Automatic Personalized Spam Filtering through Significant Word Modeling

Khurum Nazir Junejo
Dept. of Computer Science
Lahore University of Management Sciences
Lahore, Pakistan
junejo@lums.edu.pk

Asim Karim
Dept. of Computer Science
Lahore University of Management Sciences
Lahore, Pakistan
akarim@lums.edu.pk

Abstract

Typically, spam filters are built on the assumption that the characteristics of e-mails in the training set is identical to those in individual users' inboxes on which it will be applied. This assumption is oftentimes incorrect leading to poor performance of the filter. A personalized spam filter is built by taking into account the characteristics of e-mails in individual users' inboxes. We present an automatic approach for personalized spam filtering that does not require users' feedback. The proposed algorithm builds a statistical model of significant spam and non-spam words from the labeled training set and then updates it in multiple passes over the unlabeled individual user's inbox. The personalization of the model leads to improved filtering performance. We evaluate our algorithm on two publicly available datasets. The results show that our algorithm is robust and scalable, and a viable solution to the server-side personalized spam filtering problem. Moreover, it outperforms published results on one dataset and its performance is equivalent to the others on the second dataset.

1. Introduction

E-mail is arguably the most widely used web application playing an essential role in the functioning of most businesses. Globalization has resulted in an exponential increase in the volume of e-mails. Unfortunately, a large chunk of it is in the form of spam or unsolicited e-mails. In 2006, over 80% of all e-mails sent were spam resulting in a loss of 75 billion dollars to organizations worldwide [1]. Spam messages not only waste users' time and money but are also harmful for their computer's security. Commtouch, a security service provider, reported 19 new e-mail borne viruses in the month of January 2006 [2].

E-mail users spend an increasing amount of time reading messages and deciding whether they are spam or non-spam and categorizing them into folders. Some e-mail clients require users to label their received messages for training local (or personalized server-based) spam filters. E-mail service providers would like to relieve users from this burden by installing server-based spam filters that can classify e-mails as spam automatically and accurately without user feedback.

Typically, server-based spam filters are trained on general training sets and then applied to individual users' inboxes. However, the characteristics of individual users' inboxes are usually not identical to that of the general e-mail corpus used for training the spam filter, resulting in poor filtering performance. Furthermore, the characteristics of spam e-mails evolve with time making non-adaptive filters less robust to change. Thus, there is a need for personalized spam filters that learn from general training sets and adapt to the characteristics of individual users' inboxes. This adaptation must be done without asking the users to label their e-mails. Earlier works on personalized spam filtering utilize users' input. This approach is clearly not convenient for the e-mail user.

In this paper, we present an automatic statistical approach for classifying individual users' e-mails without requiring their feedback. The algorithm learns a statistical model of significant words from the general corpus of labeled e-mails in a single pass over them and then updates this learned model in one or more passes over the individual user's unlabeled e-mails. This approach allows automatic specialization of the general model to the underlying distribution of e-mails in individual users' inboxes. The significant word model is built from the tokens in the e-mail content and their frequencies. We implement and test our algorithm on datasets available from [3], and compare its performance with other results published in the literature on the same datasets.

The rest of the paper is organized as follows. In section 2, we provide a brief review of content-based spam filters with specific focus on personalized spam filtering. Section 3 describes our algorithm for automatic personalized spam filtering. Section 4 presents the results of extensive experimental evaluations together with a comparison with other algorithms. We conclude in section 5.

1082-3409/07 $25.00 © 2007 IEEE
DOI 10.1109/ICTAI.2007.66

2. Personalized content-based spam filtering

Many approaches are used in practice to control the menace of spam including global and local blacklists, global and local whitelists, IP blocking, legislation, and content-based filtering. Content-based filters employ machine learning techniques to learn to predict spam e-mails given a corpus of training e-mails. Such filters are typically deployed on the mail server that filters e-mails for all users of the server. Researchers have developed content-based spam filters using Bayesian approaches [4–7], support vector machines (SVM) [8, 9], nearest neighbor classifiers [10], rule-based classifiers [11, 12], and case-based reasoning [13]. Among these techniques, Bayesian approaches and SVMs have shown consistently good performances. Sahami et al. present one of the earliest naïve Bayes classifier for the spam classification problem [4]. Since then, numerous variations of the naïve Bayes classifier have been presented for spam filtering [5–7]. The popular Mozilla's e-mail client implements a naïve Bayes classifier for spam filtering [6]. Support vector machine (SVM) is a powerful supervised learning paradigm based on the structural risk minimization principle from computational learning theory. SVMs exhibit good generalization capabilities and have shown good spam classification performance. One of the first SVM for the spam classification problem is presented in [8]. Since then, several extensions and variations have been presented such as [9].

