Proceedings of the World Congress on Engineering 2011 Vol III
WCE 2011, July 6 - 8, 2011, London, U.K.

A Fuzzy Clustering Approach to Filter Spam E-Mail

N.T.Mohammad

Abstract— Spam email, is the practice of frequently sending unwanted email messages, usually with commercial content, in large quantities to a set of indiscriminate email accounts. However, since spammers continuously improve their techniques in order to compromise the spam filters, building a spam filter that can be incrementally learned and adapted became an active research field. Researches employed machine learning techniques which have been widely used in solving similar problems like document classification and pattern recognition, such as Naïve Bayesian, and Support Vector Machine. In this Paper, we examine the use of the fuzzy clustering algorithm (Fuzzy C-Means) to build a spam filter. The proposed use of the Fuzzy has been tested on different data set sizes collected from Spam assassin corpora by real user's emails. After testing Fuzzy C-Means using Heterogeneous Value Difference Metric with variable percentages of spam and using a standard model of assessment for the spam problem, we demonstrate the potential value of our approach.

Index Terms— Spam filtering, Fuzzy clustering, Fuzzy C-Means.

I. INTRODUCTION

Spam, or unwanted commercial email, has become an increasing problem in recent years. Estimates suggest that perhaps 70% of all email traffic is spam. As spam clutters inboxes, time and effort must be devoted to either deleting it after it is received, or preventing it from even reaching the user [9]. The problem of spam multiplies daily, and is an annoyance to every user of email. Some estimates suggest that the average per person is 10 working days per year spent solely dealing with spam[10].

Commercially available spam filters must judge an email to decide whether it is spam or legitimate, colloquially called 'ham'. These rely mostly on pattern matching rules that are manually constructed [2] .The construction of such rules is not a trivial task and requires expertise.

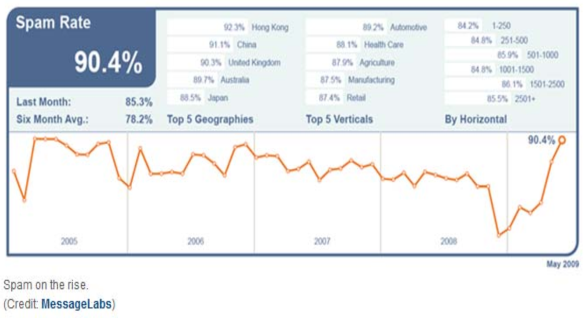

Fig. 1. Spam as a percentage of all email traffic

N.T.Mohammad is with Department of Computer Information Systems, University of Jordan. (email: nehaya.mohammad@yahoo.com)

As shown in Figure 1 [9], the proportion of spam to legitimate email changes over time, which puts the spam filtering in the category of skewed class distribution problems [6].

Spam to ham email proportion varies from person to person as well as over time. Fawcett also suggests that the proportion of spam is influenced by the email recipient's domain, how easy it is to acquire that email address, and how long an address has been in existence.

A misclassified spam that arrives in a user's inbox is annoying. A misclassified ham that the user never sees may result in loss of business, productivity, opportunity, or time. Spammers actively attempt to defeat spam filters by substituting look-alike characters for letters, hiding random text in an email, misspelling words, including pictures that show the advertisement, or embedding links into deceptively-phrased emails. Their techniques change daily. Therefore any anti-spam technology must be able to adapt quickly. Automated methods of spam filtering that can learn how to distinguish spam emails from ham emails and can be trained – learn in an updatable fashion - are of vital importance. A good anti-spam technique will have three characteristics: it will accurately classify spam and ham, it will be easily adaptable, and it will be easily scalable.

Most of the current research in spam filtering concentrates on using data mining approaches to solve the spam filtering. According to data mining, the spam is a classification problem where the filtering system aims at distinguishing spam from legitimate (ham) emails. Thus, classification algorithms that are widely used for pattern recognition can be used to solve the spam problem.

