Journal of Universal Computer Science, vol. 19, no. 16 (2013), 2404-2419
submitted: 15/7/13, accepted: 27/9/13, appeared: 1/10/13 © J.UCS

Graph-based KNN Algorithm for Spam SMS Detection

Tran Phuc Ho
(Konkuk University, Seoul, Republic of Korea
phuctran1107@gmail.com)

Ho-Seok Kang
(Konkuk University, Seoul, Republic of Korea
hsriver@gmail.com)

Sung-Ryul Kim
(Konkuk University, Seoul, Republic of Korea
kimsr@konkuk.ac.kr, corresponding author)

Abstract: In the modern life, SMS (Short Message Service) is one of the most necessary services on mobile devices. Because of its popularity, many companies use SMS as an effective marketing and advertising tool. Also, the popularity gives hackers chances to abuse SMS to cheat mobile users and steal personal information in their mobile phones, for example. In this paper, we propose a method to detect spam SMS on mobile devices and smart phones. Our approach is based on improving a graph-based algorithm and utilizing the KNN Algorithm - one of the simplest and most effective classification algorithms. The experimentation is carried out on SMS message collections and the results ensures the efficiency of the proposed method, with high accuracy and small processing time enough for detecting spam messages directly on mobile phones in real time.

Key Words: Spam SMS Detection, Graph-based KNN, Smartphone, Classification, Mobile security, Data mining
Category: E.1, E.2, H.3.0, I.2, L.7

1 Introduction

In the recent years, the number of mobile phones has been increasing rapidly. It is one of the most necessary devices for everyone. The newer generation of mobile phones (usually called smart-phones) performs more and more like small personal computers [Castiglione et al., 2009] and this allows the possibility of spreading malware on mobile phones also. SMS (Short Message Service) is one of the most widely used data services in the world because of its low cost and ease of usage. Furthermore, traditional SMS is not only used for texting among people but it is also used as an alternative authentication method, usually as a second-way token (mainly in mobile banking, one-time password delivery, etc).

Spam SMS is a kind of spamming directed at the text messaging. From the early 2000s, many users have begun to see an increase in the number of unexpected SMS being sent to their phone through text messaging. In Korea, the volume of mobile spam was already bigger than the volume of email spam at the end of 2003 [Gómez Hidalgo et al., 2006]. Most contents are usually advertisements which are sent by commercial companies or hacking messages by hackers who want to cheat and steal personal information.

Those messages really annoy mobile users because, when a spam message comes in, mobile phone will alert users by ringtones or vibrations. Even more dangerously, when users read the messages and click on a link in the content, it may happen that they will download virus unintentionally. Thus, it is obvious that the social and security risks are increasing because of spam SMS.

A variety of technical measures against spam SMS have been proposed. Most are applications of several machine learning algorithms (Bayesian filtering techniques [Cormack et al., 2007], SVM (Support Vector Machine) [Joe and Shim, 2010], etc.) and many of them achieve some level of accuracies.

In this paper, we propose an improved method to detect spam SMS and move them out of inbox. This method is a combination of graph-based text representation technique and KNN algorithm. From a big SMS collection we make separate groups of messages for performance reasons. Each group is represented by a graph and is also as an entity in KNN (K Nearest Neighbor) algorithm (Figure 7). In this graph-based model, a message or a group of messages is represented by a graph, which is capable of capturing word order, word frequency and word co-occurrence within that document.

There are many effective machine learning algorithms that are used in information retrieval and internet security such as: SVM, NB (Naive Bayes), decision tree, etc. In spite of its simplicity, KNN (K Nearest Neighbor) is one of the most effective algorithms. It is a type of instance-based learning or lazy learning algorithms. Additionally, it performs classifying based on the closet training samples in feature space. We use KNN algorithm to classify two types of SMS that are represented in graph-based model.

We evaluate our proposed method on two independent SMS datasets. The first one is the SMS collection of NUS SMS Corpus. It has been collected by Tiago A. Almeida [Almeida] and José María Gómez Hidalgo [Gómez Hidalgo] The second one is chosen from [Uysal and Yildiz]. It is collected independently of NUS SMS Corpus and used as testing messages. The experimental result shows that our method achieves high detection accuracy.

The rest of paper is organized by the following sections: Section 2 is the related work and we discuss the overview of some approaches in the literature. Section 3 describes pre-processing and feature selection methods that we use in our system. Next, section 4 describes in more detail the implementation based on our proposed method including parts such as system architecture, graph-based text representation model, traditional KNN algorithm, and classification. In the Section 5, we will give our experimental result and in the last section we conclude and give a possible future work.