The majority of the supervised machine learning techniques presented for spam filtering assumes that e-mails are drawn independently from a given distribution. That is, the statistical distribution of e-mails in the training dataset is identical to that of the individual user's e-mails on which the trained filter will be applied. This assumption, however, is usually incorrect in practice; the training dataset is typically derived from multiple Internet sources reflecting different distributions of spam and non-spam e-mails that are different from that of the individual user's e-mails. A personalized spam filter is capable of adapting to the distribution of e-mails of each individual user. Previous works on personalized spam filtering have relied upon user feedback in the form of e-mail labels from each individual user [14, 15]. This strategy burdens the e-mail user with the additional task of aiding the adaptation of the spam filter.

Recently, with the availability of appropriate datasets [3], several automatic personalized spam filtering approaches have been presented [16–19]. These works explore various supervised, semi-supervised, and unsupervised techniques. We present a statistical approach for automatic personalized spam filtering that does not require users' feedback. The approach is based on a significant word model of spam and non-spam e-mails similar to that developed in Bayesian approaches. However, unlike many Bayesian approaches presented in the literature, we specialize the model to reflect the distributions of e-mails in individual users' inboxes. A comparison of our algorithm with [16–19] is given in Section 4.

3. Our algorithm

Our personalized spam filtering algorithm consists of two phases of processing. In the first phase, called the training phase, the algorithm learns a statistical model of spam and non-spam words from the training set in a single pass over the training set. The second phase, called the specialization phase, consists of two or more passes over the user's inbox. In the first pass, the statistical model developed in the training phase is used to label the e-mails in the individual user's inbox, and to build an updated statistical model of the e-mails. This can be done multiple times. In the last pass, the updated statistical model is used to score and classify the e-mails in the individual user's inbox. The pseudo-code of our algorithm is given in Figure 1.

The statistical model is developed as follows: For each distinct word in the labeled (i.e. training or initial passes of evaluation) set, determine its estimated probability in spam and non-spam e-mails. Then, find the difference of these two values for each word. Now choose the significant words by selecting only those words for which the difference between their spam and non-spam probabilities is greater than some threshold *t*. This approach also categorizes the significant words as either a spam word or a non-spam word. Each spam and non-spam word is assigned a weight based on the ratio of its probability in the spam and non-spam e-mails. This statistical model of words can then be used to classify a given e-mail by computing its spam score and non-spam score values, where the spam score (non-spam score) of an e-mail is the weighted sum of the words of that e-mail that belong to the significant spam (non-spam) words set. If the spam score multiplied by a scaling factor (*s*) is greater than the non-spam score then the e-mail is labeled as spam; otherwise, it is labeled as non-spam. This statistical model is developed in the training phase as well as in the initial passes of the specialization phase. In the final pass of the specialization phase, the final scores and classifications of e-mails are output.

The motivation for using significant words that have differences of their probabilities in spam and non-spam e-mails greater than a specified threshold is: (1) a word that occurs much more frequently in spam e-mails (or non-

```
N = Total number of e-mails
$N_S$ = Number of spam e-mails
$N_N$ = Number of non-spam e-mails
D = number of words in dictionary (indexed from 1 to D)
$C_{Si}$ = count of word i in all spam e-mails
$C_{Ni}$ = count of word i in all non-spam e-mails
$Z_S$ = set of significant spam words
$Z_N$ = set of significant non-spam words
t = threshold
s = scale factor
$W_{Si}$ = weight associated with significant spam word i
$W_{Ni}$ = weight associated with significant non-spam word i
$T_i$ = word i

Training Phase (Phase 1 on Training Set)
Build_Significant_Word_Model Procedure
-For each distinct word i in dataset find $C_{Si}$ and $C_{Ni}$
-Find the significant spam words $Z_S$ such that for each word $T_i$ in $Z_S$, $C_{Si}/N_S – C_{Ni}/N_N > t$
-Find the significant non-spam words $Z_N$ such that for each word $T_i$ in $Z_{N,}$ $C_{Ni}/N_N – C_{Si}/N_S > t$
-For each significant spam and non-spam word find their weight as follows:
        -$W_{Si =}$ $[C_{Si} / C_{Ni}] * [N_N / N_S]$, for all words in $Z_S$
        -$W_{Ni =}$ $[C_{Ni} / C_{Si}] * [N_S / N_N]$, for all in $Z_N$

Specialization Phase (Phase 2 on Evaluation Set)
Initial Passes
Score_Emails Procedure
-For each e-mail in the evaluation dataset
    -spam_score = $\sum W_{Si}$ (sum is over all significant spam words in e-mail)
    -nonspam_score = $\sum W_{Ni}$ (sum is over all significant non-spam words in e-mail)
    -If (s * spam_score > nonspam_score) then classify as spam; other wise classify as non-spam
    - output s*spam_score – nonspam_score

    -Build statistical model concurrently with scoring e-mails (using Build_Significant_Word_Model given above)

Last Pass
    -Score and classify e-mails using the updated statistical model (procedure is identical to the Score_Emails procedure given above)
```

