On Some Feature Selection Strategies for Spam Filter Design

Ren Wang[1], Amr M. Youssef[2], Ahmed K. Elhakeem[1]

Department of Electrical and Computer Engineering[1]
Concordia Institute for Information Systems Engineering[2]
Concordia University, Montreal, Quebec, Canada
re_wan@encs.concordia.ca, youssef@ciise.concordia.ca, ahmed@ece.concordia.ca

Abstract - Feature selection is an important research problem in different statistical learning problems including text categorization applications such as spam email classification. In designing spam filters, we often represent the email by vector space model (VSM), i.e., every email is considered as a vector of word terms. Since there are many different terms in the email, and not all classifiers can handle such a high dimension, only the most powerful discriminatory terms should be used. Another reason is that some of these features may not be influential and might carry redundant information which may confuse the classifier. Thus, feature selection, and hence dimensionality reduction, is a crucial step to get the best out of the constructed features.

There are many feature selection strategies that can be applied to produce the resulting feature set. In this paper, we investigate the use of Hill Climbing, Simulated Annealing, and Threshold Accepting optimization techniques as feature selection algorithms. We also compare the performance of the above three techniques with the Linear Discriminate Analysis. Our experiment results show that all these techniques can be used not only to reduce the dimensions of the e-mail, but also improve the performance of the classification filter. Among all the strategies, simulated annealing has the best performance which reaches a classification accuracy of 95.5%.

Keywords: *Spam email filter, feature selection, Hill Climbing, Simulated Annealing, Threshold Accepting, Linear Discriminate Analysis*

1 Introduction

The junk email problem is rapidly becoming unmanageable, and threatens to destroy email as a useful means of communication. These tide of unsolicited emails flood into corporate and consumer inboxes everyday. Most spam is commercial advertising, often for dubious products, get-rich-quick schemes, or quasi-legal services. People waste increasing amounts of their time reading/deleting junk emails. According to a recent European Union study, junk email costs all of us some 9.4 billion (US) dollars per year, and many major ISPs say that spam adds about 20% to the cost of their service. There is also the fear that such emails could hide viruses which can then infect the whole network. Future mailing system should require more capable filters to help us in the selection of what to read and avoid us to spend more time on processing incoming messages.

Many commercial and open-source products exist to accommodate the growing need for spam classifiers, and a variety of techniques have been developed and applied toward the problem, both at the network and user levels. The simplest and most common approaches are to use filters that screen messages based upon the presence of words or phrases common to junk e-mail. Other simplistic approaches include black-listing (i.e., automatic rejection of messages received from the addresses of known spammers) and white-listing (i.e., automatic acceptance of message received from known and trusted correspondents). In practice, effective spam filtering uses a combination of these three techniques. In this paper, we only discuss how to classify the junk emails and legitimate emails based on the words or features.

From the machine learning view point, spam filtering based on the textual content of email can be viewed as a special case of text categorization, with the categories being spam or non-spam. In text categorization [5], the text can be represented by vector space model (VSM). Each email can be transferred into the vector space model. This means every email is considered as a vector of word terms. Since there are many different words in the email and not all classifiers can handle such a high dimension, we should choose only the most powerful discriminatory terms from the email terms. Another reason of applying feature selection is that the reduction of feature space dimension may improve the classifiers' prediction accuracy by alleviating the data sparseness problem.

In this paper, we investigate the use of Hill Climbing (HC), Simulated Annealing (SA), and Threshold Accepting (TA) local search optimization techniques [8] as feature selection algorithms. We also compare the performance of the above three techniques with Linear Discriminate Analysis (LDA) [3].

Our results indicate that, using a K-Nearest Neighbor (KNN) classifier [1], the accuracy of spam filters using any of the above strategies outperform those obtained without feature selection. Among the four approaches, SA reaches the best performance.

The rest of the paper is organized as follows. Section 2 introduce the experimental settings and related feature selection strategies. In section 3, we report the experimental results obtained and finally section 4 is the conclusion.

2 Experimental Settings

In our experiments, we first transfer the emails into vectors by TF-IDF formulas [5]. Then we apply the proposed feature

selection techniques to choose the best discriminating feature sets. Finally we compare the accuracy obtained with the four strategies.

