Expert Systems with Applications 42 (2015) 1065–1073

Contents lists available at ScienceDirect

Expert Systems with Applications

journal homepage: www.elsevier.com/locate/eswa

Semi-supervised learning using frequent itemset and ensemble learning for SMS classification

Ishtiaq Ahmed [a], Rahman Ali [a], Donghai Guan [b], Young-Koo Lee [a], Sungyoung Lee [a], TaeChoong Chung [a,*]

[a] Department of Computer Engineering, Kyung Hee University, South Korea
[b] College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, China

ARTICLE INFO

Article history:
Available online 16 September 2014

Keywords:
Short Message Service (SMS)
Ham
Spam
Frequent itemset
Ensemble learning
Semi-supervised classification

ABSTRACT

Short Message Service (SMS) has become one of the most important media of communications due to the rapid increase of mobile users and it's easy to use operating mechanism. This flood of SMS goes with the problem of spam SMS that are generated by spurious users. The detection of spam SMS has gotten more attention of researchers in recent times and is treated with a number of different machine learning approaches. Supervised machine learning approaches, used so far, demands a large amount of labeled data which is not always available in real applications. The traditional semi-supervised methods can alleviate this problem but may not produce good results if they are provided with only positive and unlabeled data. In this paper, we have proposed a novel semi-supervised learning method which makes use of frequent itemset and ensemble learning (*FIEL*) to overcome this limitation. In this approach, Apriori algorithm has been used for finding the frequent itemset while Multinomial Naive Bayes, Random Forest and LibSVM are used as base learners for ensemble learning which uses majority voting scheme. Our proposed approach works well with small number of positive data and different amounts of unlabeled dataset with higher accuracy. Extensive experiments have been conducted over UCI SMS spam collection data set, SMS spam collection Corpus v.0.1 Small and Big which show significant improvements in accuracy with very small amount of positive data. We have compared our proposed *FIEL* approach with the existing SPY-EM and PEBL approaches and the results show that our approach is more stable than the compared approaches with minimum support.

1. Introduction

Since the last couple of decades, mobile phone users have been rapidly increased as the SMS are easy, less expensive and independent upon cell phone operating systems. Consequently, SMS has become one of the popular communication medium throughout the world. According to the survey (Gsma, 2011a), 3.5 billion or 80% active users, throughout the word, use mobile SMS as a communication medium. Out of this huge number of SMS, a large number of SMS are spam, generated by offenders for a number of reasons (Delany, Buckley, & Greene, 2012). Firstly, sending SMS is less expensive, as most of the mobile operators have offered prepaid SMS package at very low cost. Secondly, as the mobile subscribers are more interactive than computer users, SMS is being considered as a trusted service and the people are getting more comfortable with sharing confidential information. According to the survey (He, Sun, & Zheng, 2008), people are receiving more spam SMS than ham SMS. Similarly, some interesting US statistics portray that cell phone users within US receive 1.1 billion spam SMS and 44% mobile device users receive spam through the whole year. In the same way, Chinese users experience 8.29 spam SMS per week (Qian, Xue, & Xiaoyu, 2009). Besides this, according to Gsma (2011b), SMS spamming problems have been treated as a major problem in Middle East and South East Asia for affecting 20–30% of the overall traffic. According to Korea Information Security Agency (KISA), people are now getting more spams messages than ham messages (Lee & Choi, 2012). Due to the spam SMS, criminal gangs become stronger and perform different criminal activities (Delany et al., 2012). Subscriber also ends up with calling premium rate numbers or signing up to expensive subscription services. Moreover, network operators are also affected financially due to the higher network and operating costs caused by spam SMS.

The statistics of Fig. 1 shows that a large number of SMS are generated throughout the world whose manual filtration and classification, as spam or ham, is impossible task. Therefore, text

* Corresponding author.
E-mail addresses: ishtiaq.khu@khu.ac.kr (I. Ahmed), rahmanali@oslab.khu.ac.kr (R. Ali), dhguan@gmail.com (D. Guan), yklee@khu.ac.kr (Y.-K. Lee), sylee@oslab.khu.ac.kr (S. Lee), tcchung@khu.ac.kr (T. Chung).

http://dx.doi.org/10.1016/j.eswa.2014.08.054
0957-4174/

classification or categorization is used to automatically process and classify SMS data into ham or spam. Among different techniques, machine learning and data mining techniques have been extensively used for the automatic detection and classification of SMS. The machine learning techniques need sufficient amount of labeled training examples for training the classification models. Though designing a very effective text classification technique is very hard and arduous work, some commonly used supervised techniques, such as Naive Bayes probabilistic classifier (Lewis, 1998), K-nearest neighbor (KNN) (Cover & Hart, 1967), Support Vector Machine (SVM) (Joachims, IT, & Augustin, 2001), Decision trees C4.5 (Letouzey, Denis, & Gilleron, 2000), Neural Network (Anthony & Bartlett, 1999) have been applied to the same problem. A supervised strategy (Ahmed, Guan, & Chung, 2014), uses Naive Bayes with Apriori for SMS classification problem. A huge number of positive and negative instances are used to train the learner without having unlabeled instances. Though applying certain minimum support, Apriori is required for finding the frequent itemset. The training and testing strategy used is different from the traditional Naive Bayes approach (McCallum & Nigam, 1998). Nuruzzaman, Lee, and Choi (2011) have proposed independent and personal SMS filtering system having good accuracy, minimum storage consumption and acceptable processing time with the help of Naive Bayes algorithm. Najadat, Abdulla, Abooraig, and Nawasrah (2014) have used a new classifier by mixing different classifiers with no altering of their original algorithm to have better performance. In order to have better accuracy, Sohn, Lee, Han, and Rim (2012) have proposed a new approach by combining the lexical features and new stylish features. Lee, Yeom, Choi, and Kang (2011) proposed distributed spam filter model to use less resources of mobile phones with the help of Naive Bayes and Support Vector Machine. Besides these, feature-based classification technique, such as Xu, Xiang, Yang, Du, and Zhong (2012) has also been used for SMS classification. The main problem of supervised learning algorithms is to train them with, spam and ham, labeled data as a training dataset (Shahshahani & Landgrebe, 1994). In real life application, labeled data could be unavailable at the very beginning of the system. However, McCallum and Nigam (1998) showed that having small volume of both positive and negative training sets, a huge volume of unlabeled dataset can be very fruitful to classify SMS.