2 Related Work

The mobile networks are indeed more and more popular in our lives but their increasing pervasiveness and widespread coverage raises serious security concerns [Palmieri et al., 2011]. In other words, smartphones may now represent an ideal target for malware writers. As the number of vulnerabilities and, hence, of attacks increase, there has been a corresponding rise of security solutions proposed by researchers [La Polla et al., 2013, Fung, 2011, Canard et al., 2011]. Currently, there are several papers related to spam SMS detection. Some works (such as [Healy et al., 2004]) have proposed methods based on machine learning algorithms (like SVM, KNN, or Naive Bayes) for classification. José María Gómez [Gómez Hidalgo et al., 2006] tries to detect spam SMS by applying a version of Bayesian filter algorithm. First, they reviewed a number of technical measures against spam email, focusing on Bayesian filtering. Then they presented how the mobile network operator had seen the problem of mobile spam in Europe, and which role Bayesian filtering could play in reducing or stopping it. After that, they describe a series of experiments designed to test to what extent it was effective addressing the mobile spam problem.

Recently, the authors of [Rafique and Abulaish, 2012] also apply graph-based model in spam SMS detection in their paper. Firstly, they tokenized the content of all sample messages using white spaces and punctuations and built the graph from those tokens. Then, they calculated probability of each token appearing in messages and used KL-Divergence for classification. However, mobile devices which have small memory and it will take a lot of time for processing a big graph extracted from sample messages.

Cross-site scripting (XSS) is a type of computer security vulnerability typically found in web applications. It enables hackers to inject client-side script into web pages viewed by other users. Jun-Ho Choi [Choi et al., 2012] proposed a method that is a combination of n-Gram and SVM to detect the malicious script code. To be more specific, the index term applying n-gram and code dictionary were generated to increase the accuracy of binary pattern classification. Furthermore, SVM which provides efficiency and accuracy in the process of a binary classifiation, is used to determine whether the malicious script code could be detected or not.

A new method was proposed for filtering spam SMS using CAPTCHA (Completely Automatic Public Turing Test to Tell Computer and Human Apart) systems [Shirali Shahreza and Shirali Shahreza, 2008]. These systems are used to distinguish between human users and computer programs automatically. One work that can be noted is a software framework named SEESMS (Secure Extensible and Efficient SMS) [De Santis et al., 2010] that allows two peers to exchange encrypted and digitally signed SMS messages. The cryptographic part of SEESMS is implemented on top of the Java binary code library, a widely used open-source library. Castiglione [Castiglione et al., 2012] analyzed some unexpected phenomena by profiling the code of SEESMS and exposed the issues causing its bad performance.

3 Pre-Processing and Feature Selection

In the classification, feature selection is necessary in order to remove the noisy features and select the good ones. Furthermore, it also simplifies the calculation avoiding overfitting and increases the accuracy in the next steps. To address this issue, some approaches are used to extract the terms by the noun words included inside the glossary of WordNet [Hwang et al., 2011] or Wikipeadia [Huang et al., 2012]. In this paper, we present two commonly used text feature selection methods: mutual information (MI) and X2-Statistic (CHI) [Wang and Liu, 2010]. Figure 1 shows an example feature selection process. The input is a mixed list of words. Besides the representative words in the document, it also includes: one-character words, common words, stop words, etc. After processing by feature selection method, the size of words list is reduced and several redundant words are removed.

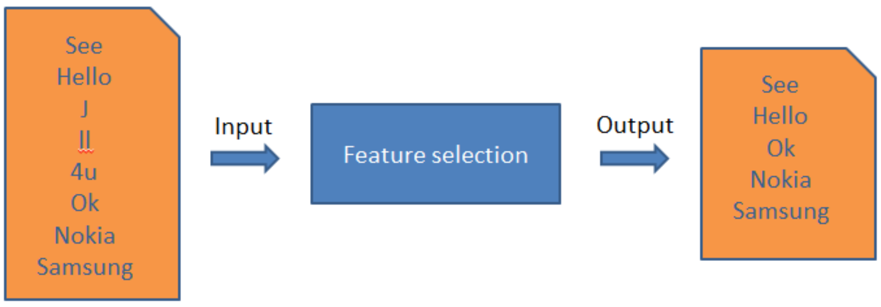


Figure 1: Feature selection example

3.1 Mutual Information (MI)

Mutual information is a concept in probability and information theory; it is used to calculate the mutual dependence of two random variables. In the feature selection field, it is usually used to compute the dependence between a feature t and a class c. In this paper, t denotes a token (word or phrase) extracted from messages and class c is type of messages: spam or ham. We apply the following formulas (the two of them are equivalent) to calculate this measurement (MI):

$$I(t,c) = \log \frac{P_r(t,c)}{P_r(t|c) \times P_r(t)} \tag{1}$$

$$I(t,c) = \log P_r(t|c) - \log P_r(t), \tag{2}$$

where $I(t,c)$ is the dependent weight between feature t and class c. $P_r(t,c)$ is the probability that feature t and class c appear simultaneously, $P_r(t|c)$ is the conditional probability of feature t in messages of class c, and $P_r(t)$ is probability of feature t.