Figure 1. Our automatic personalized spam filtering algorithm

spam e-mails) will be a better feature in distinguishing spam and non-spam e-mails than a word that occurs frequently in the dataset but its occurrence within spam and non-spam e-mails is almost similar, and (2) this approach greatly reduces the number of words that are of interest, simplifying the model and its computation. It is worth noting that this approach of significant word selection is related to the information theoretic measure of information gain. The scale factor is used to cater for the fact that the number of non-spam words, and their weighted sum in a given e-mail, is usually greater than the number of spam words and their weighted sum.

Table 1. Results for dataset A (all values in %)

	First Pass	Second Pass	Optimal
Inboxes	AUC	AUC	AUC
Eval-00	96.35	98.60	99.99
Eval-01	97.37	98.78	99.96
Eval-02	94.59	99.43	99.91
Avg.	96.10	98.94	99.95

The purpose of the weighting scheme for the significant words is to give an advantage to words for which either the spam probability or the non-spam probability is proportionally greater than the other. For example, if the word with ID '10' has spam and non-spam counts of 0 and 50, respectively, and the word with ID '11' has spam and non-spam counts of 950 and 1000, respectively, then even though their difference in counts is the same (50) the word with ID '10' gives more information regarding the classification of the e-mail than word with ID '11'.

The specialization phase adapts the general statistical model to the characteristics of the individual user's inbox. The model developed from the training phase is used for the initial classification of the user's e-mails. Subsequently, the statistical model is updated to incorporate the characteristics of the user's inbox. This updated model is then used to finally score and classify the e-mails in the user's inbox.

4. Experimental evaluation

In this section, we present the results of our experimental evaluation of the automatic personalized spam filtering algorithm. The algorithm is implemented in Java. The code uses special built-in data structures of Java such as hash maps that provide an efficient way of retrieving word objects by avoiding searching through an array list of word IDs.

4.1 Datasets

We use the datasets available from the ECML-PKDD Discovery Challenge website [3]. Dataset A contains a training set and 3 evaluation sets (users' inboxes). The training set contains 4000 e-mails collected from various sources, while the evaluation sets contain 2500 e-mails. These evaluation datasets are identified as Eval-00, Eval-01, and Eval-02. Dataset B contains a training set of 100 e-mails and 15 evaluation sets of 400 e-mails each. These

Table 2. Results for dataset B (all values in %)

	First Pass	Second Pass	Third Pass	Fourth Pass	Optimal
Inboxes	AUC	AUC	AUC	AUC	AUC
Eval-00	72.16	94.15	96.89	96.72	99.93
Eval-01	73.52	96.50	96.98	96.61	99.99
Eval-02	91.24	96.29	96.72	96.75	99.99
Eval-03	98.14	99.22	99.12	99.12	99.98
Eval-04	82.11	93.79	94.70	95.05	99.66
Eval-05	80.71	78.65	74.11	69.90	99.96
Eval-06	72.42	92.72	91.81	90.79	100
Eval-07	86.78	95.46	95.96	96.16	99.70
Eval-08	79.62	99.32	99.39	99.24	100
Eval-09	75.20	98.12	99.19	98.05	100
Eval-10	85.82	94.08	95.88	96.24	99.97
Eval-11	86.69	91.26	92.54	92.36	99.61
Eval-12	91.28	98.85	99.60	99.51	99.88
Eval-13	83.12	88.21	90.23	90.74	99.84
Eval-14	75.49	90.52	95.50	97.92	99.92
Avg.	82.29	93.81	94.57	94.38	99.89

evaluation sets are identified as Eval-00 to Eval-14. The ratio of spam and non-spam e-mails in all the datasets is 50-50. The distribution of e-mails in the training sets which is a combined source of training data is different from the distributions of the e-mails received by individual users.