To overcome the problems of supervised approaches, semi-supervised learning approaches are used which exploit both labeled and unlabeled data for learning the models. Compared to supervised approaches, semi-supervised approaches require less number of labeled data. However, if only positive labeled data are available, it cannot be directly used. In the area of semi-supervised SMS classification, firstly, Valiant (1984) studied the problem of labeled positive dataset and unlabeled dataset for training machine learning algorithm that was later worked by Denis using Probably Approximately Correct (PAC) learning technique (Denis, 1998). They studied the computational complexity of learning from positive and unlabeled dataset under the statistical query model (Kearns, 1998). Later on, modified C4.5 decision tree, that uses statistical query model, was used by Letouzey et al. (2000). Muggleton (1997) has done a theoretical study on the issue of positive dataset.

A number of well-known semi-supervised methods, such as Spy-EM (Liu, Lee, Yu, & Li, 2002), PEBL (Yu, Han, & Chang, 2004), Rough Set and Ensemble based learning (Shi, Ma, Xi, Duan, & Zhao, 2011) can also be found in literature over the same problem. The methodology of Liu et al. (2002) consists of two parts. Firstly, identifying reliable negative dataset from unlabeled dataset and secondly constructing classifier with positive, negative and unlabeled dataset based on Expectation Maximization (EM) iteratively. This system is based on naive Bayes classification (NB) and EM algorithm (Dempster, Laird, & Rubin, 1977). As naive Bayes classifier requires a large dataset to train the system (Ng & Jordan, 2001), it is not considered as appropriate for text classification. However, for considering general model assumption, the performance of the prior probabilities model is questionable. The PEBL model (Yu et al., 2004) has introduced Mapping-Convergence (M-C) approach which achieves accuracy as high as traditional SVM with positive and unlabeled dataset. The core idea of this approach is almost similar to Spy-EM. Firstly, it identifies the reliable negative documents from the unlabeled set and secondly constructs classifier with the help of SVM iteratively. The reliable negative documents do not contain any positive features. In this technique, the reliable negative dataset is constructed by 1-DNF method which could be malfunctioning if sufficient positive dataset is not available. Therefore, this method cannot be well adopted for SMS generated through cell phones. It produces significantly good result when the positive data are big, but fails to demonstrate good accuracy when the positive dataset is low. In the same area, OSVM approach (Manevitz & Yousef, 2002) has been introduced which is based on strong mathematical foundation of SVM. This method deals with one class of positive dataset and unlabeled dataset. Though it has the same advantage over SVM as scalability on different number of dimensions and standard nonlinear classification using kernel trick, it does not perform well unless the system is fed with large number of positive dataset to create an accurate class boundary especially in high dimensional spaces. The reason is that the boundary support vector comes from only positive dataset.

Though the results of these techniques seem highly promising, there are some limitations. Spy-EM needs prior probabilities to feed

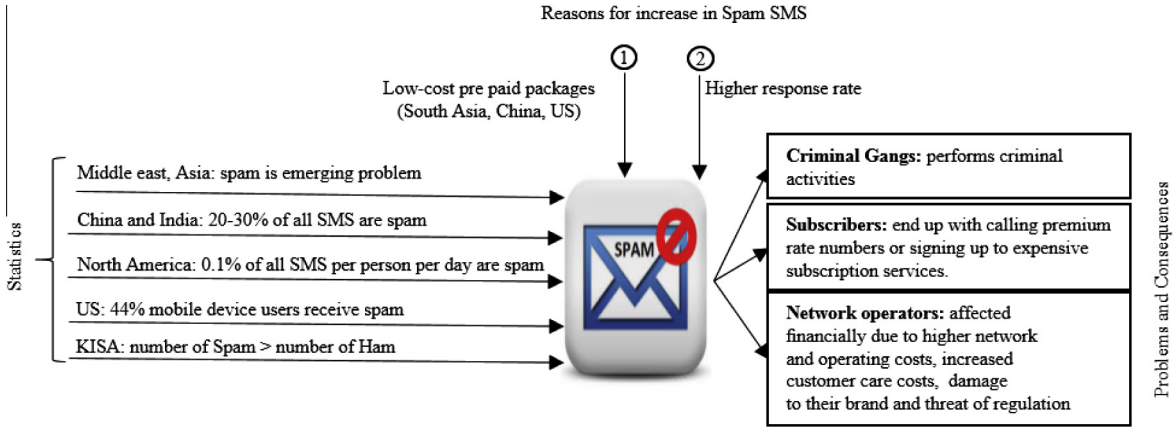


Fig. 1. Statistics, reasons and consequences of Spam SMS.

naive Bayes whereas PEBL does not work well if the number of positive training set is small. To solve these problems, we have proposed *FIEL*, a semi-supervised method to classify SMS data into ham and spam based on a novel approach using frequent itemset and ensemble learning. Our approach only needs a limited number of positive dataset, i.e., ham SMS and different amounts of unlabeled SMS. Initially, we run frequent pattern mining algorithm (Apriori) on ham dataset to find the frequent item based on the positive features under certain minimum support. Then, by running Apriori again under the same minimum support on the unlabeled SMS, we find more than one item based on frequent unlabeled features. Therefore, by extending the positive features and constructing new negative features from the unlabeled frequent features, we label the unlabeled SMS based on the frequency of positive and negative features. Finally, Multinomial Naive Bayes (Peng, Schuurmans, & Wang, 2004), Random Forest (Breiman, 2001) and LibSVM (Chang & Lin, 2011) are used as base classifier to construct an ensemble classifier and construct the model with the help of newly labeled data. To ensemble the classifiers, we use major voting scheme to predict the label of the corresponding SMS. Extensive experiments have been performed, which show that the proposed technique produces accurate classifier given that only positive class is known under certain minimum support. Using the methodology and techniques discussed here, our main contributions are:

- We propose a novel semi-supervised method that exploits frequent itemset and ensemble learning to classify SMS working with small amount of ham SMS without having spam SMS.
- The proposed method is evaluated on UCI SMS spam collection dataset, SMS spam collection Corpus v.0.1 Small and Big, where it performs better in small amount of positive dataset as compared to well-known semi-supervised methods, such as Spy-EM and PEBL.
- The proposed method achieves stability in terms of *accuracy* and *F score* as compared to Spy-EM and PEBL.

The remainder of this paper is organized as follows. Section 2 discusses the methodology of our proposed FIEL. Section 3 evaluates the FIEL approach with experiments and discusses them in details. Finally, Section 4 concludes the work done and draws future directions in this area.