3.2 χ2-statistic (CHI)

The χ2-statistic measures the lack of independence between token t and class c. If t and c are independent, CHI has a lowest value (zero). It is defined by the following formula:

$$\chi 2(t,c) = \frac{N \times (P(t,c) \times P(\bar{t},\bar{c}) - P(t,\bar{c}) \times P(\bar{t},c))}{P(w) \times P(\bar{w}) \times P(c) \times P(\bar{c})} \tag{3}$$

where N is the total number of messages in the sample set. $P(t,c)$ is the probability that token t and class c appear simultaneously, $P(t)$ is the probability of token t, and $P_r(c)$ is the probability that the text belong to category c. The barred symbols in $P(\bar{t},\bar{c})$, $P(t,\bar{c})$, $P(\bar{t},c)$, $P(\bar{c})$, and $P(\bar{t})$ signifies the complimentary events to symbols without bars.

3.3 Combined Feature Selection Method

Although, CHI method is used commonly with many advantages, there still exists some issues [Wang and Liu, 2010]. For example, it increases the weight of words which appear in appointed class with low frequency, but high in other classes and it reduces the weights of low frequency words. According to formula (1), for such words, when $P(t,c_i) \to 0$, $P(t)$ and $P(c_i)$ are not tending to zero, then $P(t,c_i)/(P(t) \times P(c_i)) \to 0$, so $I(t,c)$ will tend to negative infinitude, and these words will be thrown out.

According to the formula (2), in [Wang and Liu, 2010] they proposed a method for the words which have the same $\log P_r(t|c)$, the weights of low frequency words will be higher. So the problems above are resolved. Because of this reason, we decide to use the combined feature selection method for resolving those issues and select good features that will be used in classification.

In our method, we calculate the weight of each feature based on the indicator function:

$$E(t,c) = \alpha E_1(t,c) + \beta E_2(t,c), \qquad 0 < \alpha, \beta < 1, \alpha + \beta = 1 \tag{4}$$

where $E_1(t, c)$ is the weight calculated by $\chi 2$-statistic and $E_2(t, c)$ is calculated by the mutual information method.

After tokenizing the content of messages into space-delimited words, we remove several words that have only one character and calculate weight of each word by using the feature selection method above. Finally, we select the words which have high weight and use them for constructing the graphs in the next section.

4 Proposed Method

4.1 System Architecture

Firstly, we separate the message collection into many small message groups (the message collection has been already labeled). Each group contains X messages that all have the same label. One problem is how to choose the value of X. If we increase X, the graph will be bigger and the complexity of algorithm will increase. Otherwise, if we decrease X, the algorithm will run faster but the result quality may suffer. In our experiment, we chose $X = 5$.

Secondly, we tokenize the content of messages in a group by white spaces and punctuations. Through the feature selection method we select the good tokens for constructing the sample graph. Figure 2 illustrates this process.

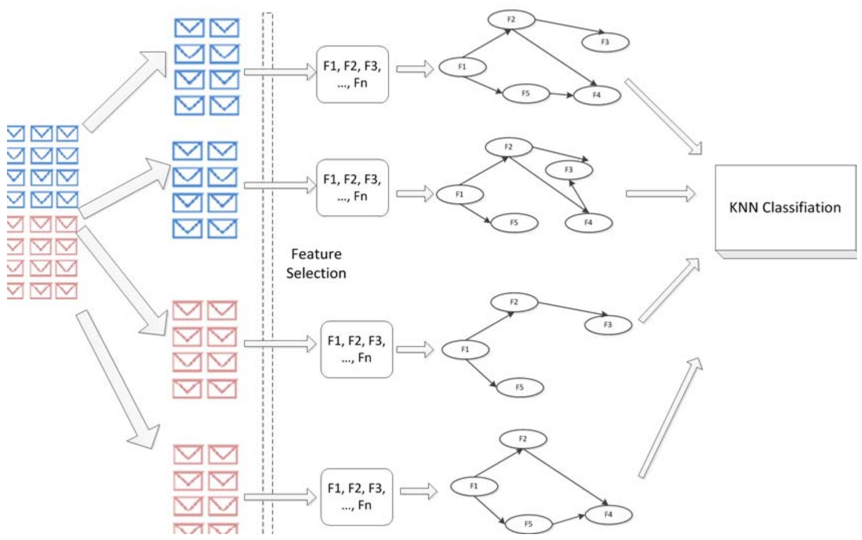


Figure 2: Graph-based system architecture

When a new message comes in (Figure 3), we also use the same algorithm for truncating and selecting tokens and then building the graph, resulting in the testing graph. In the real-time detection, a message group has only one message. Finally, we use all of sample graphs and testing graph in KNN algorithm to decide the label of the new message.

