse.com/

depends on its ability to detect the source of the email. Another approach leverages the knowledge of email recipients who have already received spam email: collaborative spam filtering. This method is the process of capturing, recording, and querying spam judgement [5]. The decisive component of this system is the real-time database of known-spam messages that can be updated and queried by multiple users.

Another type of approach in spam email detection is the machine learning classifier based approach. One of the method widely used is the Naïve Bayes classifier [6], [7]. This method is a type of supervised learning and based on the word probability value. Another widely used classification method is K-nearest neighbor (KNN). This method is an example-based classifier, which is training by determining the closest and appropriate cluster for each data. The criteria of each cluster are not predetermined [8]. Another widely used approach lately is the artificial neural network (ANN) method. This approach uses a computational model based on the principles of biological neural networks. The training process is supervised by processing input data through interconnected artificial neurons [9], [10]. In addition to the above three methods, a fairly widely used method is the Support Vector Machine (SVM) classifier [7], [11]. SVM uses the hyperplane concept that determines the grade value. This algorithm looks for an optimal hyperplane with maximal margin value for both classes.

In spam filtering, a pre-processing step is required for data preparation before email classification process using a classifier. Some papers use different pre-processing techniques. Khan and Qamar use pre-processing steps in the form of transform case, tokenization, token filter (by length), stemming, and stop words filter [12]. While Bluszcz et al. using pre-processing steps in the form of cleaning HTML tags and items normalization (currency symbols, email addresses, URLs) [11]. Wei-chih and Yu use pre-processing steps in the form of lemmatization and stop words removal, and transform case [13]. Trudgian and Yang use the pre-processing method of transform case, tokenization, non-alphabetical tokens removal, and token filter (by length) [14]. Other pre-processing steps used by Rathod and Pattewar on their spam detection research using Bayesian Classifier are HTML tag removal, stop word removal, tokenization, and word frequency [15]. Variations in the use of existing pre-processing methods motivate us to conduct studies regarding appropriate pre-processing methods for spam email detection.

## III. Pre-processing Methods in Email Spam Detection

The pre-processing steps that are widely used in some spam detection research include: noise removal, stemming, lemmatization, frequency term, and frequency term-inverse document frequency (TF-IDF) [1]. Below is the explanation of each step.

### A. Noise Removal

Noise is any piece of text that is not relevant to the context of the data and the end result. In spam email, noise include stop words, alpha numeric word, and punctuation. An example of a stop word is “a”, “is”, “this”, “it”. While the example of alpha numeric words is “lucky123”, “80”, “7a32b”. An example of punctuation is “,”, “?”, “!”. A general approach to noise removal is to prepare a list of noise words/tokens (stop words, punctuation), then iterates the text by tokens, removes tokens which are present in noise word list.

### B. Stemming

Stemming is one way to normalize the word form. Stemming is a basic rule-based process for removing suffixes from words ("ing", "ly", "es", "s", etc). For example the words "argue", "argued", "argues", "arguing", and "argus", have stem "argu".

### C. Lemmatization

Lemmatization is another way of normalizing the word form. In contrast to stemming, lemmatization is an organized procedure for obtaining the root form of the word, by utilizing vocabulary (dictionary) as well as morphological analysis. The example of lemma for the word "better" is "good".

### D. Term Frequency and TF-IDF

The term frequency states how many words exist in a document (set of documents). This value indicates how important a word is in a document (or set of documents).

Term Frequency - Inverse Document Frequency (TF-IDF) is another way to measure how important a word is in a document. TF-IDF is obtained by multiplying the value of the term frequency by the inverse document frequency value (the document frequency value of a word in the whole document) to filter out common words.

## IV. Experiments

This section discusses the datasets and evaluation metrics used in experiments, spam detection algorithms, and overall experimental results on pre-processing methods. As shown in Figure 1, the contents of the email on the dataset will be pre-processed in advance with one or more pre-processing steps. The pre-processing results are then passed to the spam classifier. This module works to classify whether email is spam or not. In the final stage, classifier performance evaluation is performed.