Each e-mail in the datasets is represented by a word (term) frequency vector. Each word in an e-mail is identified by an ID and its frequency count in the e-mail. An additional attribute identifies the label of the e-mail as either spam or non-spam.

4.2 Results

We report the performance of our algorithm using the AUC metric. The area under the receiver operating characteristics curve (ROC) has emerged as a robust criterion for evaluating classifier performance [20]. It measures the area under the false positive versus true positive curve that is obtained by sweeping through all possible thresholds of the classifier's outputs. A value of 1 (or 100%) is the highest possible AUC value. The results in this section are generated for the case when t = 0. That is, the significant word model contains all words in the dictionary.

The performance of our algorithm on the evaluation sets (users' inboxes) of dataset A is given in Table 1. After building the statistical model and finding the best scale factor s on the training set, the users' inboxes are labeled in the first and second pass over them. It is observed that the performance improves significantly in the second pass. This reflects the adaptation of the statistical model to the distribution of e-mails in individual users' inboxes that is done during the first pass. Note that the scale factor is not updated after learning over the training set. The last column of Table 1 shows the 'optimal' performance obtained when the labels of the users' inboxes are known. Our algorithm, which does not rely on the labels, is capable of approaching the optimal values in just one efficient pass over the users' inboxes.

Table 2 shows the results of our algorithm for dataset B. Dataset B represents a much more challenging problem in which the number of e-mails in the training and evaluation sets is only 100 and 400, respectively. We conduct multiple passes over the users' inboxes. The average AUC value increases by more than 10% from the first pass, and by about 2% from the second pass. There is a slight decrease in average AUC value in the fourth pass. This performance is remarkable considering that the training and evaluation sets are so small. Moreover, the average AUC value is negatively affected by one user inbox (Eval-05) whose AUC value actually decreases with number of passes. This is probably due to a markedly different distribution of spam and non-spam words in comparison to that in the raining set.

Table 3. Comparison with other techniques

Technique	Dataset A	Dataset B
Our algorithm	98.94	94.57
Junejo et al. [16]	98.75	---
Kyriakopoulou [17]	97.31	95.08
Cormack [18]	93.00	94.90
Cheng and Li [19]	93.33	---

4.3 Comparison

We compare our algorithm's performance with four recently published results on the same datasets in Table 3. Three of these results [16, 17, 18] are winning performances of the Discovery Challenge [3]. Our algorithm outperforms all algorithms on dataset A and is on par with the others on dataset B. Junejo et al. has the previous best performance on dataset A (they do not report results on dataset B) [16]. Our algorithm improves on their algorithm by using estimated probabilities rather than occurrence counts for defining the significant word model. Kyriakopoulou and Kalamboukis preprocess the dataset by clustering the training set with individual evaluation sets. The combined set is augmented with additional meta-features derived from the clustering. This combined set is then learned using a transductive SVM. This approach is computationally expensive and non-adaptive. Cormack use statistical compression models for predicting spam and non-spam e-mails [18]. His approach is adaptive but the reported performances lag the leaders. Cheng and Li present a semi-supervised classifier ensemble approach for the personalized spam filtering problem [19]. Their approach is also computationally expensive as compared to ours, in addition to lagging in performance by more than 3%.

4.4 Generalization performance

The results presented in the previous subsections assume a transductive learning problem where all the unlabeled e-mails in users' inboxes are classified. However, in practice, once a personalized spam filter is learned using labeled and unlabeled e-mails it is applied to unseen e-mails. The performance over these unseen e-mails represents the generalization performance of the filter. We evaluate the generation performance of our algorithm by splitting the evaluation sets into two: split 1 is used during learning and split 2 contain the unseen e-mails. The generalation performance of our algorithm on dataset A is shown in Figure 2. In general, the average AUC value over split 2 (the unseen e-mails) is less than that over split 1. However, this difference is typically less than 1%. Furthermore, the decrease in average AUC value with increase in size of split 2 (decrease in size of split 1) is graceful. This result demonstrates the robustness of our algorithm.