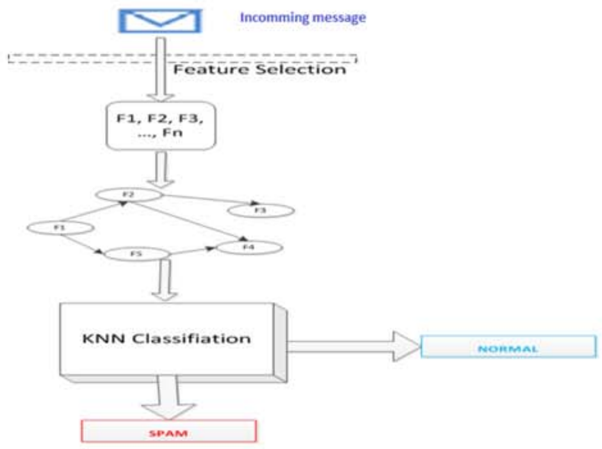


Figure 3: Classification Graph-based system architecture.

4.2 Graph-Based Text Representation Model

We now describe the structure of the graph that we use in our algorithm. Each graph (illustrated in Figure 4) is a triple: $G = (V, E, FWN)$, where V is the set of nodes, E is the collection of weighted edges linking the nodes. FWM (Feature Weight Matrix) represents the weight of edges and nodes.

- Node:
  Each node in the graph denotes a token that was selected by the feature selection method. This token is a unique word within a graph.
- Edge:
  The edge is constructed based on the order and co-occurrence relationship between two feature words. If feature words co-occur within a step length, we assign an edge to connect two nodes that contain those feature words.
- Feature Weight Matrix:
  Feature weight matrix in Figure 5 is used to represent the weight of edges and the probability of tokens represented by nodes. For more detail, the edge's weight w_{ij} can be calculated as co-occurrence frequency of two feature words f_i and f_j within a step length. Because the feature words have fixed order

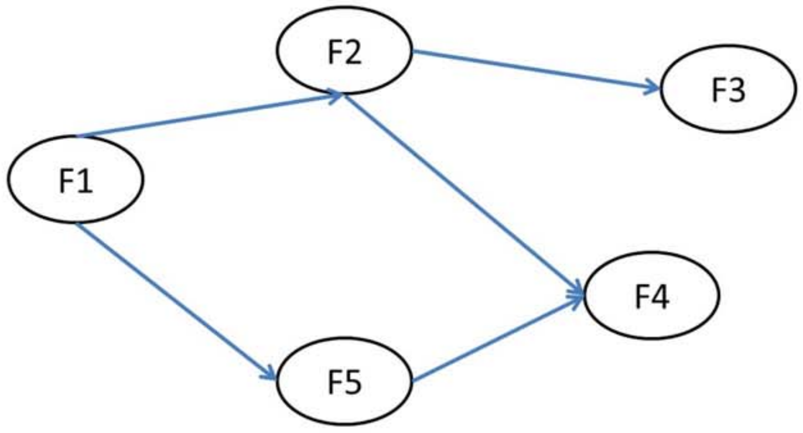


Figure 4: Graph structure

in a message (for example, we have ‘scientific paper’, but we mostly never have the reverse situation ‘paper scientific’), we just need to calculate the weight w_{ij} when $i > j$. Otherwise, w_{ij} will be zero. It reduces the complexity of our algorithm and helps to improve the performance when we construct a graph. In addition, feature weight matrix also represents the appearance frequency of feature words. This value is described in w_{ii}. By these, the matrix preserves information in text such as frequency of single words, order and co-occurrence between two words.

$$\begin{array}{c} \\ f1 \\ \vdots \\ fn \end{array} \begin{array}{c} \begin{array}{ccc} f1 & \cdots & fn \end{array} \\ \begin{pmatrix} w_{11} & \cdots & w_{1n} \\ \vdots & \ddots & \vdots \\ 0 & \cdots & w_{nn} \end{pmatrix} \end{array}$$

Figure 5: Feature weight matrix

4.3 Traditional KNN Algorithm

In the pattern recognition area, the KNN algorithm is a method for classifying objects based on the closest training example in the feature space. It is a type of instance-based learning or lazy learning where the function is only approximated locally and all computation is deferred until classification.