In this paper, we study the spam filtering as a data mining and an AI problem. We aim to evaluate the current state of research and propose our own solution to the problem. We present an implemented system that is based on using strong distinguishing features of spam email and a classification technique that suits the problem.

The rest of the paper is organized as follows. Section 2 overviews spam filtering work that look at common methods of fighting spam, including artificial intelligence influenced techniques. The fuzzy spam filtering technique is given in section 3. Results are given in section 4. Finally, conclusions and future work are given in section 5.

II. RELATED WORK

Various techniques exist for filtering spam. These methods can be generally categorized into techniques that have been influenced by artificial intelligence and machine learning, and other techniques. These other techniques tend to be older and less robust. For example, use of white lists, black lists, and gray lists is straightforward; if the email is sent from a known spammer, it is marked as spam; if it is sent from a user-approved address, it is allowed through to the inbox. Anything else is "gray listed" to a folder where the user can approve it as valid or mark it as spam. The difficulty with this approach is that the burden on the user can be considerable. Rules-based spam filters apply pre-written rules to a spam, such as "if subject contains 'Viagra', email is spam". These may accidentally result in misclassification of a real email as spam, classification of spam as valid email, and must be updated frequently to stay abreast of spammers' techniques. Both of these techniques have their place; however, they should not be relied upon as the only filter.

Content-based filters are founded on the premise that it is possible to create a set of rules, exemplars or features that represent the degree to which an email is to be considered as a spam, and that if this is over some threshold, is considered to be spam. Such filters have been the focus of considerable interest, with work on rule-based filters, nearest neighbor classifiers [12], decision trees [5] and Bayesian classifiers [11]. Initial implementations of these filters were centralized, but with spam comprising 50% of all emails traffic.

As the knowledge base is now in the hands of the system administrators, it can be customized to suit the characteristic email and spam that individual domains receive. Users can feed information back about false positives and false negatives that enables the filter to be retrained. Spam Assassin given in [13] is perhaps the most known example of this approach. Thus the huge content-based filters have been developed towards a higher degree of collaboration as they have become decentralized Clutters.

Machine learning techniques are more varied and flexible. Decision trees [5] classify email as spam or ham based on previous data. They are costly to calculate and recalculate as spammers change techniques. Bayesian networks [11] are the most popular anti-spam technique currently, but they can be difficult to scale up and rely on many features to make their judgments.

In this paper, we evaluate the use of fuzzy clustering and text mining for spam filtering. Fuzzy clustering is a scalable and easy to update approach. If each email that comes in is used as part of the data pool to make decisions about future emails, spam trends will be detected and adapted to automatically. There is not the large cost of recalculation that would occur with decision trees, or the manual maintenance of rules-based filters.

Even if there was one optimal solution to the spam problem currently, with thousands of spammers looking for new ways to defeat it, the payoff of research into alternate methods is apparent. In this paper, we evaluate the use of text mining and fuzzy clustering as an anti-spam technique.

III. FUZZY SPAM FILTER MODEL

Figure 2 shows the main steps of the spam filter developed in this paper. The filter consists of three main stages: feature extraction, training and testing. In the coming discussion, each of these stages is discussed.

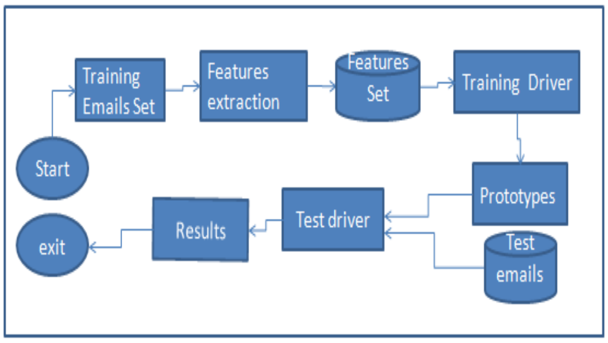

Fig. 2. Spam Filter Model

A. *Feature Extraction*

Before we could extract features, we first had to find the data we would use for testing and training. We sought actual emails, both spam and valid, and as up to date as possible. There are several corpuses of email available on the web; we eventually used two of the collections from the SpamAssassin Spam Corpus [13]. These contained over 3,000 emails submitted by various people, all labeled as either spam or ham, in a text-only format. Many other projects and experiments have been performed upon SpamAssassin corpuses, and they are easily available. Therefore, making the experiments we performed easy to replicate and to compare with other approaches. Figure 3 shows how the text mining works in our approach to finally provide the classification of the emails.