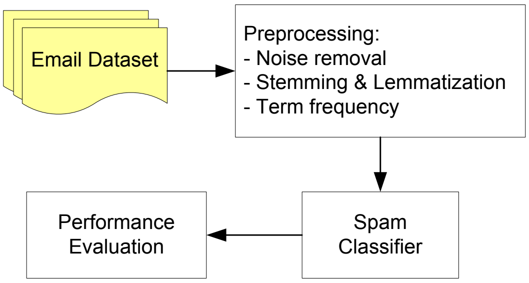

Fig. 1. Overview of Experiment in Pre-processing for Spam Email Detection

### A. Dataset and Evaluation Metric

TABLE I
Dataset Statistics

| **Ling-Spam Email Dataset** | | **Number of Emails** | **Average word count per email** |
|---|---|---|---|
| Training set | Ham | 351 | 379.6 |
| | Spam | 351 | 716.9 |
| Test set | Ham | 130 | 473.9 |
| | Spam | 130 | 706.8 |

The dataset used for the experiments is Ling-spam Corpus [16], a collection of spam and ham emails from a mailing list on linguistics. This dataset is divided into two parts, training set and test set, each containing total 702 emails and total 260 email. Both parts contain balanced distributed spam and ham (Table 1). For evaluating the performance of the classifier algorithm, metrics in the form of accuracy is used. Accuracy is a ratio of correctly predicted/classified (true positives and true negatives) emails to the total emails in dataset.

$$Accuracy = \frac{TruePositives\ +\ TrueNegatives}{Total\ Emails} \tag{1}$$

*B. Spam Detection Algorithms*

For evaluating the effect of pre-processing steps to the performance of spam detection algorithms, we used two algorithms (classifiers): Naïve Bayes and Support Vector Machine [17], [18].

*1) Naïve Bayes:* Naïve Bayes is a probabilistic classifier [19]. This means for a document (email) *d*, out of all classes $c \in C$ the classifier returns the class $\hat{c}$ which has a maximum posterior probability given the document.

$$\hat{c} = \underset{c \epsilon C}{\operatorname{argmax}} P(c|d) \tag{2}$$

An email document *d* consisting of many words $x_1, x_2, \ldots, x_n$, its posterior probability value is

$$\hat{c} = \underset{c \epsilon C}{\operatorname{argmax}} P(x_1, x_2, \ldots, x_n|c)\, P(c) \tag{3}$$

To simplify calculations, Naïve Bayes classifier uses two simplifying assumptions. The first assumption is the bag of words, i.e. email is represented as bag-of-words – an unordered set of words that do not take its positions, only keep the word's frequency in email. The assumption used is the position of the word does not matter. The second assumption is the conditional independence assumption that the probabilities $P(x_i\ |c)$ are independent given the class *c* and can be naïvely multiplied as follows:

$$P(x_1, x_2, \ldots, x_n|c_j) = \prod_i P(x_i|c_j) \tag{4}$$

*2) Support Vector Machine:* Support Vector Machine (SVM), is one of supervised learning techniques, which is a combination of linear learning machine and kernel function [11]. Given a set of training data, each labeled as a class/category, the SVM algorithm builds a model by defining a new instance between the existing data: a hyperplane. This algorithm maximizes the margin between the hyperplane and the nearest data points by assigning weights *w* to the feature vector. This makes it a non-probabilistic binary linear classifier. The superiority of this algorithm for spam detection is its reliability and its ability to handle large feature spaces. This algorithm does not try to minimize the error rate, but separates the pattern in high dimensional space, making it insensitive to the class size.

*C. Compared Pre-processing Methods and Discussion*

Pre-processing methods compared in this study are: noise removal, stemming, lemmatization, and term frequency (TF-IDF). In the experiments conducted, the option of not using against using one of the pre-processing method is compared. Then, experiments were also conducted on a combination of pre-processing methods.

In this study, commonly-used software for Natural Language Processing: NLTK[3] and scikit-learn[4], are utilized. These tools are packages for Python that provides a large set of functions and corpus for natural language processing. For stemming process, three kinds of stemmer that are part of NLTK are used: Snowball, Porter, and Lancaster stemmer. As for lemmatization, WordNet Lemmatizer is used.

[3] http://nltk.org/

[4] http://scikit-learn.org/

In these experiments, feature extracted from dataset is bag-of-words, or we call it here: dictionary. Before pre-processing steps, the size of the dictionary is more than twenty thousand of words. The size of dictionary after the pre-processing steps is listed in Table 2.

TABLE II
Dictionary Size After Pre-processing Methods and Discussion

| Pre-processing Method | Size of Dictionary (Bag-of-Words) |
|---|---|
| No pre-processing | 20488 |
| Stop words removal | 20416 |
| Punctuation marks removal | 20422 |
| Alpha-numeric words removal | 16907 |
| Stemming | 18194 |
| Lemmatization | 5921 |

*1) Noise Removal Experiments:* Experiments on noise removal method were conducted on three dictionary size values: 2000, 3000, and 4000 words. Dictionary size is a measure of the number of words with the largest frequency used for training classifier. From the experimental results (Figure 2) can be seen that in Naïve Bayes Classifier, stop words removal method gives better accuracy results. However it is the opposite in SVM Classifier. For punctuation marks removal, experimental result (Figure 3) shows that for both classifiers, this method does not give better results than without it. While on alpha-numeric words removal, not much different from the previous experiment, this step also does not give better accuracy results (Figure 4).