In the text classification area, the idea of KNN is very simple: in K (constant) nearest neighbors of the text T to be classified, the class of T is the most frequently appearing class in this collection (see Figure 6).

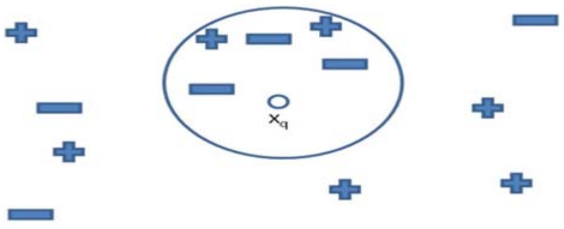


Figure 6: KNN algorithm

KNN can be represented by the following formula [Zhang et al., 2009]:

$$y(d, c_j) = \sum_{d_i \in KNN} Sim(d, d_i) \times y(d_i, c_j) - b_j \tag{5}$$

and

$$y(d, c) = \begin{cases} 0 & d \in c \\ 1 & d \notin c \end{cases}$$

where d is the document to be classified, d_i is the i-th sample document, c_j is the j-th category, $y(d, c)$ indicates whether document d belongs to category c_j ($y(d, c_j)$ is 1 if d belong to c_j and 0, otherwise), and b_j is a preset threshold of c_j. $Sim(d, d_j)$ is the similarity between the document d and d_j. A commonly used distance metric for continuous variables is the Euclidean distance to measure the similarity among them. For text classification, another metric can be used such as the overlap metric (or Hamming distance). Often, the classification accuracy of KNN can be improved significantly if the distance metric is learned with specialized algorithms for instance, large margin nearest neighbor or neighborhood components analysis.

4.4 Classification

In Figure 7, after building sample graphs from selected tokens, we use them as sample elements in the KNN algorithm. When a new message comes in, we select good tokens and build a testing graph as described before.

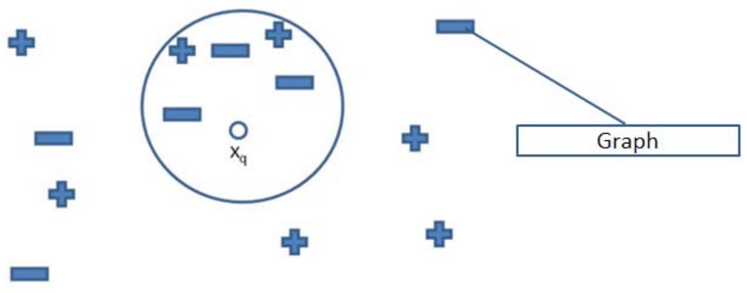


Figure 7: KNN algorithm

To apply KNN with graph-based representation, we need to define a new algorithm for measuring the similarity of two graphs by using their nodes, edges, and the weights of edges.

– FW (Feature Weight) [Wang and Liu, 2010]: it describes the similarity between two graphs. Mainly the calculation is to add weights of the edges of which both endpoint nodes and the edge itself appear simultaneously in the two graphs.

We have improved this algorithm from [Wang and Liu, 2010]. It is more strict and helps to improve the accuracy. It is stated as follows:

```
Input:
  Testing graph $tg$, sample graph $sg$
Output
  FW
Procedure:
1. For each edge in $tg$
2.     If edge in $sg$
3.         nextEdge = getNextEdge(edge,tg) //get next edge of
           // edge in testing graph
4.         If nextEdge in sg
5.             If $w_{ij}$(tg) $\geq$ $w_{ij}$(sg)
6.                 If j >i
7.                     FW+=$\alpha \times w_{ij}$(sg)
8.                 Else if i=j
9.                     FW+= $w_{ij}$(sg)
10.                End if
11.            Else If $w_{ij}$(tg) <$w_{ij}$(sg)
12.                If j >i
```

```
13.             FW+= $\alpha \times w_{ij}$(tg)
14.          Else if i=j
15.             FW+= $w_{ij}$(tg)
16.          End if
17.       End if
18.     End if
19.  End if
20. End for
```

Algorithm 1. Calculate the similarity weight between two graphs

In the algorithm, α is the value that we use to enhance the effect of edge weight in classification. The reason is because we believe that the weight of edges are more important than that of the nodes. The experimental results verifies this.

We define a factor called Nfp (Not fit percent) to be used in the classification. It indicates how many nodes in the sample graph with their weights larger than zero also appear in the test graph.

$$Nfp = \frac{|\{node|node \in tg \land node \in sg\}|}{|Number\ of\ Feature\ terms|} \tag{6}$$

We calculate this value of testing graph and sample graph before calculating the similarity weight of two graphs. If Nfp is greater than threshold that we defined, we will calculate FW value of two graphs. Otherwise, it means two graphs does not belong to same category and we do not need to calculate their similarity weight. This process helps to improve the performance and the complexity of similarity algorithm can be reduced greatly.