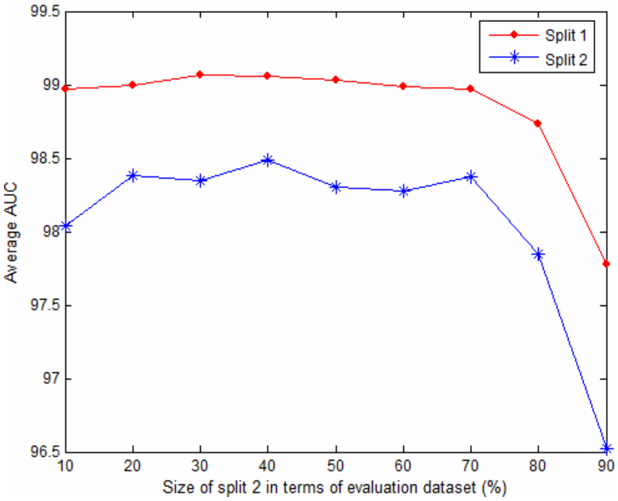

Figure 2. Average AUC vs. split size for dataset A

4.5 Effect of threshold *t*

The threshold *t* controls the size of the significant word model. We explore the effect of *t* on the size of the significant word model and the performance of our algorithm in Figures 3 and 4, respectively. Figure 3 shows that the size of the significant word set for each user inbox in dataset A decreases exponentially with increase in *t*. With *t* = 0.24, the number of significant words is less than 50 for each user inbox. However, even with this parsimonious model, our algorithm outperforms the previous second best algorithm (Figure 4, Table 3). This result demonstrates the robustness and scalability of our algorithm. In particular, our algorithm can be scaled cost-effectively to thousands of users served by an e-mail service provider.

4.6 Additional observations

The only parameter of our algorithm is the scale factor *s*. The scale factor is learned from the training set and then used for all users' inboxes. We select the *s* that maximizes the accuracy of classification. The adaptation of the algorithm occurs through the update of the significant word model.

We build the statistical model by considering word occurrence rather than frequency. That is, if a word occurs more than once in an e-mail we count it as one irrespective of how many times it occurs in the e-mail. We have compared the performance of our algorithm

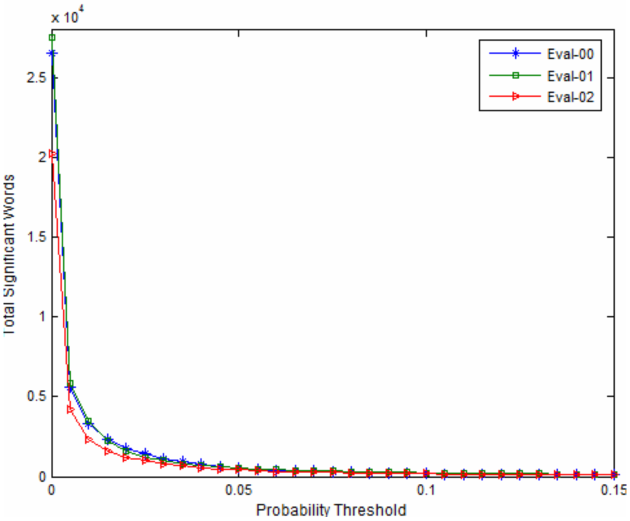

Figure 3. Number of significant words vs. threshold (dataset A)

with the algorithm based on word frequencies. We find that building a significant word model using frequencies decreases performance slightly. Thus, the simple word occurrence approach is more accurate and computationally efficient.

5. Conclusion

We present a robust and scalable algorithm for automatic personalized spam filtering. The algorithm uses an adaptable significant word model to capture the differing distributions of e-mails in users' inboxes. This model is first built from the training set consisting of labeled e-mails. Subsequently, it is adapted to the unlabeled e-mails in individual users' inboxes. This adaptation is done in one or more passes over the users' inboxes. We perform extensive empirical evaluation of our algorithm reporting performance and scalability results. Our algorithm outperforms all published results for one dataset and is on par on the second. Moreover, our algorithm is capable of acceptable performance with a small memory footprint required for each user. As such, the algorithm is a viable solution to the server-based personalized spam filtering problem.

6. Acknowledgment

The first author gratefully acknowledges scholarship support from Lahore University of Management Sciences and Higher Education Commission (HEC) of Pakistan.