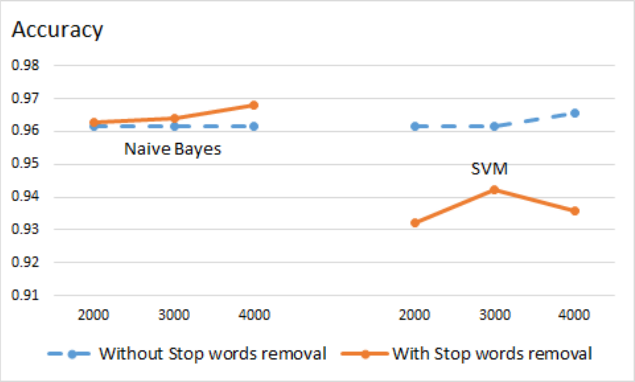

Fig. 2. Experiment Result on Stop Words Removal Pre-processing Method

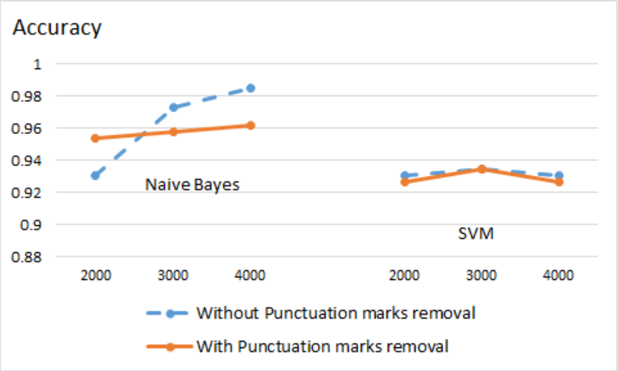

Fig. 3. Experiment Result on Punctuation Marks Removal Pre-processing Method

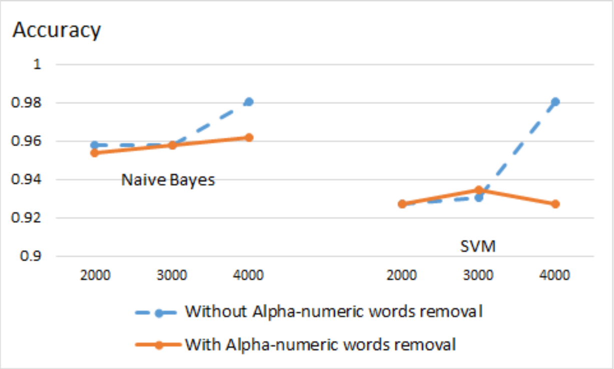

Fig. 4. Experiment Result on Alpha-numeric Words Removal Pre-processing Method

*2) Stemming and Lemmatization Experiments:* For stemming and lemmatization pre-processing methods, experiments were also conducted on three dictionary size values: 2000, 3000, and 4000 words. The first experiment (Figure 5) showed the fact that in the Naïve Bayes classifier, the stemming step gave the best results. While in SVM, lemmatization gives better results than stemming. The second experiment (Figure 6) is a comparison of the three types of stemmer: Snowball, Porter, and Lancaster. The results show that Lancaster gives better accuracy results for both classifier.