The detail of the graph-based KNN algorithm used to detect spam SMS is described as follows:

```
Input:
  Testing graph $g_i$
  Sample graph collection SG={$sg_1$,$sg_2$,...,$sg_i$,...,$sg_n$}
  Value K
Output
  Class of $message_i$
Procedure:
 1.   Initial List $RL$ to store FW and message category
 2.   For each $sg$ in Sample Graph
 3.     If Nfp($tg$,$sg$) > $THRESHOLD$
 4.       Calculate FW($tg$,$sg$)
 5.       If $RL$ is not full
 6.         Add FW($tg$,$sg$) and Cateogory of $sg$ into RL
```

```
7.          Else If $RL$ is full
8.            If FW($tg$,$sg$) >min (FW in $RL$)
9.              Replace minFW in $RL$ by FW($tg$,$sg$)
10.           End if
11.         End if
12.       End if
13.     End for
14.     Result is category that is most appearance in RL
```

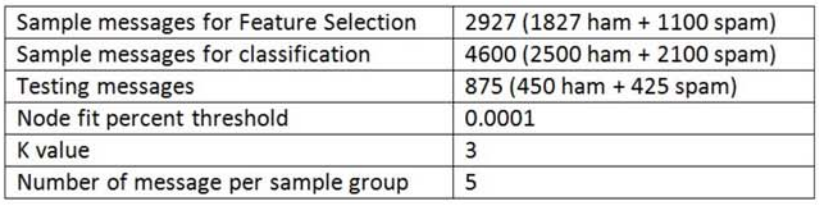
Algorithm 2. Graph-based KNN algorithm

5 Experimental Results

In our experiment, we choose two independent spam SMS datasets. The first one is the dataset of NUS SMS Corpus. It has a total of 5574 messages and includes 4,827 SMS legitimate messages (86.6%) and 747 (13.4%) spam messages. This collection is used to build sample graphs from selected tokens after going through the feature selection process.

The second one is the collection of SMS downloaded from [Uysal and Yildiz]. It is totally different from NUS SMS Corpus and includes 875 SMS messages (450 ham messages and 425 spam messages). In our experiment, these messages simulate the new ones that comes in the real time.

In this experiment, we choose input data and constant values used in the proposed method as in Table 1:



Sample messages for Feature Selection	2927 (1827 ham + 1100 spam)
Sample messages for classification	4600 (2500 ham + 2100 spam)
Testing messages	875 (450 ham + 425 spam)
Node fit percent threshold	0.0001
K value	3
Number of message per sample group	5

Table 1: Experimental input

In building sample graph phase, we have compared the processing time of our work with the previous work. The comparison in Figure 8 shows that our work performs much better in terms of computation time. We did experiment in many cases with different quantity of sample messages and as the number of messages increases, the performance difference also increases.

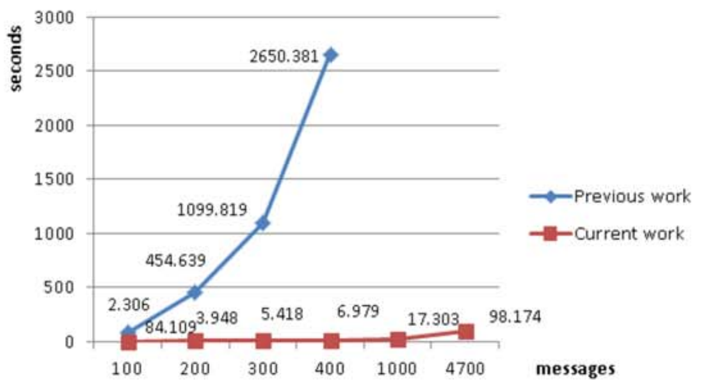
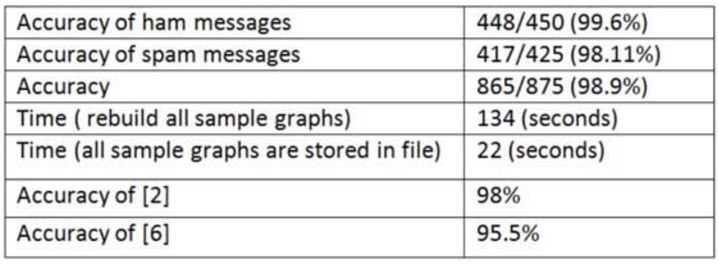


Figure 8: Processing time comparison

One more difference we have from the previous work is that we use difference step length values for spam message(S_s) and ham messages(S_h) (used to consider the co-occurrence relationship between two tokens of nodes for building edges in graph). Specifically, when $S_s = 3$ and $S_h = 5$, we obtain the best result that is shown in Table 2. One can see that the quality has been improved from the previous works. Note that we have improved the quality of the classification result while simultaneously having a huge improvement in performance. Also note that it takes just 22 seconds to process 875 testing messages, meaning about 0.025 seconds per message. Even though this experiment was done on a PC, we fully expect that the processing time on mobile devices will be as negligible as that on a PC, considering the processing power of current mobile devices.