2. frequent itemset and ensemble learning (FIEL)

In this section, we present *FIEL* methodology that works on ham and unlabeled SMS under certain minimum support. *FIEL* methodology consists of several steps and each step is elaborated in the subsequent subsections.

2.1. Architecture of the system

The architecture of our system is divided into four components: (a) *Preprocessing*, (b) *Feature Construction* and (c) *Labeling Unlabeled Dataset* and (d) *Ensemble Classification*. All the components are closely coupled with each other as shown in Fig. 2. Initially, positive dataset and unlabeled dataset are fed to *Preprocessing* step which consists of two sub parts: (i) *Elimination of Stop Word* (ii) *Stemming*. After successfully finishing the *Preprocessing* tasks, the positive and unlabeled dataset are fed to the *Feature construction* component. Here, the process starts with the *Positive Feature-set Construction*. Then with the help of unlabeled dataset, *Feature set Construction [+,−]* is done. After that, with the help of positive features and newly constructed features [+,−], the positive features set is expanded in *Expansion of Positive Features-set* component. After having the positive feature set expansion, Negative Features set is constructed with the help of positive feature set and feature

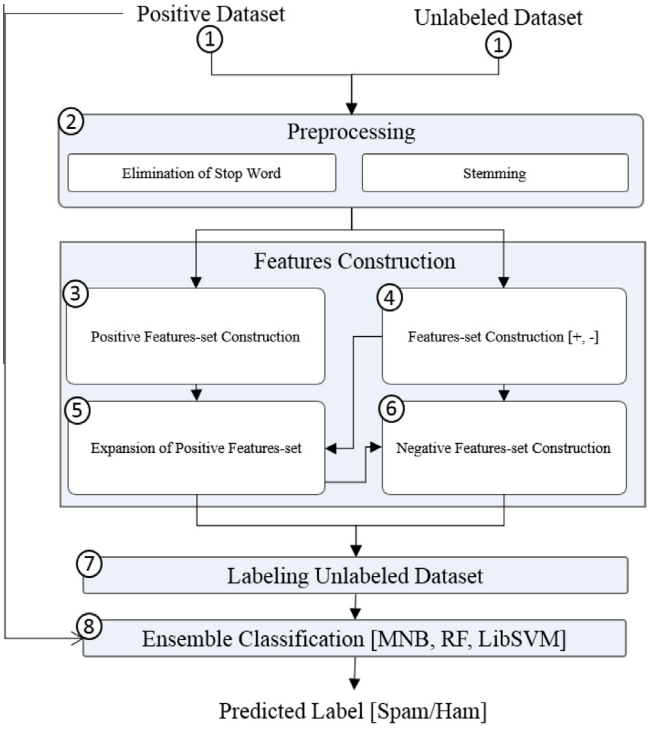


Fig. 2. Proposed architecture of FIEL for classification of SMS data.

set [+,−]. Once the positive and negative features are constructed, the *Feature Construction* step is finished and the newly generated features are ready to be fed to the next component *Labeling Unlabeled Dataset*. After labeling the unlabeled dataset, it is ready to feed the ensemble classifier along with the positive dataset. The ensemble classifier consists of three base classifiers: Multinomial Naive Bayes, Random Forest and LibSVM. Our ensemble classifier works on major voting scheme. The detailed discussion of each component is given bellow.

2.1.1. Preprocessing

In this step, we have used most common 25 verbs, 25 nouns, 25 adjective, 25 prepositions, 25 adverbs, 25 pronouns, 25 conjunctions, 25 numbers, 10 interjections and 100 words in pre-processing step to eliminate the unnecessary words from the SMS. These commonly used words do not play important role in classification techniques. Therefore, they are discarded these words. We collect this list from English Club official website.[1] For overcoming the redundancy of words, stemming is used. For this purpose, we have adopted Porter's algorithm (Porter, 1997).

2.1.2. Feature construction

Feature construction is one of the main tasks of our proposed system. The following sub-sections discuss this process in detail.

2.1.2.1. Positive features-set construction. In this step we construct positive features with the positive labeled dataset. To achieve this goal, we run Apriori algorithm to find out the one-item based frequent items. As our intention is to find out the one item based frequent item, we do not go into the details of the Apriori algorithm. We return from the Apriori algorithm once we get the desired one-item based on frequent items. Here, "item" means individual word. We treat every word as a unique item. For example, if an SMS seems like "*Ilo veML, doI*?", then we have a total of 5 words. But

[1] http://www.englishclub.com/vocabulary/common-words.htm.

as we consider the unique words, the total unique words or items are four which are "*I*", "*love*", "*ML*", and "*do*". So, after running Apriori algorithm under certain minimum support α on positive dataset (PD), we collect the items and enlist them into positive feature-set (PF). As we consider *FIEL* on cell phone, considering computational complexity and space is a severe factor, because most of the featured and smart cell phones have less computational power and run out of memory space. Algorithm 1 describes the process of positive feature construction in details.

Algorithm 1: Construction of Positive Features

Input: Labeled positive dataset (PD)
Output: Positive feature set (PF)

$PF \leftarrow \emptyset$
$C_1 \leftarrow$ Generate Candidate 1-itemset from PD
$L_1 \leftarrow$ Generate frequent 1-itemset from C_1 under minimum support α
$FI \leftarrow L_1$.
for *i = 1 to size(FI)* **do**
  $PF \leftarrow PF \cup FI_i$;
return PF

2.1.2.2. Features-set construction [+, −]. In this phase, unlabeled features as known as unlabeled frequent itemset *(UFI)* are generated from unlabeled dataset (*UD*) with the help of Apriori algorithm. Basically, Apriori is operated with the same threshold as known as minimum support α. But, in this process, rather than running the algorithm for finding the one-item based frequent itemset, we continue the process for finding two-items based frequent itemset. The details of this step is described in Algorithm 2.

Algorithm 2: Construction of Unlabeled Features

Input: Unlabeled dataset set (UD)
Output: Unlabeled frequent itemset (UFI)

$UFI \leftarrow \emptyset$
$C_1 \leftarrow$ Generate Candidate 1-itemset from UD
$L_1 \leftarrow$ Generate frequent 1-itemset from C_1 under minimum support α
$C_2 \leftarrow$ candidates generated from L_1 `/* Cartesian product of` $L_1 \times L_1$ `and eliminating any 1 size itemset that is not frequent */`;
increment the count of all candidates in C_2 that are contained in UD
$L_2 \leftarrow$ candidates in C_2 having satisfied with α;
for *k = 1 to size(L)* **do**
  $UFI \leftarrow UFI \cup L_k$;
return UFI

2.1.2.3. Expansion of positive features-set. Once we construct the positive features *PF* from Algorithm 1 and unlabeled features *UFI* from Algorithm 2, we continue this step because it needs *PF* and *UFI*. Once we have *UFI* and find the maximum size of it, we enumerate, two-itemset based *UFI* which also eventually means UFI_2 and then find the total size of UFI_2 to expand the positive features *PF*. Thus, if any one of the item is a subset of positive feature *PF*, then we include every item of that frequent itemset to *PF* and thus *PF* set expand gradually. Initially, *UFI* is empty and at later stage, it only consists of generated frequent items that we collect by performing Apriori algorithm. The full operation is shown in Algorithm 3.