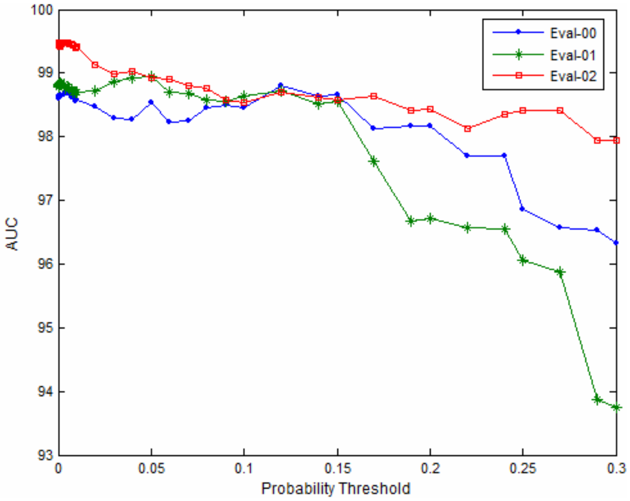

Figure 4. Number of significant words vs. average AUC (dataset A)

7. References

[1] N. Leavitt. Vendor's fight spam's sudden rise. *IEEE Computer*, 16–19, March 2007.

[2] Commtouch. January virus and spam statistics: 2006 starts with a bang. *Commtouch Press Release*, http://www.commtouch.com/Site/News_Events/pr_content.asp?news_id=602&cat_id=1, 2006.

[3] S. Bickel. Discovery challenge. http://www.ecmlpkdd2006.org/challenge.html, 2006.

[4] M. Sahami, S. Dumais, D. Heckerman, and E. Horvitz. A Bayesian approach to filtering junk e-mail. *Proc. of AAAI Workshop on Learning for Text Categorization*, AAAI Technical Report WS-98-05, 1998.

[5] P. Graham. Better Bayesian filtering. *Proc. of 2003 Spam Conference*, http://www.paulgraham.com/better.html, 2003.

[6] E. Michelakis, I. Androutsopoulos, G. Paliouras, G. Sakkis, and P. Stamatopoulos. Filtron: a learning-based anti-spam filter. *Proc. 1st Conf. on Email and Anti-Spam (CEAS 2004)*, 2004.

[7] A.K. Seewald. An evaluation of naive Bayes variants in content-based learning for spam filtering. Kluwer Academic Publsihing, 2005.

[8] H. Drucker, D. Wu, and V.N. Vapnik. Support vector machine for spam categorization. *IEEE Transactions on Neural Networks*, 10(5), 1048–1054, 1999.

[9] A. Kolcz and J. Alspector. SVM-based filtering of e-mail spam with content-specific misclassification costs. *Proc. of the TextDM Workshop on Text Mining*, 2001.

[10] G. Sakkis, I. Androutsopoulos, G. Paliouras, V. Karkaletsis, C. D. Spyropoulos, and P. Stamatopoulos. A memory-based approach to anti-spam filtering for mailing lists. *Information Retrieval*, Springer, 6(1), 49–73, 2003.

[11] W.W. Cohen. Learning rules that classify e-mail. *Proc. of 1996 AAAI Spring Symposium in Information Access*, 1996.

[12] E.C.J. Kay and E. McCreath. Automatic induction of rules for e-mail classification. *Proc. of the Sixth Australasian Document Computing Symposium*, 13–20, 2001.

[13] S.J. Delany, P. Cunningham, and L. Coyle. An assessment of case-based reasoning for spam filtering. *Artificial Intelligence Review*, Springer, 24(3-4), 359–378, 2005.

[14] R. Segal. J. Crawford, J. Kephart, and B. Leiba. SpamGuru: an enterprise anti-spam filtering system. *Proc. of Conference on Email and Anti-Spam (CEAS '04)*, 2004.

[15] A. Gray and M. Haahr. Personalised collaborative spam filtering. *Proc. of Conference on Email and Anti-Spam (CEAS '04)*, 2004.

[16] K.N. Junejo, M.M. Yousaf, and A. Karim. A two-pass statistical approach for automatic personalized spam filtering. *ECML-PKDD Discovery Challenge Workshop*, 2006.

[17] A. Kyriakopoulou and T. Kalamboukis. Text classification using clustering. *ECML-PKDD Discovery Challenge Workshop*, 2006.

[18] G.V. Cormack. Harnessing unlabeled examples through application of dynamic Markov modeling. *ECML-PKDD Discovery Challenge Workshop*, 2006.

[19] V. Cheng and C.H. Li. Personalized spam filtering with semi-supervised classifier ensemble. *Proc. of International Conf. on Web Intelligence (WI '06)*, 2006.

[20] C. Cortes and M. Mohri: AUC optimization vs. error rate minimization. Advances in Neural Information Processing Systems, *NIPS*, 2004.