2.1 Data Sets

Unlike general text categorization tasks where many standard benchmark collections exist, it is very hard to collect legitimate e-mails for the obvious reason of protecting personal privacy. In our experiment we use Pu1Corpus [10]. This corpus consists of 1099 messages; 481 of which are marked as spam and 618 are labeled as legitimate, with a spam rate of 43.77%. The messages in PU1 corpus have header fields and HTML tags removed, leaving only subject line and mail body text. To address privacy, each token was mapped to a unique integer. The corpus comes in four versions: with or without stemming and with or without stop word removal. In our experiment we use Lemmatize enabled, stop-list enabled version from PU1 corpus. This corpus has already been parsed and tokenized into individual words with binary attachments and HTML tags removed. We randomly chose 62 legitimated e-mails and 48 spam e-mails for testing and the rest e-mails for training.

2.2 Classifiers

K-nearest neighbor classification is an instance-based learning algorithm that has shown to be very effective in text classification. The success of this algorithm is due to the availability of effective similarity measure among the *K* nearest neighbor.

The algorithm starts by calculating the similarity between the test e-mail and all e-mail in training set. It then picks the *K* closet instances and assigns the test e-mail to the most common class among these nearest neighbors.

Thus after transforming the training e-mails and test e-mails into vectors, the second step is to find out the *K* vectors from training vector which are most similar to the test vector. In this work, we used the Euclidian distance as a measure for the similarity between vectors.

2.3 Transfer E-mails to Vectors

In text categorization, the text can be represented by vector space model. For terms appearing frequently in many e-mails has limited discrimination power, we use Term frequency and Inverse document frequency (TF-IDF) representation to represent the e-mail [5]. Accordingly, the more often a term appears in the e-mail, the more important this word is for that e-mail. In our experiment ,we sorted the features by its document frequency (DF), i.e., the number of e-mails that contain the i^{th} features to choose 100 features which DF range is 0.02 to 0.5 [2]. Thus the input to the feature selection algorithm is a feature vector of length 100. We then applied the feature selection algorithms to find out the most powerful discriminatory terms from the 100 features and test the performance of the e-mail filter.

2.4 Performance Measures

We now introduce the performance measures used in this paper. Let $N=A+B+C+D$ be the total number of test e-mails in our corpus.

Table 1 Confusion Matrix

	Spam	Non-Spam
Filter Decision: Spam	A	B
Filter Decision: Non-Spam	C	D

If table 1 denotes the confusion matrix of the e-mail classifier, then we define the accuracy, precision, recall, and F1 for spam e-mails as follows:

$$ACCURACY = \frac{A+D}{N},\ PRECISION\ (P) = \frac{A}{A+B}$$

$$RECALL\ (R) = \frac{A}{A+C},\ F1 = \frac{2PR}{P+R}.$$

Similar measures can be defined for legitimate e-mails.

2.5 Feature Selection Strategies

The output of the vector space modeling (VSM) is a relatively long feature vector that may have some redundant and correlated features (curse of dimensionality). This is the main motivation for using the feature selection techniques. The proposed feature selection algorithms are classifier dependant. This means that different possible feature subsets are examined by the algorithm and the performance of a pre-specified classifier is tested for each subset and finally the best discriminatory feature subset is chosen by the algorithm. There are many feature selection strategies (FSS) that can be applied to produce the resulting feature set. In what follows, we describe and report results conducted with our proposed FSS.

2.5.1 Hill Climbing (HC)

The basic idea of HC is to choose a solution from the neighborhood of a given solution, which improves this solution best, and stops if the neighborhood does not contain an improving solution [8].

The hill climbing used in this paper can be summarized as follows:

1. Randomly create an initial solution S1. This solution corresponds to a binary vector of length equal to the total number of features in the feature set under consideration. The 1's positions denote the set of features selected by this particular individual. Set I*=S1 and calculate its corresponding accuracy Y(S1).

2. Generate a random neighboring solution S2 based on I* and calculate its corresponding accuracy Y(S2).
3. Compare the two accuracies. If the corresponding accuracy of the neighboring solution Y(S2) is higher than Y(S1), set I*=S2.
4. Repeat step 2 to 3 for a pre-specified number of iteration (or until a certain criterion is reached.)

Although hill climbing has been applied successfully to many optimization problems, it has one main drawback. Since only improving solutions are chosen from the neighborhood, the method stops if the first local optimum with respect to the given neighborhood has been reached. Generally, this solution is not globally optimal and no information is available on how much the quality of this solution differs from the global optimum. A first attempt to overcome the problem of getting stuck in a local optimum was to restart iterative improvement several times using different initial solutions (multiple restart). All the resulting solutions are still only locally optimal, but one can hope that the next local optimum found improves the best found solution so far. In our experiments, we used 10 different initial solutions.