Algorithm 3: Augmentation of Positive Features

Input: Positive feature (PF) and unlabeled frequent itemset (UFI)
Output: Positive feature set (PF)

for *i = 1 to size(PF)* **do**
  for *j = 1 to size*(UFI_2) **do**
    if $PF_i \in UFI_{2j}$ **then**
      $PF \leftarrow PF \cup UFI_{2j}$
return PF

2.1.2.4. Negative features-set construction. In this step, we construct the negative features set (*NF*) from the unlabeled frequent itemset *UFI*. Once we get the full set of PF performing Algorithm 2, we begin to construct *NF*. Initially *NF* is considered as null and then we enumerate *UFI* through 1 to maximum size of the *UFI*. In a single iteration, if none of the item of UFI_i belongs to *PF*, then the corresponding UFI_i is included to negative feature set *NF*. Thus, as the iteration increases, *NF* is continuously expanded. Algorithm 4 elaborates this process.

Algorithm 4: Constructing of Negative Features

Input: Positive feature (PF) and unlabeled frequent itemset (UFI)
Output: Negative feature set (NF)

$NF \leftarrow \emptyset$
Import *UFI Algorithm 2* and PF from *Algorithm 3*
for *i = 1 to size(UFI)* **do**
  if $UFI_i \notin PF$ **then**
    $NF \leftarrow NF \cup UFI_i$
return NF

2.1.3. Labeling unlabeled dataset

As we get the positive and negative feature set from Algorithms 3 and 4 respectively, this step describes how the unlabeled dataset are labeled to feed the learning algorithm. In every SMS of *UD*, the total number of words is counted. Then, we check the individual word whether it is subset of *PF* or *NF* and if it is, then increase the frequency of the corresponding feature set. We run this iteration till the last word of the corresponding SMS and when it is done, we count the total number of frequencies of *PF* and *NF* as f_p and f_n. If $f_p > f_n$, then we label the corresponding unlabeled dataset (UD_i) as ham (*LT*) and vice versa if $f_p < f_n$, UD_i as spam LT. We continue this process till the last SMS of the *UD* is processed. Algorithm 5 describes this process.

Algorithm 5: Tagging of unlabeled training dataset

Input: Positive feature PF, Negative feature (NF) and unlabeled dataset (UD)
Output: Labeled training set (LT)

for *i = 1 to size(UD)* **do**
  $f_p^i \leftarrow 0$ and $f_n^i \leftarrow 0$;
  for *j = 1 to size*(UD_i) **do**
    if $w_j^i \in PF$ **then**
      $f_p^i + +$;
    else
      if $w_j^i \in NF$ **then**
        $f_n^i + +$;
    if $f_p^i > f_n^i$ **then**
      Label UD_i as Ham LT;
    else
      Label UD_i as Spam LT;
return LT

2.1.4. Ensemble classification

After having the newly labeled training set *LT*, we train all the three classifiers (Multinomial Naive Bayes, Random Forest and LibSVM) with the help of *PD* and *LT*. Once we train our system with the training dataset, we test every SMS of unlabeled dataset *UD* and classify the corresponding SMS with the help of majority voting scheme. Algorithm 6 describes the detailed process of ensemble classification.

Algorithm 6: Construction of Ensemble Classifier

Input: Labeled positive dataset (PD), labeled training set (LT), unlabeled dataset (UD)
Output: Predicted label (PL)

Train Multinomial Naive Bayes, Random Forest and LibSVM by PD and LT.
for *i = 1 to size(UD)* **do**
  Test UD_i by Multinomial Naive Bayes
  Test UD_i by Random Forest
  Test UD_i by LibSVM
  Classify UD_i to *Spam* or *Ham* by the majority voting scheme by above three classifiers and label PL_i.
return PL

2.2. Computational complexity analysis of proposed FIEL approach

The variables used in formulating the computational complexity are defined as follow:

- MW mean number of unique words in an SMS after pre-processing.
- TW_p total unique words in *PD* satisfied by minimum support alpha in Algorithm 1.
- TW_u total unique words in *UD* satisfied by minimum support α.
- ME_u mean elements in *UFI*

Assumption. Since pre-processing is very small compare to the whole complexity analysis and almost every classification technique uses similar techniques, we omit that in our calculation. Here is the computational complexities of Algorithms 1–6.

Computational complexity of Algorithm 1: $O(MW \times Size(TW_p) \times Size(PD))$.
Computational complexity of Algorithm 2: $O(Size(TW_u) \times Size(TW_u) \times Size(UD) \times MW)$.
Computational complexity of Algorithm 3: $O(Size(TW_p) \times Size(UFI) \times ME_u)$.
Computational complexity of Algorithm 4: $O(Size(newPF) \times Size(UFI) \times ME_u)$.
Computational complexity of Algorithm 5: $O(Size(TW_p \cup TW_u) \times MW \times Size(UD))$.

The computational complexity of Multinomial Naive Bayes (MNB), Random Forest (RF) and LibSVM can be found in McCallum and Nigam (1998), Breiman (2001) and Chang and Lin (2011) respectively.

Computational complexity of Algorithm 6: $O(Size(UD) \times \{MNB + RF + LibSVM\} \wr complexity)$.

The final computational complexity of *FIEL* is the summation of Algorithms 1–6.

3. Experimental results and discussion

In this section, the empirical evaluation of *FIEL* is made in terms of experimental dataset, performance measure, minimum threshold and comparison with existing approaches.

3.1. Experimental dataset

For conducting the experiments, we used three different dataset: (i) UCI repository called as "SMS Spam Collection Data Set"[2] (ii) SMS Spam Corpus v.0.1 Small and (iii) SMS Spam Corpus v.0.1 Big[3] (Almeida, Hidalgo, & Yamakami, 2011). Every message of these repository consists of two parts: *(a)* Label *(b)* body of the SMS. The label of SMS is followed by the content of the SMS. Detailed information of these repositories are described in Table 1.