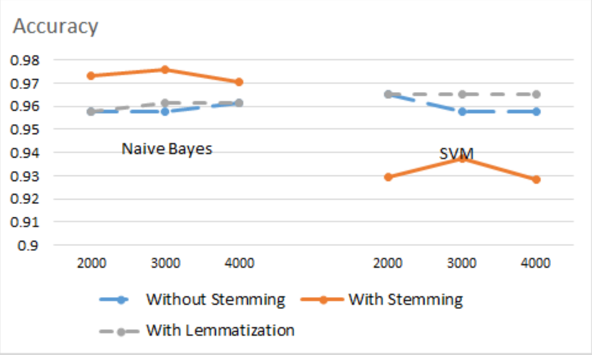

Fig. 5. Experiment Result on Stemming and Lemmatization Pre-processing Method

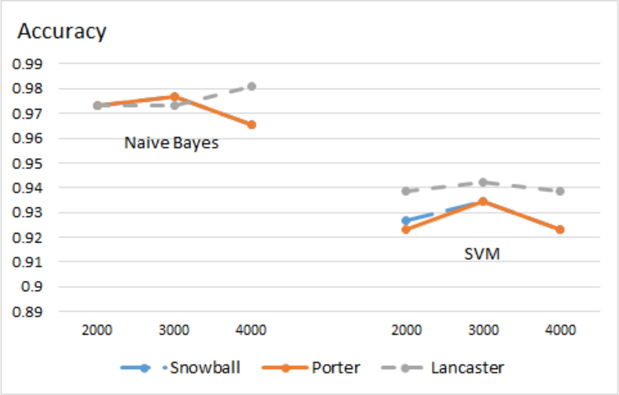

Fig. 6. Experiment Result on Three Kind of Stemmers

*3) With/without Pre-processing Experiments:* This experiment gives results (Figure 7) that pre-processing steps provide significant results in Naïve Bayes classifier. While in SVM, this step gives lower accuracy results.

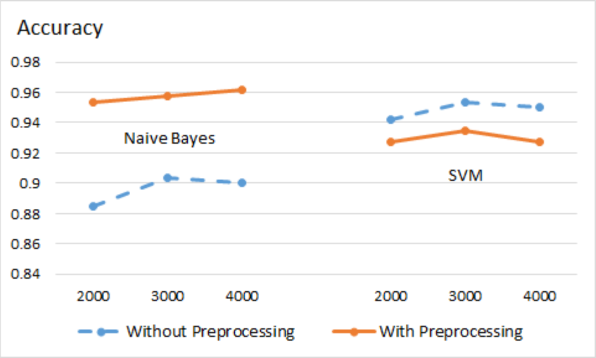

Fig. 7. Experiment Result on With/without Pre-processing Steps

*4)Term Frequency Experiments:* The experiments carried out gave result that determining the importance of the word using TF-IDF gives better accuracy results than the term frequency only (Figure 8).

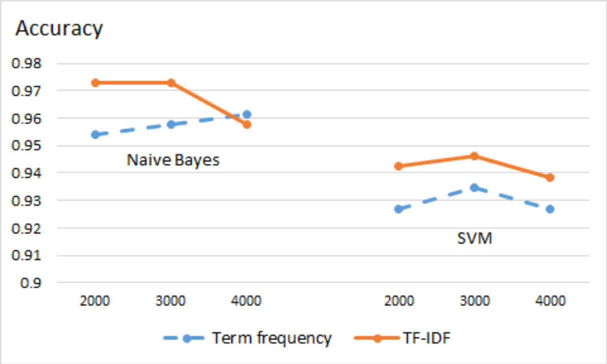

Fig. 8. Experiment Result on Term Frequency and TF-IDF

Based on the information given in the previous figures, the effect of some pre-processing methods on the performance of spam detection algorithms is described. The results show that each pre-processing method gives different results. In the Naive Bayes classifier, pre-processing methods that deliver better results are stop words removal and stemming. Meanwhile, in SVM classifier, better results are obtained if not using the pre-processing method. This is due to stop words removal and stemming make the probability calculation on Naïve Bayes more suitable for the classification of spam email task. While SVM as non-probabilistic classifier, building hyperplanes can be done without going through pre-processing stages.

From the overall experimental results it can also be seen that Naïve Bayes classifier gives better accuracy results for almost all parameters. In addition, in terms of dictionary size, the dictionary size of 3000 often gives better results than the two other size values. This is because the dictionary size which is too small (2000) or too large (4000) making the classification process with Naïve Bayes and SVM less accurate.

## V. Conclusion

Pre-processing methods plays an important role in all classification tasks, including spam email detection. In most studies of spam email detection, the right combination of pre-processing methods has not been considered seriously. Actually, if used correctly, the pre-processing method will provide a significant increase in classification results. This paper provides the effect of using pre-processing methods on spam email detection. We use combinations of 5 pre-processing methods (noise removal, stop words removal, stemming, lemmatization, and term frequency) on 2 spam classifiers (Naïve Bayes and Support Vector Machine) to evaluate their effect on the classification results.

This study recommends that the use (or no use) of the appropriate pre-processing methods on each classifier will result in better accuracy. This depends on the classifier used. For Naïve Bayes classifier, the combination of stop words removal and stemming gives better results than other combinations. However, for Support Vector Machine (SVM) classifier, the pre-processing stage often does not provide an increase in classification results. This difference is caused by the characteristics of these two classifiers. The Naïve Bayes classifier, which is a probabilistic classifier, is sensitive to word forms and presence of stop words. On the other hand, SVM as a non-probabilistic classifier, does not require almost all pre-processing methods. Further enhancement can be made in this study for more combination of pre-processing methods and more classifiers in the field of email spam detection. Bigger size of datasets are also needed for better evaluation in further study.

