A Collaborative Learning Method For Spam Filtering

Hsiu-Sen Chiang, Jui-Chi Shen[*], Dong-Her Shih, and Chia-Shyang Lin

National Yunlin University of Science and Technology, Department of Information Management, 123, Section 3, University Road, Touliu, Yunlin, Taiwan
Kaohsiung Hospitality College, Sung-Ho Rd., Shiao-Kang. Kaohsiung, Taiwan[*]
{g9023728,shihdh,g9223728}@yuntech.edu.tw

Abstract. Spam, also known as Unsolicited Commercial Email (UCE), is the bane of email communication. It has brought enormous cost for the companies or users that use Internet. Spam filtering has made considerable progress in recent years. The predominant approaches are data mining methods and machine learning methods. Researchers have largely concentrated on either one of the approaches since a principled unifying framework is still lacking. This paper suggests that both approaches can be combined under a collaborative learning framework. We propose a collaborative learning algorithm that parallelly uses three different machine learning methods. The resultant algorithm is simple and understandable, and offers a principled solution to combine content-based filtering and collaborative filtering. Within our algorithm, we are now able to interpret various existing techniques from a unifying point of view. Finally we demonstrate the success of the proposed collaborative filtering methods in the experiment.

1 Introduction

Unsolicited Bulk Email (UBE), also referred to as Unsolicited Commercial Email (UCE), is commonly called spam or junk mail. Spamming is the practice of sending mass mailings to large numbers of people who have no relationship with the sender and who didn't ask for such mail. Different reasons motivate spammers, but the spam exists primarily because of the low cost. Spam filtering denotes a family of techniques that help users to find the right emails while filtering out undesired ones. The filter can be implemented at either server side (mail transport agent, MTA) or the end user side (mail user agent, MUA).

In this paper, we parallelly employ three methods to the filtering work with our collaborative learning algorithm. It is not easy to judge whether a mail is spam or not by any single one method. That is because under a single one method, some features occurred in spam will also occur in legitimate mail. There are many technologies and researches on spam mail detection, most of them use a single client agent to filter mail, and lack of the ability of collaboration. The junk mail has the characteristic that massively spreads, but the present client filter agents don't fully utilize this characteristic. Hence, we propose a distributed architecture to learn collaboratively with the knowledge of spam in improving the ability of client spam detection.

© A. Gelbukh, R. Monroy. (Eds.)
Advances in Artificial Intelligence Theory
Research on Computing Science 16, 2005, pp. 119-128

2 Related Work

Spam filtering has made considerable progress in recent years. Many data mining and machine learning researchers have worked on spam detection and filtering. The problem is popular enough that it has been the subject of a Data Mining Cup contest [4] as well as numerous class projects. Bayesian analysis has been very popular [1; 14], but researchers have also used SVMs [5], decisions trees [19], rule learning [3; 18] and even genetic programming [12]. Some widely used methods for anti-spam are list as below:

Blacklist. A blacklist spam filter can be a DNS-based (DNS-based Blacklists, DNSBL) or email-address-based blacklist. The mechanism behind the method is keeping the source of spammers in a database. The legal mail server can access to the database then deny receiving the messages from the source of spam. Blacklist is very useful at ISP level. But it has several weaknesses. First, more than half of the spam mail servers are not in the blacklist. Second, the effect of blacklist depended on the administrator of the blacklist. If the black is wrong, it is possible that filtering legitimate mails [11].

Signature Based Filtering. The method of signature based filtering is comparing incoming mails with the spam prior received. In order to know whether two mails are the same, the filter calculates "signatures" for them. Signature based filter rarely blocks legitimate mails, but the weakness is that spammers can add random stuff to each copy of spam and give it a distinct signature, so that they can trick the signature based filters.

Rule Based Filtering. Rule-based filters try to discover the patterns, e.g. words or phrases, malformed headers and misleading dates. For example, RIPPER is based on keyword-spotting rules, which is a rule set generated by users' manual setting. SpamAssassin, popularly used open source spam filter, uses a large set of heuristic rules. But the main disadvantage of rule-based filters is that they tend to have high false positive rates. For example, SpamAssassin has a problem with false positives rates [3] [17] [18].