3.2. Performance measures

As our task is to identify positive SMS from the unlabeled SMS, it is relatively suitable to use information retrieval measures. There are two types of information retrieval measures and they are *F score* and *break-even point*. As *F score* measures the performance of a system on a particular class, it is more convenient to use and to evaluate our system. Since *break-even point* is the value where recall and precision are equal, it does not depict a good indication of classification performance. We also incorporate accuracy to predict the overall performance. However, *F score* reflects the average effects of both *precision* and *recall*. *Accuracy* and *F score* measurements are illustrated bellow:

Table 1
Description of the dataset used in FIEL experiments.

Dataset name	Key feature
UCI SMS Spam Collection Data Set	Total of 4,827 SMS legitimate messages (86.6%) and a total of 747 (13.4%) spam messages
SMS Spam Corpus v.0.1 Small	Total of 1,002 SMS legitimate messages (92.4%) and a total of 83 (7.6%) spam messages
SMS Spam Corpus v.0.1 Big	Total of 1,002 SMS legitimate messages (75.6%) and a total of 323 (24.4%) spam messages

TP (true positive)	the number of examples correctly classified to the class.
TN (true negative)	the number of examples correctly rejected from that class.
FP (false positive)	the number of examples incorrectly rejected to that class.
FN (false negative)	the number of examples incorrectly classified to that class.

By using these terms, we can form the measurement factors as:

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

$$Precision = \frac{\#of\ correct\ positive\ predictions}{\#of\ positive\ predictions} = \frac{TP}{TP + FP} \quad (2)$$

$$Recall = \frac{\#of\ correct\ positive\ predictions}{\#of\ positive\ data} = \frac{TP}{TP + FN} \quad (3)$$

$$F\ score = 2\frac{Precision\ Recall}{Precision + Recall} \quad (4)$$

To evaluate the effectiveness of *FIEL*, we compare *F score* and *accuracy* of our system with Spy-EM and PEBL. We observed that *FIEL* produces comparatively good result with low amount of positive labeled dataset, competitive result when the amount of positive labeled dataset is high.

3.3. Finding minimum threshold of α

Before going to have the comparison, we present the importance of choosing the right value while for minimum support denoted by α to operate the system which produces significant important factor of *F scores* and *accuracy*. To conduct this experiment, we consider the whole *UCI dataset* which is mentioned earlier. The *accuracy* and *F-scores* of our system largely rely on different positive and negative features and these features are generated by frequent pattern mining algorithm (Apriori). To find the frequent itemset, known as features, we use Apriori algorithm on different α and thus we have different amount of positive and negative features. When the threshold α is low, we have huge number of positive and negative features. On the other hand, as we increase α, the number of features is also getting lower accordingly. Since, the frequency of the features determines the predicted level which is fed to the final ensemble classifier of *FIEL*, the right amount of features play a crucial role to predict the accurate label. When α is lower, huge number of features are generated and as we know, the features which occur in large number of SMS, have the tendency to become very important factor to determine the right label. Hence when α values are increased, the *accuracy* and *F scores* are also gradually increased. But at the same time, the number of features is also decreased accordingly. Thus at a certain moment,

[2] Sms spam collection data set. Last Retrieved April 25, 2014. from http://archive.ics.uci.edu/ml/datasets/SMS+Spam+Collection.
[3] http://www.esp.uem.es/jmgomez/smsspamcorpus/.

Table 2
α factor of *FIEL*'s accuracy and F-score.

Minimum Support(α)	FIEL *Accuracy*	FIEL *F-score*
20	76.6735	0.8539
30	85.2936	0.8883
40	64.2352	0.7683
50	75.7441	0.8337
60	81.6885	0.8629
70	85.6242	0.8963
80	80.1540	0.8570
90	83.4783	0.8749
100	85.4908	0.8962
150	86.7725	0.9093
250	87.8130	0.9179
350	88.2325	0.9293
450	91.5659	0.9624
550	89.5913	0.9439
650	13.4019	0
700	13.4038	0
800	13.4097	0
900	13.4023	0

there will be no positive or negative features and our system's performance will be drastically decreased. Therefore, according to our algorithm, the whole unlabeled dataset are considered as negative dataset. Hence, our ensemble classifier will be fed only by the given positive dataset and negative dataset (whole unlabeled dataset) which produces very poor performance overall.

To illustrate this scenario, we consider only 150 SMS as positive SMS and whole UCI repository as unlabeled dataset to perform our experiments on various values of α. We can clearly visualize from Table 2 that, when α is very low at the initial point, our overall *accuracy* and *F score* is not quite satisfactory. This performance is poor because of huge number of unnecessary features which does not predict the desired label of the corresponding SMS. Here, when the value of α is from 20 to 100, our system does not produce the desired good output. Though, initially our system is highly unstable to predict the overall *accuracy* and *F score*, it gradually increases the overall performance since α values also increases. Hence, the system begins to perform consistently well, when the threshold is gradually increased. However, at a certain point, there will be no features as running Apriori fails to generate the frequent itemset a.k.a. features and thus the system performance collapses drastically. From Table 2 and Fig. 3, we also notice that when α is 650, *FIEL*'s accuracy sharply falls down to 13.401507 due to unavailable of features and hence it remains steady for the next α values. The poor performance remains constant, because from this point, the whole unlabeled dataset is considered as negative dataset. We also noticed that, *F score* is drastically decreased to 0 and remains constant because of not finding the positive dataset in the unlabeled dataset.

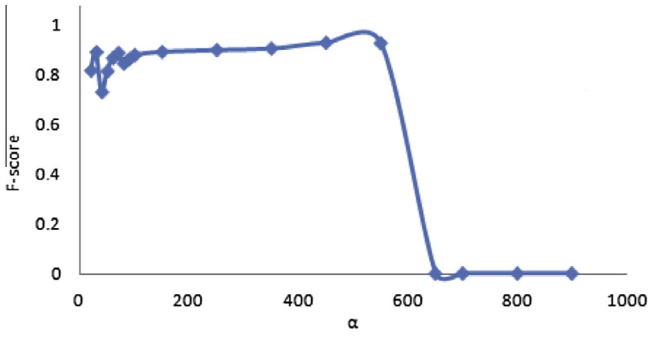


Fig. 3. α factor on F-score.