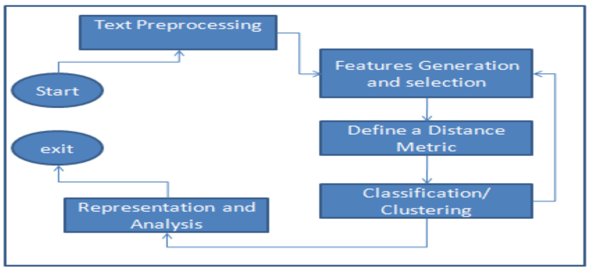
Text Preprocessing
Features Generation and selection
Start
Define a Distance Metric
exit
Representation and Analysis
Classification/ Clustering

Fig. 3. Text mining system main phase

The next step in the procedure was to define and extract features. The main difficulty we encountered here was in choosing features to extract. Though most of the previous works emphasized the importance of feature selection for accurate spam detection, they only give abroad description of the features they used.

We came up with a preliminary list of features that included email length in characters, percent of non alphanumeric characters in the email, percent of white space characters, average word length, and number of email addresses in the header. After running the first few experiments, we added a "word list" of suspected spam words such as "mortgage" and kept a count of how many of those words appeared in the email, as well as counting HTML tags and whether font or background colors had been set. Each email was run through our Java program to extract these features, which were saved to an individual file as values separated by blank spaces. The files were named with an "s" for spam or "h" for ham, followed by the email id number, to make our record keeping easier. Data file name was not a feature we used in calculating the clusters. Our Java code took a folder and processed every file in it, producing one feature file per initial data file.

Once this information is extracted, it fed it to the fuzzy clustering algorithm. As our experiments progressed, features were refined, or expanded.

B. Fuzzy Clustering

Fuzzy clustering has been successfully applied to a variety of problems ranging from vector quantization coding and neural networks training, to more specific fields as diverse as food classification, water quality analysis, and weather forecast. However, the two main fields where fuzzy clustering excels are pattern recognition and image processing. Fuzzy clustering for pattern recognition can be applied to a text classification problem such as spam, where the patterns to be classified are texts.

While many clustering algorithms have been introduced, the Fuzzy C-Mean (FCM) algorithm, first presented by Bezdek [4] is the most popular one. FCM assumes the number of clusters is known or if not known, then at least some fixed number. Each of the c clusters is represented by a prototype v_i. These prototypes are chosen randomly at the beginning and each training vector is assigned a degree of membership to clusters with respect to the vector's distance from the cluster prototype. The cluster prototypes are then replaced by the center of gravity of the vectors that belong to each cluster. The algorithm repeatedly alters assignment of patterns to their nearest cluster and updates prototypes until the algorithm converges. Convergence is reached when changes are less than a specified threshold. Figure 4 describes the steps performed by the FCM.

FCM aims to minimize the objective function given in equation 1:

$$\boldsymbol{f} = \sum_{i=1}^{c}\sum_{j=1}^{n} \boldsymbol{u_{ij} d_{ij}} \qquad (1)$$

Under the constraint in equation 2:

$$\boldsymbol{f} = \sum_{i=1}^{c} \boldsymbol{u_{ij}} = \boldsymbol{1} \quad \text{,for j =1,…,n} \qquad (2)$$

Where:
$u_{ij} \in \{0,1\}$ indicates the degree of membership of vector v_j in cluster c_i.