Accuracy of ham messages	448/450 (99.6%)
Accuracy of spam messages	417/425 (98.11%)
Accuracy	865/875 (98.9%)
Time (rebuild all sample graphs)	134 (seconds)
Time (all sample graphs are stored in file)	22 (seconds)
Accuracy of [2]	98%
Accuracy of [6]	95.5%

Table 2: Detection quality comparison with previous works

The chart in Figure 9 shows the detail of our experiment in many cases with different step length value (S_s: step length for spam message and S_h: step length for ham message).

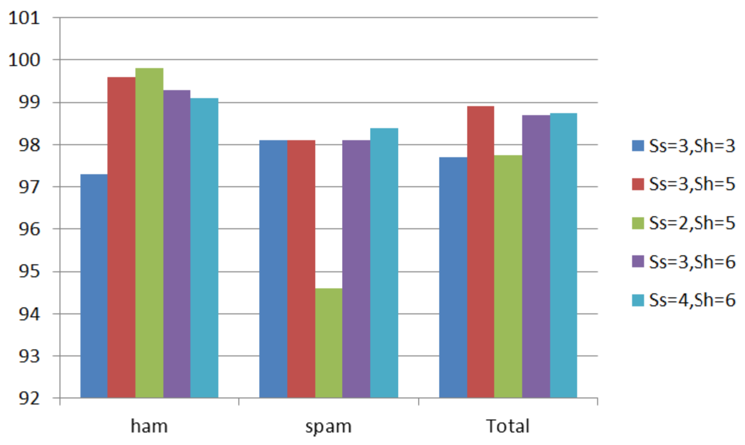


Figure 9: Experimental result

Figure 10 is the performance comparison with different step length values. When step length values are increased, the processing is more complex and the performance is lower.

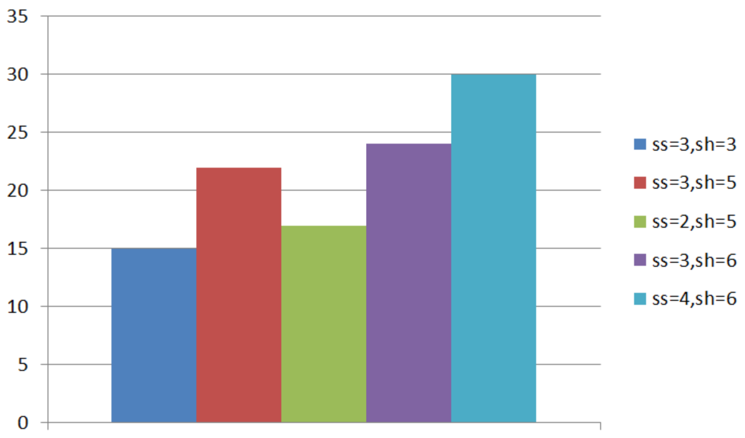


Figure 10: Experimental performance

6 Conclusion and Future Work

In this paper, we have proposed the Graph-based KNN algorithm for spam SMS detection. According to this method, detection accuracy and performance are improved from a previous version. The performance and the accuracy of our method are verified by experimenting on real SMS message collections. We also compared our method with other detection methods and the result proves the efficiency of our proposed method. In the future, we plan to apply this method to classifying multi-language SMS messages or detecting spam in chatting applications.

Acknowledgements

This research was supported by Next-Generation Information Computing Development Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Science, ICT & Future Planning (2011-0029924) and MSIP (Ministry of Science, ICT & Future Planning), Korea, under the ITRC (Information Technology Research Center) support program (NIPA-2013-H0301-13-3006) supervised by the NIPA (National IT Industry Promotion Agency).

References

[Almeida] Almeida, T.A.: `http://www.dt.fee.unicamp.br/~tiago`

[Canard et al., 2011] Canard, S., Devigne, J., Laguillaumie, F.: "Improving the Security of an Efficient Unidirectional Proxy Re-Encryption Scheme"; Journal of Internet Services and Information Security (JISIS) vol 1, pp. 140-160, 2011.