2.5.2 Simulated Annealing (SA)

Kirkpatrick *et. al* [6] proposed SA, a local search technique inspired by the cooling processes of molten metals. It merges HC with the probabilistic acceptance of non-improving moves. Similar to HC, SA iteratively constructs a sequence of solutions where two consecutive solutions are neighbored. However, for SA the next solution does not necessarily have a better objective value than the current solution. This makes it possible to leave local optimum.

First, a solution is chosen from the current solution. Afterwards, depending on the difference between the objective values of the chosen and the current solution, it is decided whether we move to the chosen solution or stay with the current solution. If the chosen solution has a better objective value, we always move to this solution. Otherwise we move to this solution with a probability which depends on the difference between the two objective values. More precisely, if S1 denotes the current solution and S2 is the chosen solution, we move to S2 with probability:

$$p(s_1, s_2) = e^{-\max\{Y(s_1)-Y(s_2),0\}/T} \quad (1)$$

The parameter T is a positive control parameter (temperature) which decreases with increasing number of iterations and converges to 0. As the temperature is lowered, it becomes ever more difficult to accept worsening moves. Eventually, only improving moves are allowed and the process becomes 'frozen'. The algorithm terminates when the stopping criterion is met. [7]. Furthermore, the probability above has the property that large deteriorations of the objective function are accepted with lower probability than small deteriorations.
The simulated annealing used in this paper can be summarized as follows:

1. Randomly create an initial solution S1 .This solution corresponds to a binary vector of length equal to the total number of features in the feature set under consideration. The 1's positions denote the set of features selected by this particular individual. Set I*=S1 and calculate its corresponding accuracy Y(S1).
2. Create the parameter (temperature) T and constant cooling factor α, $0< \alpha<1$.
3. Generate a random neighboring solution S2 based on I* and calculate its corresponding accuracy.
4. Compare the two accuracy. If the corresponding accuracy of the neighboring solution Y (S2) is higher than Y(S1), set I*=S2. Otherwise, generate U = rand (0...l). Compare U and P(S1,S2).If U> P(S1,S2), I*=S2.
5. Decrease the temperature by T=T* α.
6. Repeat step 3 to 5 for a pre-specified number of iteration (or until a certain criterion is reached).

To compare with HC, we also set the same 10 initial solutions and record the best solutions.

2.5.3 Threshold Accepting (TA)

A variant of simulated annealing is the threshold accepting method. It was designed by Ducek and Scheuer [8] as a partially deterministic version of simulated annealing. The only difference between simulated annealing and threshold accepting is the mechanism of accepting the neighboring solution. Where simulated annealing uses a stochastic model, threshold accepting uses a static model: if the difference between the objective value of the chosen and the current solution is smaller than a threshold T, we move to the chosen solution. Otherwise it stays at the current solution [8]. Again, the threshold is a positive control parameter which decreases with increasing number of iterations and converges to 0. Thus, in each iteration, we allow moves which do not deteriorate the current solution more than the current threshold T and finally we only allow improving moves.

The steps of the threshold accepting algorithm used in this paper are identical to the SA except that step 4 above is replaced by the following:

4. Compare the two accuracy Y(S2) and Y(S1). If the corresponding accuracy of the neighboring solution Y(S2) is higher than Y(S1), set I*=S2. Otherwise, set β = Y (S2)-Y (S1). Compare β and T. If T> β, I*=S2.

2.5.4 Linear Discriminant Analysis (LDA)

LDA is a well-known technique for dealing with the class-separating problem. LDA can be used to determine the set of the most discriminate projection axes. After projecting all the samples onto these axes, the projected samples will form the maximum between-class scatter and the minimum within-class scatter in the projective feature space [3].

Let $x_1 = \{x_1^1 \ldots . x_{l1}^1\}$ and $x_2 = \{x_1^2 \ldots . . x_{l2}^2\}$ be samples from two different classes and with some abuse of notation $x=x_1 \cup x_2=\{x_1 \ldots . x_l\}$. The linear discriminant is given by the vector W that maximizes [9]

$$J(w) = \frac{w^T S_B w}{w^T S_W w}$$

where

$$S_B = (m_1 - m_2)(m_1 - m_2)^T,$$

$$S_W = \sum_{i=1,2} \sum_{x \in x_i} (x - m_i)(x - m_i)^T$$

are the between and within class scatter matrices respectively and m_i is the mean of the classes. The intuition behind maximizing J(w) is to find a direction which maximizes the projected class means (the numerator) while minimizing the classes variance in this direction (the denominator). After we find the linear transformation matrix W, the dataset can be transformed by $y=w^T*x$. For the c-class problem, the natural generalization of linear discriminant involves c-1 discriminant functions. Thus, the projection is from a d-dimensional space to a (c-1)-dimensional space [1].