<u>Step 1:</u> Select the number of clusters c, initial partition matrix u, the termination criterion ε. Also, set the iteration index l to 0. Set the fuzzifier [1,2]. Choose a Distance Metric.

<u>Step 2:</u> Calculate the fuzzy cluster centers

$$\boldsymbol{V_i} = \frac{\sum_{x \in X}(\boldsymbol{\mu_i(x)})^2}{\sum_{x \in X}(\boldsymbol{\mu_i(x)})^2} \boldsymbol{, 1 \le i \le c}$$

<u>Step 3:</u> Calculate the new partition matrix

$$u_{ik} = \left[\sum_{j=1}^{c}\left(\frac{D_{ik}}{D_{jk}}\right)^{\frac{2}{m-1}}\right]^{-1}, \forall i, k$$

<u>Step 4:</u> if maximum change in u > ε, return to step 2.

Fig. 4. Fuzzy C-Mean Clustering Algorithm (FCM)

When FCM is trained, a given vector is declared to belong to the cluster for which it has the maximum membership. A modified approach constraints the degree of membership to exceed a predefined threshold [16]. The level of success of the FCM algorithm relies on three main factors. First,setting the Correct number of clusters is vital. Second, the fuzzifier m in formula 4 must be set. Third, a suitable distance metric must be selected. Choosing these settings is a non-trivial task. Many approaches have been proposed for predicting the number of clusters and for setting the fuzzifier [16] . A discussion of such approaches is beyond the scope of this paper.

For our work, we adopted the following two simple approaches:

1. Experimentally setting the value of the fuzzifier.

2. Experimentally finding the minimum number of the clusters that gives the best results using the same environmental setting. Clusters that contain less than a predefined number of vectors are discarded. Vectors which are members of these discarded clusters are also discarded as outliers

C. Distance Metric

A learning algorithm must have a bias in order to generalize. A bias is "a rule or method that causes an algorithm to choose one generalized output over another" (Mitchell 1980). Wilson [14] showed that no learning algorithm can generalize more accurately than any other when summed over all possible problems. The bias of the learning algorithm depends on the distance metric used. This gives the conclusion that no distance metric can be better than any other in terms of generalization ability, when considering all possible problems [14].

A distance metric can be suitable to a particular problem or set of problems if it can improve the generalization ability of the learning algorithm by being able to catch the characteristics of the problem, normally presented by a vector in clustering problems. Wilson [14] presented a distance metric that handles problems involving nominal and continuous features. The distance metric improved performance on a collection of 48 applications. Since our features vector consists of nominal and continuous features. The *Heterogeneous Value Difference Metric* (HVDM) is used.

The HVDM is defined as:

$$\mathrm{HVDM}(x,y)=\sqrt{d_a^2(x_a,y_a)} \qquad (5)$$

Where *m* is the number of attributes, function $d_a(x,y)$ returns the distance between two values *x* and *y* for attribute *a* and is defined as:

$$f=\begin{cases}\mathbf{1,}\ \text{if x or y is unknown}\\ \text{normalized\_vdm}(x,y),\ \text{if f is nominal}\\ \text{normalized\_diff}(x,y),\ \text{if a is linear}\end{cases} \qquad (6)$$

The function *normalized_diff* is defined in equation (7) and function *normalized_vdm* have three alternatives, the one used in this paper is labeled as N1 in [14] and is defined in equation (8).

$$\boldsymbol{normalized\_diff}_a(x,y)=\frac{|x-y|}{4\sigma_a} \qquad (7)$$

$$\boldsymbol{normalized\_vdm}_a=\sum\left|\frac{N_{a,x,c}}{N_{a,x}}-\frac{N_{a,y,c}}{N_{a,y}}\right| \qquad (8)$$