3.4. Experiment and comparison

We implement the frequent pattern mining algorithm (Apriori) and the core Algorithms 1–6 by Java SE in *IntelCorei5*^{TM} machine with 3 GB RAM. For the implementation of machine learning algorithm like Multinomial Naive Bayes, Random Forest and LibSVM, we have used WEKA (Holmes, Donkin, & Witten, 1994). When we conduct with LibSVM, we select the kernel function as *RadialBasisFunction*. Besides this, we also filtered the texts by *StringToWordVector* and for stemming purpose we selected *IteratedLovingsStemmer*.

In order to make comparison, we include the classification results of *Spy-EM*, *PEBL* and *FIEL*. For *Spy-EM* and *PEBL*, both the systems have two steps. In the first step, both the systems identify a set of reliable negative dataset form the unlabeled dataset. In this step, *Spy-EM* uses a *Spy* technique whereas *PEBL* uses a technique called *1-DNF*. In the second step, *EM* uses Expectation Maximization *(EM)* algorithm with naive Bayes classifier whereas *PEBL* uses *SVM* and always uses the last classifier at convergence which sometimes leads to poor performance of the system. We use *Spy-EM* from a publicly available site at http://www.cs.uic.edu/liub/S-EM/S-EM-download.html (provided by corresponding author of *Spy-EM*). However, we could not find any publicly available *PEBL* system and thus we implemented *PEBL* with the help of *SVM*^{light} system (Joachims, 1999).

We performed our experimentation on three different dataset to verify the performance of our proposed *FIEL* system, making comparison with Spy-EM and PEBL. However, each dataset differs from the other in terms of spam and ham SMS ratio. In these experiments, our task is to find the positive SMS from the unlabeled SMS. Firstly, we randomly select $\varphi\%$ of the SMS from the positive SMS and are considered as positive set (P). The remaining $(1-\varphi\%)$ SMS, from both of the classes construct unlabeled set (U). Besides this, we also treat (U) as the test set for our experimentation which means that we do not use separate test set. Throughout this experiment, we use different values of φ which starts from 5% to 65% having equal interval of 10% to construct different ratio of positive dataset.

Tables 3–5 demonstrates the classification result of various techniques including *FIEL* in terms of *F value* and *accuracy* in UCI

Table 3
F-score and *accuracy* comparison among *PEBL*, *Spy-EM* and *FIEL* in UCI repository.

φ (%)	PEBL		Spy-EM		FIEL	
	F-score	Accuracy	F-score	Accuracy	F-score	Accuracy
5	0.7705	67.29	0.8429	76.32	0.9517	91.60
15	0.7359	63.32	0.8793	81.26	0.9511	91.57
25	0.8637	79.1	0.9086	85.44	0.9651	93.95
35	0.8794	81.23	0.9261	87.49	0.9713	95.01
45	0.8355	75.42	0.9385	90.32	0.9714	95.03
55	0.9515	94.82	0.9564	92.74	0.9771	95.99
65	0.9747	96.01	0.9677	95.76	0.9783	96.21

Table 4
F-score and *accuracy* comparison among *PEBL*, *Spy-EM* and *FIEL* in Corpus v.0.1 Small.

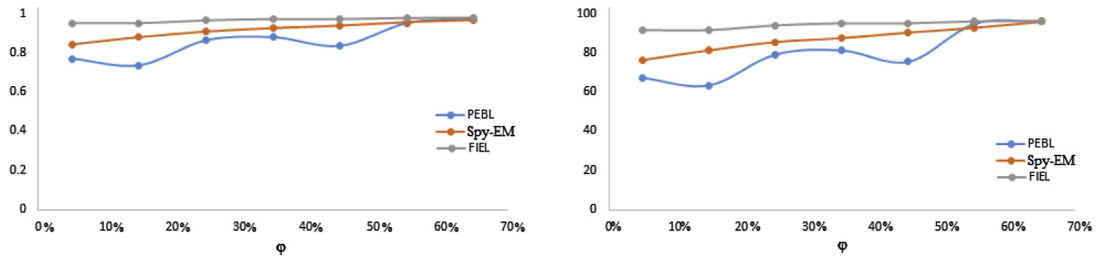
φ (%)	PEBL		Spy-EM		FIEL	
	F-score	Accuracy	F-score	Accuracy	F-score	Accuracy
5	0.1632	15.77	0.3739	27.46	0.5913	41.97
15	0.2945	23.52	0.4057	30.87	0.7057	54.52
25	0.4089	31.18	0.6173	48.47	0.7202	56.27
35	0.5267	41.31	0.6195	48.94	0.7949	65.96
45	0.6699	50.37	0.8034	70.28	0.8738	77.58
55	0.9319	87.27	0.8783	79.90	0.8819	78.87
65	0.9671	94.10	0.9389	88.56	0.9028	82.29

Table 5
F-score and *accuracy* comparison among *PEBL*, *Spy-EM* and *FIEL* in Corpus v.0.1 Big.

φ (%)	PEBL		Spy-EM		FIEL	
	F-score	Accuracy	F-score	Accuracy	F-score	Accuracy
5	0.3964	38.82	0.4425	44.67	0.7121	59.06
15	0.2872	37.01	0.4326	44.98	0.7127	60.50
25	0.3836	42.22	0.5676	54.11	0.7326	61.78
35	0.5093	50.08	0.7983	73.25	0.8176	73.72
45	0.6689	62.16	0.8441	79.54	0.8966	84.06
55	0.9673	94.66	0.9593	93.96	0.9098	85.95
65	0.9709	96.64	0.9672	95.09	0.9254	88.21