3 Experiment Results

We conducted experiments to compare the performance of the proposed feature selection algorithms. Throughout our experiments, we used a KNN classifier with *K=30*. The obtained results are shown in Table 2 and Table 3. It is clear that the system performance with the feature selection strategies is better than the system performance without the feature selection strategies. For LDA, only 9 spam e-mails among 48 spam cases and 2 legitimate e-mails of 62 legitimate cases were misclassified. For HC, only 4 spam e-mails among 48 spam cases and 3 legitimate e-mails of 62 legitimate cases were misclassified. For TA, only 3 spam e-mails among 48 spam cases and 3 legitimate e-mails of 62 legitimate cases were misclassified. For SA, only 4 spam e-mails among 48 spam cases and 1 legitimate e-mail of 62 legitimate cases were misclassified. Among all the four strategies, SA reached the best performance. Accuracy ordering: SA> TA> HC > LDA.

Table 2 Best Accuracy Classification Results

	Without FSS	LDA	HC	TA	SA
Accuracy	88.1%	90.0%	93.6%	94.6%	95.5%
Spam Precision	85.7%	98.1%	93.6%	93.8%	92.2%
Spam Recall	87.5%	81.3%	91.7%	93.8%	97.9%
Spam F1	86.6%	87.6%	92.6%	93.8%	94.6%
Legitimate Precision	90.1%	97.0%	93.7%	95.2%	98.3%
Legitimate Recall	88.7%	96.8%	95.2%	95.2%	93.6%
Legitimate F1	89.4%	91.6%	94.4%	95.2%	95.9%

Table 3 Accuracy Comparison for 10 Different Initial Solutions

	Hill Climbing	Threshold Accepting	Simulated Annealing
Average	90.2%	92.0%	92.2%
Min	86.4%	90.0%	88.2%
Max	93.6%	94.6%	95.5%
VAR	0.00053366	0.00021715	0.00064630

4 Conclusions

In this paper, we proposed the use of three different local search optimization techniques as feature selection strategies for application in spam e-mail filtering. The experimental results show that the proposed strategies not only reduce the dimensions of the e-mail, but also improve the performance of the classification filter. We obtained a classification accuracy of 90.0% for LDA, 93.6% for HC, 94.6% for TA and 95.5% for SA as compared to 88.1% for the system without feature selection.

REFERENCES

[1] R. Duda, P. Hart and D. Stork, "Pattern Classification," John Wiley and Sons, 2001.

[2] N. Soonthornphisaj, K. Chaikulseriwat and P. Tng-on, "Anti-Spam Filtering: A Centroid-Based Classification Approach," 6th IEEE International Conference on Signal Processing , pp. 1096 - 1099 , 2002.

[3] L.F. Chen, H.Y.M. Liao, M.T. Ko, J.C. Lin and G.J Yu, "A new LDA-based face recognition system which can solve the small sample size problem." Patten recognition. vol. 33. pp. 1713-1726, 2000.

[4] C. Lai and M. Tsai, "An empirical performance comparison of machine learning methods for spam email categorization," Proceeding of the 4th international conference on hybrid intelligent systems (HIS'04), 2004.

[5] J. F. Pang, D. Bu and S. Bai, "Research and Implementation of Text Categorization System Based on VSM," Application Research of Computers, 2001.

[6] S. Kirkpatrick, C. D. Gelatt and M. P. Vecchi, "Optimization by Simulated Annealing." Science, pp 671-580, May 1983.

[7] J. A. Clark, J. L. Jacob and S. Stepney, "The Design of S-Boxes by Simulated Annealing." Evolutionary Computation, pp. 1533 – 1537, Vol.2, June 2004.

[8] J. Hurink "Introduction to Local Search." http://wwwhome.math.utwente.nl/~hurinkjl/socra-text.pdf

[9] S. Mika, G. Ratsch, J. Weston, B. Scholkopf and K. Mullers, "Fisher discriminant analysis with kernels." Neural Networks for Signal Processing IX, 1999. pp.41 - 48 Madison, WI Aug. 1999

[10] http://iit.demokritos.gr/skel/i-config/downloads