[Castiglione et al., 2009] Castiglione, A., De Prisco, R., De Santis, A.: "Do You Trust Your Phone?"; In E-Commerce and Web Technologies, pp. 50-61, 2009.

[Castiglione et al., 2012] Castiglione, A., Cattaneo, G., Cembalo, M., De Santis, A., Faruolo, P., Petagna, F., Ferraro Petrillo, U.: "Engineering a secure mobile messaging framework"; Computers & Security 31, no. 6: pp. 771-781, 2012.

[Choi et al., 2012] Choi, J.H., Choi, C., Ko, B.K., Kim, P.K.: "Detection of cross site scripting attack in wireless networks using n-Gram and SVM"; Mobile Information Systems 8, no. 3 pp. 275-286, 2012.

[Cormack et al., 2007] Cormack, G.V., Gómez Hidalgo, J.M., Sánz, E.P.: "Spam filtering for short messages"; In Proceedings of the sixteenth ACM conference on Conference on information and knowledge management, pp. 313-320, 2007.

[De Santis et al., 2010] De Santis, A., Castiglione, A., Cattaneo, G., Cembalo, M., Petagna, F., Petrillo, U.F.: "An extensible framework for efficient secure SMS"; International Conference on Complex, Intelligent and Software Intensive Systems (CISIS), pp. 843-850, 2010.

[Fung, 2011] Fung, C.: "Collaborative intrusion detection networks and insider attacks"; Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications 2.1. pp. 63-74, 2011.

[Gómez Hidalgo] Gómez Hidalgo, J.M.: `http://www.esp.uem.es/jmgomez`

[Gómez Hidalgo et al., 2006] Gómez Hidalgo, J.M., Bringas, G.C., Sánz, E.P., García, F.C.: "Content based SMS spam filtering"; Proceedings of the 2006 ACM symposium on Document engineering, pp. 107-114, 2006.

[Healy et al., 2004] Healy, M., Delany, S.J., Zamolotskikh, A.: "An assessment of case base reasoning for short text message classification"; Proceedings of the 15th. Irish Conference on Artificial Intelligence and Cognitive Sciences (AICS'04), pp.9-18, 2004.

[Huang et al., 2012] Huang, L., Milne, D., Frank, E., Witten, I.H.: “Learning a concept-based document similarity measure”; Journal of the American Society for Information Science and Technology 63, no. 8, pp. 1593-1608, 2012.

[Hwang et al., 2011] Hwang, M.G., Choi, C., Kim, P.K.: “Automatic enrichment of semantic relation network and its application to word sense disambiguation”; IEEE Transactions on Knowledge and Data Engineering, vol. 23, no. 6, pp 845-858, 2011.

[Joe and Shim, 2010] Joe, I., Shim, H.: “An SMS spam filtering system using support vector machine”; Future Generation Information Technology, pp. 577-584, 2010.

[La Polla et al., 2013] La Polla, M., Martinelli, F., Sgandurra, D.: “A survey on security for mobile devices”; IEEE Communications Surveys & Tutorials, vol. 15, no. 1. pp. 446-471, 2013.

[Palmieri et al., 2011] Palmieri, F., Fiore, U., Castiglione, A.: “Automatic security assessment for next generation wireless mobile networks”; Mobile Information Systems 7, no. 3, pp. 217-239, 2011.

[Rafique and Abulaish, 2012] Rafique, M.Z., Abulaish, M.: “Graph-based learning model for detection of SMS spam on smart phones”; 8th International Wireless Communications and Mobile Computing Conference (IWCMC), pp. 1046-1051, 2012.

[Schenker et al., 2003] Schenker, A., Last, M., Bunke, H., Kandel, A.: “Classification of web documents using a graph model”; Proceedings of the Seventh International Conference on Document Analysis and Recognition (ICDAR), pp. 240-244, 2003.

[Shirali Shahreza and Shirali Shahreza, 2008] Shirali Shahreza, M.H., Shirali Shahreza, M.: “An anti-SMS-spam using CAPTCHA”; ISECS International Colloquium on Computing, Communication, Control, and Management (CCCM’08), vol. 2, pp. 318-321, 2008.

[Uysal and Yildiz] Uysal, A., Yildiz, C.: `https://code.google.com/p/smsspamdetector/`

[Wang and Liu, 2010] Wang, Z., Liu, Z.: “Graph-based KNN text classification”; Seventh International Conference on Fuzzy Systems and Knowledge Discovery (FSKD), pp. 2363-2366, 2010.

[Zhang et al., 2009] Zhang, X., Huang, H., Zhang, K.: “KNN Text Categorization Algorithm Based on Semantic Centre”; International Conference on Information Technology and Computer Science, pp. 249-252, 2009.