Text Classification Filtering. A text classification filter uses text classification technique to filter spam. There have been several studies in this application, which include keyword-based, phrase-based, and character-based. Naïve Bayes-based [8; 18] method is also another efficient approach of keyword and phrase-based. It is a probabilistic classification by using features extracted from emails. Additionally, Boosting [21], Support Vector Machine [5], Rocchio [10], and decision tree based on ID3, C4.5, or C5 [19] algorithm, can be identified as the representative methods to analyze keywords in email.

Multi-agent Based Filtering. Due to the massive-distribute characteristic of spam, Multi-agent-based Filter, a new architecture, was proposed. The main feature is that clients can exchange knowledge about spam. Metzger et al. proposed an architecture that combines signature-based filter, SVM text classification, and multi-agent system [15].

3 Methodology

In this section, we will describe three statistical methods for individual learning. Each algorithm has different features. Then we will describe the preprocessing for the collaborative filtering and how our collaborative filtering method works.

3.1 Individual learning method

We employed three wildly used methods in the collaborative spam filtering process for individual learning. These algorithms include Naïve Bayes, Fisher's probability combination method, and Chi-square by degree of freedom. Fig. 1 shows a typical individual learning model for spam filtering.

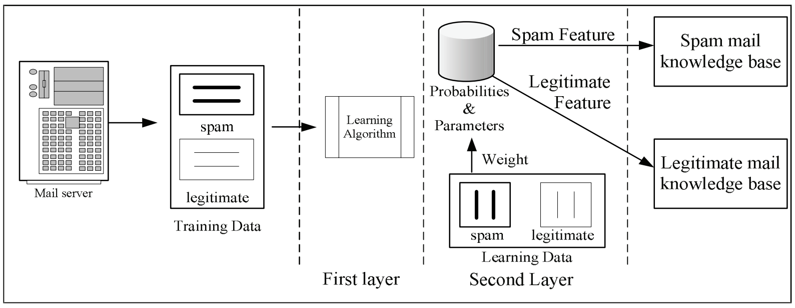

Fig.1. The individual learning model for spam filtering

Naïve Bayes Classifier. A Naïve Bayes classifier computes the likelihood that whether a mail is spam or not given the features that are contained in the mail [8]. The model, output by the Naïve Bayes algorithm, labels examples based on the features that they contain. We define *C* to be a random variable over the set of classes: legitimate and spam. That is, we want to compute *P(C/F)*, the probability that a mail is in a certain class given the mail contains the set of features F. We apply Bayes rule and express the probability as:

$$P(C/F) = \frac{P(F/C)*P(C)}{P(F)} \tag{1}$$

To use the Naïve Bayes rule we assume that the features occur independently from one another. If the features of a mail F include the features F_1, F_2, F_3 ...F_n, then equation (1) becomes:

$$P(C|F) = \frac{\prod_{i=j}^{n} P(F_i|C)*P(C)}{\prod_{j=1}^{n} P(F_j)} \tag{2}$$

Each *P(Fi /C)* is the frequency that features F_i occurs in a mail of class *C*. *P(C)* is the proportion of the class *C* in the entire set of mails. The output of the classifier is the highest probability class for a given set of strings. Since the denominator of equation (1) is the same for all classes we take the maximum class over all classes e of the probability of each class computed in equation (2) to get

$$\text{Most Likely Class} = \max_C \left(P(C) \prod_{i=1}^{n} P(F_i \mid C) \right) \tag{3}$$

Where, we use max$_C$ to denote the function that returns the class with the highest probability. Most Likely Class is the class in *C* with the highest probability and hence the most likely classification of the example with features *F*. Then we applied equation (3) to compute the most likely class for the mail.