IV. Results

As stated in the introduction, the proportion of spam emails to ham emails varies from person to person and over time. As a result, it is not easy to prove the performance of any spam filter since it will be biased to its training and testing sets. Fawcett [6] suggested using the *Probability Cost Function (PCF)*, which is the x-axis of a cost curve as a non-biased measure of spam filter's performance. The filter is viewed as a "spam detector" where a spam email is a positive input and a ham email is a negative one. A spam PCF can be defined as:

$$PCF_{spam}=\frac{p(spam)*\cos t(FalseNegative)}{p(spam)*\cos t(FalseNegative)+p(ham)*\cos t(FalsePositive)}$$

Where p(spam) and p(ham) is the probability of a spam email and a ham email respectively. False Positive is a classifying a ham email as a spam while false negative is considering a spam email as a ham.
Clearly, a false positive error has a greater cost than a false negative. As suggested in [6] a value of 10 times cost of the negative false can be used, accordingly the PCF for spam and ham can be reduced to:

$$\mathrm{PCF}_{\mathrm{spam}}=\frac{\mathrm{p(spam)}}{\mathrm{p(spam)}+\mathrm{p(ham)}*10} \qquad (9)$$

$$\mathrm{PCF}_{\mathrm{ham}}=\frac{\mathrm{p(ham)}*10}{\mathrm{p(spam)}+\mathrm{p(ham)}*10} \qquad (10)$$

Equations (9) and (10) will be used to evaluate the performance of the proposed spam filter.

A. Experiments

Experiment 1

In order to validate the presented approach, several experiments were conducted. The goal is to measure the performance of the various enhancements as well as coming out of a model that gives the best performance using the proposed approach, namely, the FCM algorithm. As stated in the feature selection discussion, in order to reduce the dimensionality of the feature space and to provide the classifier with domain-knowledge features, feature selection is performed.

In experiment 1, we evaluated the performance of the FCM algorithm as a spam filter using Euclidian distance as a distance metric and without performing any normalization on the feature's values. Table 1 summarizes the various settings used to conduct the experiment. Note that the initial number of clusters and the fuzzifier values are experimentally set.

TABLE 1: SETTINGS FOR EXPERIMENT 1

FCM Parameters	Initial number of clusters	12
	fuzzifier	1.9
	Distance Metric	Euclidian distance
	Initial setting of weights	Random
	Stopping criteria	Max change in Uij< ϵ E=.0001
Training set	Spam proportion	70%
	Ham proportion	30%
	Size	2000
	Normalization	None

Experiment 1 results are given in table 2. The performance was below the desired values of the PCF for both the spam and the ham emails. To understand why that might happen, consider the sample values of some of the features presented in table 3, which indicates a wide variety in the feature value ranges. Features with large values and wider ranges will be dominant; for instance, the total number of characters varies more between emails than does the number of HTML tags. In addition, Euclidian distance is difficult to calculate properly when used with discrete values. Clearly normalization and a more suitable distance metric are required to have a better performance.

TABLE 2: RESULTS OF EXPERIMENT 1

	Error Rates	Number of misclassified emails
Total Error Rate	66%	851
False Positive	34.1%	133
False Negative	52%	473
PCF(Spam)	19%	
PCF(Ham)	81%	

Experiment 2

Here, we applied the HVDM described in the distance metric section on our set of features before feeding then into the clustering algorithm, using the same sitting as in experiment 1. Table 4 gives a summary of the experiment. Clearly, the Performance is dramatically better than the previous one and exceeds the PCF for both spam and ham emails. This agrees with the results obtained by [Wilson] regarding the suitability of the HVDM metric to the text classification problems set of which spam filtering is part.
To verify the performance of our approach, we run several tests with variable proportions of spam to ham emails starting from 10% spam up to 90% spam. The results can be seen in figures 3 and 4. Varying the proportion of the spam emails increased the false negative error when the proportion of spam emails was lower, but the error rate was still quite low.

Since the total amount of spam is small, the actual amount of misclassified spam should not be a problem. In comparison, the false positive error rate shows almost stable behavior, even when the proportion of ham was only 10%. The false positive error did not exceed 1.5%. Since the repercussions of false positives are much higher than that of false negatives, our results are promising.