## Acknowledgment

We would like to thank the lecturers and colleagues of Doctoral Program in Electrical Engineering and Informatics of Bandung Institute of Technology who inspired writer in this research. We also thank the colleagues in Bachelor Program in Informatics of Widyatama University for the support.

## References

[1] G. V. Cormack, “Email Spam Filtering: A Systematic Review,” *Foundations and Trends® in Information Retrieval*, vol. 1, no. 4, pp. 335–455, 2008.

[2] E. Blanzieri and A. Bryl, “A survey of learning-based techniques of email spam filtering,” *Artificial Intelligence Review*, vol. 29, no. 1, pp. 63–92, 2008.

[3] W. Yerazunis, “Correspondence with Paul Graham.” 2002.

[4] B. Leiba, J. Ossher, V. Rajan, R. Segal, and M. Wegman, “SMTP Path Analysis,” in *Conference on Email and Anti-spam*, 2005, vol. 2, no. 1, pp. 54–66.

[5] S. Balakrishnan and K. L. Shunmuganathan, “An Agent Based Collaborative Spam Filtering Assistance Using JADE,” *International Journal of Applied Engineering Research*, vol. 10, no. 21, pp. 42476–42479, 2015.

[6] T. A. Almeida, J. Almeida, and A. Yamakami, “Spam filtering: How the dimensionality reduction affects the accuracy of Naive Bayes classifiers,” *Journal of Internet Services and Applications*, vol. 1, no. 3, pp. 183–200, 2011.

[7] W. Feng, J. Sun, L. Zhang, C. Cao, and Q. Yang, “A Support Vector Machine based Naive Bayes Algorithm for Spam Filtering,” in *2016 IEEE 35th International Performance Computing and Communications Conference (IPCCC)*, 2016, no. IEEE, p. 8.

[8] A. Sharma and A. Suryawanshi, “A Novel Method for Detecting Spam Email using KNN Classification with Spearman Correlation as Distance Measure,” *International Journal of Computer Applications*, vol. 136, no. 6, pp. 975–8887, 2016.

[9] O. Kufandirimbwa and R. Gotora, “Spam Detection Using Artificial Neural Networks (Perceptron Learning Rule),” *Online Journal of Physical and Environmental Science Research*, vol. 1, no. 2, pp. 22–29, 2012.

[10] A. S. Rao, P. S. Avadhani, and N. B. Chaudhuri, “A Content-Based Spam E-Mail Filtering Approach Using Multilayer Perceptron Neural Networks,” *International Journal of Engineering Trends and Technology (IJETT)*, vol. 41, no. 1, pp. 44–55, 2016.

[11] J. Bluszcz, D. Fitisova, A. Hamann, A. Trifonov, and P. Jahnichen, “Application of Support Vector Machine Algorithm in E-Mail Spam Filtering,” pp. 1–5, 2016.

[12] Z. Khan and U. Qamar, “Text Mining Approach to Detect Spam in Emails,” *Proceedings of The International Conference on Innovations in Intelligent Systems and Computing Technologies*, no. February, 2016.

[13] H. Wei-chih and T. Yu, “E-mail Spam Filtering Using Support Vector Machines with Selection of Kernel,” *Information and Control*, pp. 764–767, 2009.

[14] D. C. Trudgian and Z. R. Yang, “Spam Classification Using Nearest Neighbour Techniques,” in *Intelligent Data Engineering and Automated Learning – IDEAL 2004*, 2004, pp. 578–585.

[15] S. B. Rathod and T. M. Pattewar, “Content Based Spam Detection in Email using Bayesian Classfifier,” in *2015 International Conference on Communications and Signal Processing (ICCSP)*, 2015, pp. 1257–1261.

[16] G. Sakkis, I. O. N. Androutsopoulos, G. Paliouras, V. Karkaletsis, C. D. Spyropoulos, and P. Stamatopoulos, “A Memory-Based Approach to Anti-Spam Filtering,” pp. 49–73, 2003.

[17] S. K. Trivedi, “A study of machine learning classifiers for spam detection,” in *2016 4th International Symposium on Computational and Business Intelligence (ISCBI)*, 2016, pp. 176–180.

[18] A. R. On and D. Glaucoma, “A Review on Different Spam Detection Approaches,” vol. 11, no. 6, pp. 2–7, 2015.

[19] J. Daniel and J. Martin, “Naive Bayes and Sentiment Classification,” in *Speech and Language Processing Stanford University*, 2017.