Fisher's Probability Combination Method. Robinson proposed a Bayes-like method that can release the independent assumption through Fisher's method to combine probabilities [20]. For each word that appears in the training data, we calculate:

$$b(w) = \frac{\text{number of spam containing the word } w}{\text{total number of spam}} \tag{4}$$

$$g(w) = \frac{\text{number of legistimate mail containing the word } w}{\text{total number of legistimate mail}} \tag{5}$$

$$p(w) = \frac{b(w)}{b(w) + g(w)} \tag{6}$$

p(w) can be interpreted as the probability that randomly chosen an email that containing word "*w*" will be spam. There is a problem with the probabilities calculated as above when some words are very rare in the training set. For instance, if a word appears in exactly one email and is a spam, the value of *p(w)* is 1.0.

The Fisher's probability combination approach lets us combine our general background information with the data we have collected for a word in such a way that both aspects are given their proper importance. In this way, we determine an appropriate degree of belief about whether, when we see the word again, it will be in a spam. We calculate this degree of belief, *f(w)*, as follows:

$$f(w) = \frac{(s \times x) + (n \times p(w))}{s + n} \tag{7}$$

s: the strength we want to give to our background information

x: our assumed probability, based on our general background information, that a word we don't have any other experience of will first appear in a spam

n: the number of emails we have received that contain word

In practice, the values for *s* and *x* are found through testing to optimize performance. Reasonable starting points are 1 for s and 0.5 for *x*.

In the proposed method, first, we should calculate $(\ \ 2)\ln(p_1\ \ \ p_2\ \ \dots\ \ p_n)$. Then, consider the result to have a Chi-square with 2n degrees of freedom, and use Chi-square Table to compute the probability. The "spammness" probability of a mail that contains specific w is:

$$H = C^{-1}[-2\ln \prod_{w} f(w), 2n] \tag{8}$$

where

H: the "spammness" probability of a mail

C^{-1}: the inverse Chi-square function, used to derive a p-value from a Chi-square distributed random variable

Chi-square by Degree of Freedom. O'Brein and Vogel suggested using an authorship identification technique known as Chi-square by degree of freedom method for spam filtering [16]. This idea is based on Pearson's Chi-square statistic. They argued that as over 90% world spam was the work of just 140 spammers [14], Methods should be devised to identify the "textual fingerprints" of these spammers [17]. Baayen et al. note that authors may have textual fingerprints on texts they produced [2]. At least writers who are not consciously charging their style of writing across texts will leave their fingerprints in the text. If this is the case, we could use authorship identification methods to identify these textual fingerprints and eliminate a large proportion of spam.

The Chi-square test is a non-parametric test of statistical significance. In order to carry out a Chi-square analysis, the sample must be randomly drawn from the population. Also the data must be frequencies as opposed to percentages. The measured variables must be independent and finally the frequencies must not less than 5 are disregarded [17]. The Chi-square statistic can be calculated by

$$\chi^2 = \sum \frac{(O_i - E_i)^2}{E_i} \tag{9}$$

where

χ^2: The Chi-square value of an incoming mail

O_i: The observed value of an incoming mail

E_i: The expected value, calculated from the training set

3.2 Collaborative learning method

The proposed Collaborative Learning Model is based on three data mining algorithms. Each method contributed to the overall spam detection work through learning and collaboration. The whole collaborative filtering algorithms use the following steps to make recommendations to a user.

Construct posterior probabilities sets. We use training data to construct the posterior probabilities and parameters for each individual learning algorithm. According to the methods used, there are different posterior probabilities and parameters, in our case, spam and legitimate mail:

$$Pm_k c_x \mid C \leftarrow \{spam, legitimate\} \tag{10}$$

Where C are the set of classes and m are the set of learning algorithm.

Normalize the inconsistent scales. In order to collaborate to the output generated from the method we discussed previously, it is important to normalize the ratings of different scales to the same. We use the Gaussian Normalization Method for the collaborative work [9].

$$\hat{R}(x) = \frac{R_y(x) - \bar{R}_y}{\sqrt{\sum_x (R_y(x) - \bar{R}_y)^2}} \tag{11}$$

Where $\hat{R}(x)$ is the normalized value, $R_y(x)$ is the output value of each method, and $\bar{R}_y$ is the average output value of each method, derived from the weighting set.