The false positive rate is too high compared to commercial spam filters but we believe it could be lowered with additional tweaking of features extracted from the sample emails, a large sample population, or perhaps modification of the distance metric.

TABLE 3: SAMPLE FEATURE VALUES FOR A HAM EMAIL.

Attribute Name	***Value***
Total number of characters in the message	*2424*
Number of white space characters	*244*
Number of special characters	*558*
Number of words	*240*
Average word length	*7.6*
Percent of special characters	*23.2*
Percent of non-alphanumeric characters	*32.9*
Number of URLs	*3*
Number of suspected Spam words	*11*
Number of HTML tags	*0*
Number of colors	*1*
Average number of characters per word	*7.1*
Percent of total characters to special characters	*4.9*
Percent of total characters to Alphanumeric characters	*13.3*
Percent of capital word to simple word	*0.01*
Percent of capital characters to total characters	*0.005*
Number style format used in the html code such that (head, body, font,...,etc)	*0*
Number of smile face symbols	*0*

TABLE 4: RESULTS OF EXPERIMENT 2

	Error Rates	Number of misclassified emails
Total Error Rate	4.5%	59
False Positive	0.7%	3
False Negative	6.1%	56
PCF(Spam)	19%	
PCF(Ham)	81%	

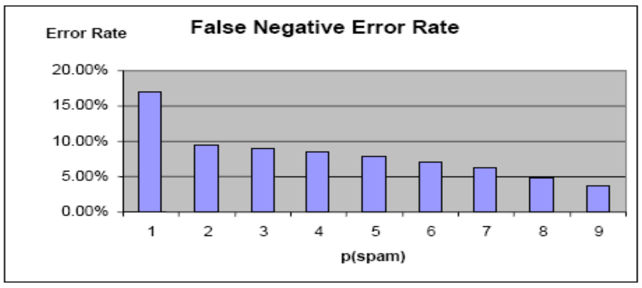

Fig. 4. False Negative Error Rate. Note number on x-axis is multiplied by 10, so 9 stands for 90%.

The higher false negative rate can be explained by the type of extracted features, which seem to be biased toward classifying email as ham. Consider the 'number of spam

words' attribute, a very strong feature in most spam filtering approaches. In approaches like the one used in [11], a predefined set of words is given to the classifier. If the size of the set is *n,* and number of spam words is *m*, then *m/n* of the given words are spam which decreases the false negative error if the email is a real spam, but increases the false negatives if the email is a ham.
However, in our approach, this will only strengthen one feature, which is combined with other features to decide whether the email is a real spam.

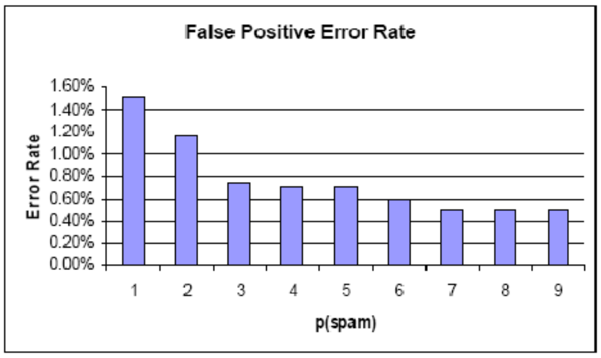

Fig 5. False Positive Error Rate. Note numbers x-axis is multiplied by 10, so 9 stands for 90%.

V. Conclusion and Future Work

Spam Filtering is a problem of great importance and has gained a great attention in the last decade. The problem's difficulty and interestingness arises from the changing nature of spam. The high accuracy required from any useful spam filter makes the problem even more demanding. In this paper, the Fuzzy C-Mean Clustering algorithm is evaluated as a tool of building a spam filter. The algorithm was tested with a set of features normalized using the HDVM function. The approach has been testing using a variant proportion of spam emails which reflects nature of the problem. The approach is evaluated using a standard model suggested by [6] for evaluating spam filters.