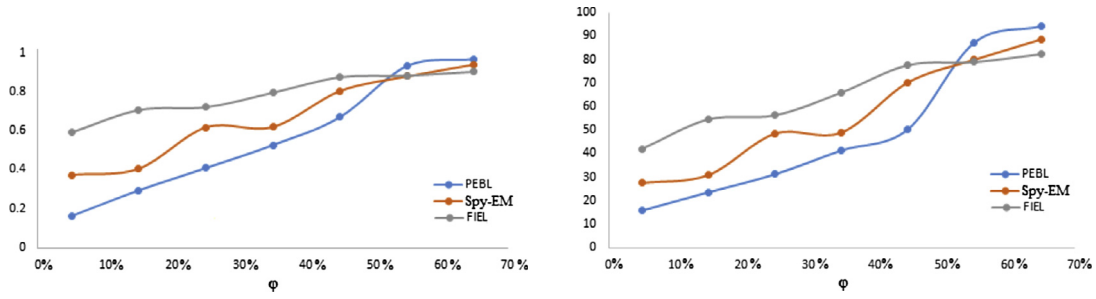
repository, Corpus v.0.1 Small and Corpus v.0.2 Big respectively. We observe that *PEBL* performs well when the positive set is big enough i.e. φ is equal to 65% in UCI repository and equal or more than 55% in Corpus small and big repository and fails to produce good enough when φ is smaller in comparison to big positive dataset. It happens because *PEBL* performs badly when φ is small as the strategy of extracting the initial set of strong negative dataset could go wrong without the help of sufficient positive dataset. Basically when the positive dataset is too small, *1-DNF* method has the tendency to assume most of the positive dataset as negative and thus produce bad results. Moreover, from Figs. 4–6 we can predict that this *PEBL* system is unstable and thus it fails to produce a good result when the dataset is not large enough. Here, we can observe from Fig. 4 that when φ = 45%, which is quite big enough, it still performs poorly compared to the two other methods. Moreover from Fig. 5 and 6, we also observe that when φ is less than or equal to 45% it does not have significant result to compare with the other methods. On the other hand, *Spy-EM*'s performance is stable throughout the whole range of condition. Though the performance of this system is stable, but it fails to depict the overall good performance than the other systems because *EM* uses naive Bayes, which is weaker classifier than the others, whereas our system is a combined classifier of Multinomial Naive Bayes, Random Forest and LibSVM; and *PEBL* uses SVM. But we can also predict that *Spy-EM* performs good enough than *PEBL* when φ is i.e. (φ is less than or equal to 65% in Fig. 4 and φ is less than or equal to 55% both in Figs. 5 and 6).

As expected, all the techniques improve their results when the positive dataset are gradually increased. From Tables 3–5, Figs. 4–



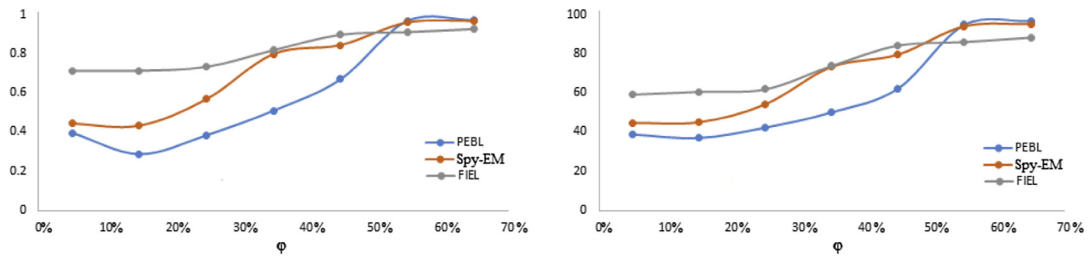
(a) F-score comparison on different φ settings (b) Accuracy comparison on different φ settings

Fig. 4. F-score and accuracy comparison on different φ settings in UCI repository.



(a) F-score comparison on different φ settings (b) Accuracy comparison on different φ settings

Fig. 5. F-score and accuracy comparison on different φ settings in Corpus v.0.1 Small repository.



(a) F-score comparison on different φ settings (b) Accuracy comparison on different φ settings

Fig. 6. F-score and accuracy comparison on different φ settings in Corpus v.0.1 Big repository.

6; we observe that *FIEL* consistently performs well throughout different conditions compared to both the systems except when φ is too high. Though *FIEL* beats other two approaches in UCI dataset as demonstrated in Table 4 through all ratio of φ, it fails to overcome with other dataset. In experimenting with different dataset, we assume minimum support (α) as 40, which eventually means out of different unlabeled SMS, words that occur at least 40 times will be considered as a feature. Even though we notice that our system defeats other systems by producing consistently good result through different ratio of positive dataset in UCI repository, but fails to beat *Spy-EM* till 55% while it is not possible to defeat *PEBL* because it produces extremely good result when φ is very big as we know it produces a good result when φ is big. We also notice that our algorithm has produced extremely good results when the positive dataset is low enough, compared to the other two systems and it continues on various constraints. As we consider our algorithm with frequent pattern mining concept, we notice that there are huge similarities among the ham and so on in spam. There are many common words among these messages that certain words seem to occur in large number of SMS, which eventually boost up our *F scores* and *accuracy* also. Thus, for a certain percentage of φ, when both system's performance is degraded, our system still produces stable and very good result through all ranges. Hence, as we discussed earlier, our system for SMS classification, is efficient with low amount of positive SMS. Most of the featured phone and other smart phones have less memory and computation power and hence these cell phones require such a system which can handle very low number of SMS to successfully detect the right label of the corresponding SMS. Based on the above, *FIEL* is ideal for this situation. Therefore, we notice that when the positive set is small, the improvement is exceptionally good.

4. Conclusions and future work

In this paper, we presented a frequent itemset and ensemble learning based semi-supervised approach to classify SMS with the help of small amount of ham SMS and unlabeled SMS without having spam SMS. Initially, positive features are calculated using certain minimum support in ham SMS and later on, these features are gradually expanded with the help of unlabeled SMS. Moreover, the negative features are also generated by positive features and unlabeled SMS. Therefore, these unlabeled instances (SMS) are labelled with the argument that maximizes the mode of the features. Finally these newly labeled dataset are fed to train the base classifiers such as Multinomial Naive Bayes, Random Forest and LibSVM. This method is evaluated on UCI SMS spam collection dataset; SMS spam collection Corpus v.0.1 Small and Big dataset. The proposed approach produces good result especially when the positive instances ratio is low and throughout different positive dataset ratio it is highly stable. However, the minimum support for finding frequent itemset varies largely that depends on the dataset volume.

This research has a number of possible future extensions that can be carried out as future research work. When we experiment on different dataset, exploiting different minimum support produces different result. Hence choosing appropriate minimum support for different corpora would be future interesting research. Furthermore, as we use WEKA tool for implementation of different base learners with filtered classifier option, exploiting the training and testing phase of these learners with different frequent itemset along with individual words would be a good research topic if we implement these learners by ourselves. Though this research work is semi-supervised methodology, but a good research direction would be to used it for supervised learning. This research work is only conducted for English language, but can also be carried out for other languages in future.

Acknowledgment

We would like to express our deep gratitude to anonymous reviewers for their insightful comments on the paper. We believe that their constructive suggestions had significantly enriched the quality of the paper. We have tried our best to modify the paper according to their recommendations. This work was supported by the Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education, Science, and Technology (2010-0012609).

References

Ahmed, I., Guan, D., & Chung, T. C. (2014). Sms classification based on naive bayes classifier and apriori algorithm frequent itemset. *International Journal of Machine Learning and Computing, 4*, 183–187.