Determining Weighting Value. Using learning data to determine weight value based on the result of individual learning. In each method, if the individual classifier makes right judgment to spam. It will be rewarded; otherwise, it will be punished. Because of the cost among the right and wrong decisions bring different cost to user. We give the reward value twice than the punishment. The learning weight generated from the subtraction of reward value and punishments, as shown in equation (12).

$$W_{mk} = (2 * va) - vp \tag{12}$$

Combine weights and probabilities to take shape collaborative learning model. The collaborative learning model is constructed by combining the posterior probabilities sets and weight values from individual learning. Weight each probability of individual learning algorithm and use collaborative learning model to calculate the predicted rating in testing data. Individual learning process and use collaborative learning model to derive the predicted rating to incoming mail.

$$Max(Pc_x = \sum_{g=1}^{k} Pm_g c_x * W_{mg} \mid C \leftarrow \{spam, legitimate\}) \tag{13}$$

The predicted rating $Pm_g c_x$ and the learning weight W_{mg} will depend on the collaborative learning algorithm. If the predicted rating of one class is higher than other classes, the system will recommend this class to the result.

We designed a simple collaborative learning algorithm as illustrated in Fig. 2, and the notations of the algorithms are shown in Table 1:

Table 1. Notations of the collaborative learning algorithm

T // Training data, L // Learning data, M // Individual learning algorithm
W // Weight, A // Attributes, C // Classes
va // Reward weight values, vp // Punishment weight values
$Pm_k c_x$ // the probability output by the individual learning process
W_{mk} // the weight derive from the learning data with the individual learning process
Pc_x // the summation of $Pm_k c_x \times W_{mk}$

```
Algorithm：A classifiable algorithm based on collaborative and learning
Input：$T, L, M$
Output：Collaborative learning model
Method：
$C \leftarrow \{c_1, c_2 ... c_x\}$, $T \leftarrow \{t_1, t_2 .. t_i\}$, $A \leftarrow \{a_1, a_2 ... a_j\}$, $M \leftarrow \{m_1, m_2 ..... m_k\}$, $L \leftarrow \{l_1, l_2 .. l_n\}$
//Create posterior probability for every method
    For each $m_k \in M$ do
        Gain $Pm_k c_x$
//Computing probability of each attributes in C
      For each $a_j \in A$ do
          For each $t_i \in T$ do
              IF $t_i \in c_x$ then
                  $s = s + 1$
```

```
                    $P(C_x) = s / c_x$
                  End if
              Next $t$
          Next $a$
        Next $m$
// Learning weight
        For each $m_k \in M$ do
          Gain $W_{mk}$
              For $l_n \in L$ do
//Gain the filter result for each individual algorithm to compare probability ( $va$ , $vp$ )
                    $Max = (Pm_k c_x \mid C \leftarrow \{c_1, c_2 ... c_x\})$
                IF the result of individual algorithm is same as the result of the class $C_x$
                    $W_{mk} = (2 * va) + vp$
                Else IF result of individual algorithm is opposite as the result of the class $C_x$
                    $W_{mk} = (2 * va) - vp$
                End if
              Next $w$
          Next $m$
// Collaborative model
                    $Max(Pc_x = \sum_{g=1}^{k} Pm_g c_x * W_{mg} \mid C \leftarrow \{c_1, c_2 ... c_x\})$
```

Fig.2. The proposed collaborative learning algorithm

4 Experiment and Results

In this section, we will use the Spam Email Database from the UCI Machine Learning Repository to examine our proposed collaborative method.

4.1 Data set

The Spam Email Database was created by Hewlett-Packard Labs [22]. It had been used for the HP internal-only technical report and other spam detection studies. The database contains 4601 instances and 58 attributes. We estimate our results over new data by using cross validation. Cross validation is the standard method to estimate the likely predictions over unseen data in measuring the result of data mining or machine learning Mining [13]. We randomly choose 50% instances for the algorithm training, 25% for the weight generating, and the remaining partition is then used to evaluate our method. Then we repeated the process leaving out a different partition for testing each time. This gave us a very reliable measure about our method's accuracy over unseen data. The data set and its usage in our study is summarized in Fig. 3:

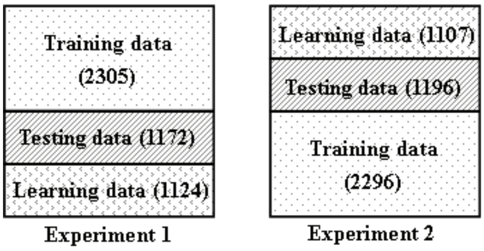

Fig.3. Cross validation of experiment 1 and 2

4.2 Results and Analysis

In order to evaluate the performance of three machine learning methods and collaborative learning that we proposed in spam filtering. We were interested in several quantities typically used in measuring the query result of information retrieval. These are:
True Positives (TP). The number of spam mail classified as spam.
True Negatives (TN). The number of legitimate mail classified as legitimate.
False Positives (FP). The number of legitimate mails falsely classified as spam.
False Negatives (FN). The number of spam mails falsely classified as legitimate.

The Detection Rate is defined as TP / (TP +FN), False Positive Rate as FP / (TN +FP), and Overall Accuracy as (TP +FN) / (TP +FP + FN + TN). The results of all individual learning methods and collaborative learning method are presented in Table 2 and Table 3. The voting scheme of three individual learning methods is also included for comparison.

Table 2. Results of the experiment (1)

	TP	TN	FP	FN	Detection Rate	False Positive Rate	Overall Accuracy
Naïve Bayes	365	679	52	76	82.77%	7.11%	75.43%
Fisher's	426	509	222	15	96.60%	30.37%	67.56%
Chi-square	404	628	103	37	91.61%	14.09%	74.57%
Voting	416	613	118	25	94.33%	16.14%	74.35%
Collaborative	422	646	85	19	95.69%	11.63%	77.17%

Table 3. Results of the experiment (2)

	TP	TN	FP	FN	Detection Rate	False Positive Rate	Overall Accuracy
Naïve Bayes	472	594	28	104	81.94%	4.50%	77.02%
Fisher's	568	457	165	8	98.61%	26.53%	74.06%
Chi-square	544	558	64	32	94.44%	10.29%	79.62%
Voting	557	543	79	19	96.70%	12.70%	79.48%
Collaborative	567	579	43	9	98.44%	6.91%	82.80%

As can be seen from Table 2 and 3, we can find that Fisher's method has the highest precision rate, but the recall and accuracy rates are not as good as others. And it suffers from the false positives rate. The Naïve Bayes has the lowest false positives rate and good in accuracy rate, though the detection rate is the lowest of the three. The Chi-square method generally has better performance than others. From the results of the experiment, the detection and false positive rate of collaborative learning scheme and voting scheme are down little less than those individual learning algorithms. But the accuracy rate of the collaborative learning is the best. Generally, collaborative learning has better performance than those individual learning algorithms or voting scheme.

5 Conclusion and Future Work

In order to deal with the huge amount of spam received day by day, powerful email filters with high reliability are needed. One problem of traditional text classification and signature-based methods is that they are based on a single method. In this paper, we introduced a collaborative learning scheme that can parallel filter spam with three different methods. In the proposed scheme, emails that are difficult to classify with a single one method, can be detected and filtered through the collaborative filtering architecture based on the collaborative weighting and learning. Thus, it has several advantages: first, the filtering system will not be affected by the failure of one method; second, the system can be deployed to P2P networks easily and provide higher reliability than centralized network; third, the training process is enhanced by the weight learning process, as we showed previously, the weight learning process provides a group learning mechanism for the three different filtering method and brings more accuracy result.

One of the most important areas of future work for this application is the development of more efficient algorithm. The current probabilistic method requires significant computing resources. Many incremental learning algorithms may solve this problem [6]. Another one is adding signature based function to our system. Although signature based methods have several weakness, but under the collaborative scheme, it helps building clusters for users' preferences.

Acknowledgement: The author would like to give thanks to the National Science Council of Taiwan for grant NSC 93-2213-E-224-038- to part of this research.