The results gained were promising. The false positive error rate did not exceed 1.5% and stabled around 0.7% when ham emails proportion is more than 50%. We achieved between 16% to 4% for the false negative error rate. These results support our hypothesis regarding the suitability of the combined approaches used. Our approach takes advantage of the generalization ability of the FCM algorithm, extracts representative features from the data, and uses a suitable distance metric.

Finally, Our future work includes Optimizing Parameters; which are the fuzzifier value and number of clusters that give the best classification success rate compared with the spam filter techniques.,

References

[1] S. Aksoy, and R. M. Haralick, "Feature normalization and likelihood-based similarity measures for image retrieval". Pattern Recognition Letters, 22(5):563—582., 2001.

[2] I. Androutsopoulos, J. Koutsias, K. V. Chandrinos, and C. D. Spyropoulos, "An experimental comparison of naive bayesian and keyword-based anti-spam Filtering with personal e-mail messages", In Proc. Of SIGIR-2000, ACM, 2000.

[3] I. Androutsopoulos, J. Koutsias, K. V. Chandrinos, G. Paliouras, and C. Spyropoulos, 2000, "An evaluation of naive bayesian anti-spam Filtering", In Proc. of the Workshop on Machine Learning in the New Information Age, 9-17, 2000.

[4] J. C. Bezdek, "Pattern Recognition with Fuzzy Objective Function Algorithms", Plenum Press, New York, 1981.

[5] Carreras, X., and Marquez, L., 2000, "Boosting trees for anti-spam email filtering", Proc. of SIGIR-2000, ACM, 160-167.

[6] T. Fawcett, "In vivo" spam filtering: a challenge problem for KDD, ACM SIGKDD Explorations Newsletter, 5(2): 141-148, 2003.

[7] R. Feldman, Y. Aumann, M. Fresko, O. Liphstat, B. Rosenfeld, and Y. Schler, "Text Mining via Information Extraction", In Principles of Data Mining and Knowledge Discovery, Chapter in Book. Pages 165-173, 1999.

[8] M. A. Hearst, "Untangling text data mining". In Proceedings of the 37th Annual Meeting of the Association For Computational Linguistics on Computational Linguistics (College Park, Maryland, June 20 - 26, 1999). Annual Meeting of the ACL. Association for Computational Linguistics, Morristown.

[9] Message Labs Spam Intercepts data, 2006, http://www.messagelabs.com/publishedcontent/publish/threat_watch_dotcom_en/threat_statistics/spam_intercepts/DA_114633.chp.html

[10] N. Nie, A. Simpser, I. Stepanikova, and L. Zheng, "Ten years after the birth of the internet, how do americans use the internet in their daily lives?" Technical report, Stanford University, 2004.

[11] M. Sahami, S. Dumasi, D. Heckerman, and E. Horvitz, "A bayesian approach to filtering junk e-mail. In Learning for Text Categorization", Papers from the 1998 Workshop, Madison, Wisconsin, 1998.

[12] G. Sakkis, I. Androutsopoulos, G. Paliouras, V. Karkaletsis, C. Spyropoulos, and C. Stamatopoulos, "A Memory-Based Approach to Anti-Spam Filtering for Mailing Lists", Information Retrieval, 6:49–73, 2003.

[13] SpamAssassin Public Corpus, 2006, http://spamassassin.apache.org/publiccorpus/

[14] D. R. Wilson, and T. R. Martinez, "Improved Heterogeneous Distance Functions", J. Artificial Intelligence, Res. 6: 133-140, 1997.

[15] E. P. Xing, A. Y. Ng, M. I. Jordan, and S. Russell, "Distance metric learning, with application to clustering with side-information", In Advances in Neural Information Processing Systems 15, pages 505--512, Cambridge, MA, 2003. MIT Press.

[16] S. Nascimento, "Fuzzy Clustering Via Proportional Membership Model", IOS Press, 2005.