Almeida, T. A., Hidalgo, J. M. G., & Yamakami, A. (2011). Contributions to the study of sms spam filtering: New collection and results. In *Proceedings of the 11th ACM symposium on document engineering* (pp. 259–262). NY, USA: ACM.

Anthony, M., & Bartlett, P. L. (1999). *Neural network learning: Theoretical foundations*. NY, USA: Cambridge University Press.

Breiman, L. (2001). Random forests. *Machine Learning, 45*, 5–32.

Chang, C.-C., & Lin, C.-J. (2011). Libsvm: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology (TIST), 2*.

Cover, T. M., & Hart, P. E. (1967). Nearest neighbor pattern classification. *IEEE Transactions on Information Theory, 13*, 21?–27.

Delany, S. J., Buckley, M., & Greene, D. (2012). Sms spam filtering: Methods and data. *Expert Systems with Applications, 39*, 9899–9908.

Dempster, A. P., Laird, N. M., & Rubin, D. B. (1977). Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society, 39*, 1–38.

Denis, F. (1998). Pac learning from positive statistical queries. In *Proceedings of the 9th international conference on algorithmic learning theory* (pp. 112–126). Berlin Heidelberg, Otzenhausen, Germany: Springer.

Gsma. (2011a). Operator faqs. gsma spam reporting service.

Gsma. (2011b). Sms spam and mobile messaging attacks – introduction. trends and examples. gsma spam reporting service. (January, 2011). Blog. Retrieved April 24, 2014.

He, P., Sun, Y., & Zheng, W. (2008). Filtering short message spam of group sending using captcha. In *Proceedings of the first international workshop on knowledge discovery and data mining* (pp. 558–561). Washington, DC, USA: IEEE Computer Society.

Holmes, G., Donkin, A., & Witten, I. H. (1994). Weka: A machine learning workbench. In *Proceedings of the 2nd Australian and New Zealand international conference on intelligent information systems* (pp. 357–361). Brisbane, Qld. Australia: IEEE publisher.

Joachims, T. (1999). *Making large-scale support vector machine learning practical. Advances in Kernel methods*. MA, USA: MIT Press Cambridge.

Joachims, T., IT, G. F., & Augustin, S. (2001). A statistical learning model of text classification with support vector machines. In *Proceedings of the 24th annual international ACM SIGIR conference on research and development in information retrieval* (pp. 128–136). NY, USA: ACM.

Kearns, M. (1998). Efficient noise-tolerant learning from statistical queries. *Journal of the ACM (JACM), 45*, 983–1006.

Lee, K., & Choi, D. (2012). Mobile junk message filter reflecting user preference. *Ksii Transactions on Internet and Information Systems, 6*, 2849–2865.

Lee, K. -J., Yeom, J. -K., Choi, D. -J., & Kang, D. -W. (2011). Personalized sms spam filtering system for smartphone user. In *Proceedings of the KSII 3rd international conference on internet (ICONI)*.

Letouzey, F., Denis, F., & Gilleron, R. (2000). Learning from positive and unlabeled examples. In *Proceedings of the 11th international conference, ALT 2000 Sydney, Australia, December 11–13, 2000* (pp. 71–85). Berlin Heidelberg: Springer.

Lewis, D. D. (1998). Naive (bayes) at forty: The independence assumption in information retrieval. In *Proceedings of the 10th European conference on machine learning* (pp. 4–15). New York: Springer.

Liu, B., Lee, W. S., Yu, P. S., & Li, X. (2002). Partially supervised classification of text documents. In *Proceedings of the 19th ICMLC* (pp. 8–12). San Francisco, CA, USA: Morgan Kaufmann Publishers Inc.

Manevitz, L. M., & Yousef, M. (2002). One-class svms for document classification. *The Journal of Machine Learning Research, 2*, 139–154.

McCallum, A., & Nigam, K. (1998). A comparison of event models for naive bayes text classification. In *Proceedings of the ICMLC/AAAI-98 workshop on learning for text categorization, Madison, Wisconsin*.

Muggleton, S. (1997). Learning from positive data. In *Proceedings of the 6th international workshop. ILP-lecture notes in computer science* (Vol. 1314, pp. 358–376). Berlin Heidelberg, Stockholm, Sweden: Springer.

Najadat, H., Abdulla, N., Abooraig, R., & Nawasrah, S. (2014). Mobile sms spam filtering based on mixing classifiers. *International Journal of Advanced Computing Research, 1*.

Ng, A. Y., & Jordan, M. I. (2001). On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes. In *Proceedings of the advances in neural information processing*.

Nuruzzaman, M. T., Lee, C., & Choi, D. (2011). Independent and personal sms spam filtering. In *Proceedings of the 11th IEEE international conference on computer and information technology* (pp. 429–435). Berlin Heidelberg, Pafos: Springer.

Peng, F., Schuurmans, D., & Wang, S. (2004). Augmenting naive bayes classifiers with statistical language models. *Information Retrieval, 7*, 317–345.

Porter, M. F. (1997). *Readings in information retrieval*. CA, USA: Morgan Kaufman Publishers Inc.

Qian, W., Xue, H., & Xiaoyu, W. (2009). Studying of classifying junk messages based on the data mining. In *Proceedings of the international conference on management and service science, Sept. 2009* (pp. 1–4). IEEE Press.

Shahshahani, B. M., & Landgrebe, D. A. (1994). The effect of unlabeled samples in reducing the small sample size problem and mitigating the hughes phenomenon. *IEEE Transactions on Geo-science and Remote Sensing, 32*, 1087–1095.

Shi, L., Ma, X., Xi, L., Duan, Q., & Zhao, J. (2011). Rough set and ensemble learning based semi-supervised algorithm for text classification. *Expert Systems with Applications, 38*, 6300–6306.

Sohn, D.-N., Lee, J.-T., Han, K.-S., & Rim, H.-C. (2012). Content-based mobile spam classification using stylistically motivated features. *Journal Pattern Recognition Letters, 33*, 364–369.

Valiant, L. G. (1984). A theory of the learnable. *Communications of the ACM, 27*, 1134–1142.

Xu, Q., Xiang, E. W., Yang, Q., Du, J., & Zhong, J. (2012). Sms spam detection using noncontent features. *IEEE Intelligent Systems, 27*, 44–51.

Yu, H., Han, J., & Chang, K. C.-C. (2004). Pebl: Web page classification without negative examples. *IEEE Transactions on Knowledge and Data Engineering, 16*, 70?–81.