References

1. Androutsopoulos, I., Koutsias, J., Chandrinos, K. V., and Spyropoulos, C. D.: An Experimental Comparison of Naïve Bayesian and Keyword-Based Anti-Spam Filtering with Personal E-mail Messages. In Proc. of the 23 rd Annual International ACM SIGR Conference on Research and Development in Information Retrieval, Athens, Greece, (2000) 160–167

2. Baayen, H., Van Halteren, H., Neijt, A., and Tweedie, F.: An experiment in authorship attribution. In Journ´ees internationales d'Analyse statistique des Donn`ees Textuelles, (2002)
3. Cohen, William W.: Learning Rules that classify e-mail, the AAAI Spring Symposium on Machine Learning in Information Access. (1996) 18–25
4. Data Mining Cup 2003. Contest data, instructions and results (2003) available from: http://www.data-mining-cup.com/2003/Wettbewerb/1059704704/
5. Drucker, H., Wu, D., and Vapnik, V., N.: Support Vector Machines for Spam Categorization. IEEE Transaction on Neural Networks, Vol 10, No 5, (1999)
6. Giraud-Carrier, C.: Unifying Learning with Evolution through Baldwin an Evolution and Lamarckism: A Case Study. In: Proceedings of the Symposium on Computational Intelligence and Learning (CoIL-2000), MIT GmbH, June (2000) 36–41
7. Gomez Hidalgo, J. M.: Evaluating Cost-sensitive Unsolicited Bulk Email Categorization. In Proceedings of SAC-02, 17th ACM Symposium on Applied Computing, Madrid, ES, (2002) 615–620
8. Han, J. and Kamber, M.: Data mining concepts and techniques (USA, Morgan Kaufmann, 2001) page 284–287
9. Jin, R. and Si, L.: A study of methods for normalizing user ratings in collaborative filtering. Proceedings of the 27th annual international conference on Research and development in information retrieval, July (2004) 568–569
10. Joachims, T.: A probabilistic analysis of the Ricchio algorithm with TFIDF for text categorization. In Proc. of 14th International Conference on Machine Learning, (1997)
11. Jung, Jaeyeon, and Sit, Emil: An empirical study of spam traffic and the use of DNS black lists. Proceedings of the 4th ACM SIGCOMM conference on Internet measurement, October (2004) 370–375
12. Katirai, H.: Filtering junk e-mail: A performance comparison between genetic programming & naive bayes. (1999) available from: http://members.rogers.com/hoomank/papers/katirai99filtering.pdf
13. Kohavi, R.: A study of cross-validation and bootstrap for accuracy estimation and model selection. IJCAI, (1995)
14. Ludlow, M., Just 150 'spammers'blamed for e-mail woe. The Sunday Times, 1st December (2002) page 3
15. Metzger, J., Schillo, M. and Fischer, K.: A multiagent-based peer-to-peer network in java for distributed spam filtering. In Proc. of the CEEMAS, Czech Republic, June (2003)
16. O'Brien, C. and Vogel, C., Spam filters: Bayes vs. chi-squared; letters vs. words. In Proceedings of the International Symposium on Information and Communication Technologies, (2003)
17. O'Brien, C. and Vogel, C.: Comparing SpamAssassin with CBDF Email Filtering. In Proceedings of the 7th Annual CLUK Research Colloquium, (2004)
18. Provost., J.: Naive-bayes vs. rule-learning in classification of email. Technical Report AI-TR-99-284, University of Texas at Austin, Artificial Intelligence Lab, (1999)
19. Quinlan., J. R.: C4.5: Programs for Machine Learning. Morgan Kaufmann series in machine learning. Morgan Kaufmann, (1993)
20. Robinson, G.: A Statistical Approach to the Spam Problem. Linux Journal, March 2003. Linux Journal, Volume 2003, Issue 107, March (2003) Page 3
21. Schapire, R.E., and Singer, Y.: BoosTexter: A Boosting-based System for Text Categorization, Machine Learning. vol 39, (2000) 135–168
22. UCI Machine Learning Repository. Retrieved Jan 20, (2005) available from: http://www.ics.uci.edu/~mlearn/MLRepository.html