

# Combating Good Word Attacks on Statistical Spam Filters with Multiple Instance Learning

Yan Zhou, Zach Jorgensen, Meador Inge

School of Computer and Information Sciences

University of South Alabama, Mobile, AL 36688 USA

zhou@cis.usouthal.edu, zdjorgen@jaguar1.usouthal.edu, wmi601@jaguar1.usouthal.edu

## Abstract

*Statistical spam filters are known to be vulnerable to adversarial attacks. One such adversarial attack, known as the Good Word Attack, thwarts spam filters by appending to spam messages sets of “good” words, which are common in legitimate e-mail but rare in spam. We present a counter-attack strategy that first attempts to differentiate spam from legitimate e-mail in the input space, by transforming each e-mail into a bag of multiple segments, and subsequently applies multiple instance logistic regression on the bags. We treat each segment in the bag as an instance. An e-mail is classified as spam if at least one instance in the corresponding bag is spam, and as legitimate if all the instances in it are legitimate. We show that a spam filter using our multiple instance counter-attack strategy stands up better to good word attacks than its single instance counterpart and the commonly practiced Bayesian filters.*

## 1. Introduction

It has been nearly thirty years since the debut of the first e-mail spam. Today, to most end users, spam does not seem to be a serious threat due to the effectiveness of spam filters. Behind the scenes, however, is a seemingly never-ending battle between spammers and spam fighters. With millions of e-mail users, profit-driven spammers have great incentives to spam. With as little as 0.001% response rate, a spammer could potentially profit \$25,000 on a \$50 product [3]. Over the years, spammers have grown in sophistication with cutting-edge technologies and have become more evasive. The best evidence of their growing effectiveness is a recent estimate of over US\$10 billion worldwide spam-related cost [7]. In this paper, we target one of the attacking techniques spammers often use on existing spam filters.

Adversarial attacks on spam filters have become an increasing challenge to the anti-spam community. The Good

Word Attack [11] is one of the most popular techniques frequently employed by spammers. This technique involves appending to spam messages sets of “good” words that are common to legitimate e-mails (ham) but rare in spam. Spam messages inflated with good words are more likely to bypass spam filters. So far, relatively little research has been done on how spam filters can be trained to account for such attacks. This paper presents an effective defense strategy using multiple instance (MI) learning against spam disguised with good words.

The concept of multiple instance learning was initially proposed by Dietterich et al. for predicting drug activities [5]. The challenge of identifying a drug molecule that binds strongly to a drug target is that a drug molecule can have multiple conformations. A molecule is positive if at least one of its conformations binds tightly to the target, and negative if none of its conformations bind well to the target. The problem was tackled with an MI model that aims to learn axis-parallel rectangles (APR). Later, learning APR in the multiple instance setting was further studied and proved to be NP-complete by several other researchers in the PAC-learning framework [1, 9, 2]. Several probabilistic models, Diverse Density (DD) [12] and its variation EMDD [20], and multiple instance logistic regression (MILR) [14], employ a maximum log-likelihood estimation to solve problems in the MI domain. More recent work in this area has focused on reducing a multiple instance learning problem to a single instance learning problem [6, 16, 13, 21].

Our spam filtering strategy adopts the classical MI assumption, i.e. a bag is positive if at least one of its instances is positive, and negative if all instances are negative. We treat each e-mail as a bag of instances. An e-mail is classified as spam if at least one instance in the corresponding bag is spam, and as legitimate if all the instances in it are legitimate. We split each e-mail into multiple instances so that even when the spam is inflated with good words, a multiple instance learner is still able to recognize the spam part of the message. Our experimental results show that a multiple instance learner stands up better to good word attacks

than its single instance counterpart and the Bayesian filters.

In the remainder of our paper, we first discuss recent research that has motivated our work. Next, we formalize the spam filtering problem as a multiple instance learning problem. We present our experimental results to demonstrate the effectiveness of our filtering strategy. Finally, we conclude our work and discuss future directions.

## 2. Related Work

Our work is primarily motivated by recent research on adversarial learning [4, 10, 8]. Dalvi et al. consider classification as a game between classifiers and adversaries in problem domains where adversarial attacks are expected [4]. They model the computation of the adversary's optimal strategy as a constrained optimization problem and approximate its solution based on dynamic programming. Subsequently, an optimal classifier is produced against the optimal adversarial strategy. Their experimental results demonstrate that their game-theoretic approach outperforms traditional classifiers in the spam filtering domain. However, in their adversarial classification framework, they assume both the classifier and the adversary have perfect knowledge of each other, which is practically unrealistic.

Instead of assuming the adversary has perfect knowledge of the classifier, Lowd and Meek formalize the task of adversarial learning as the process of reverse engineering the classifier [10]. In their adversarial classifier reverse engineer (ACRE) framework, the adversary aims to identify difficult spam instances, the ones that are hard to detect by the classifier, through membership queries. The goal is to find a set of negative instances with minimum adversarial cost within a polynomial number of membership queries.

A practical aspect of adversarial learning is the techniques used to initiate the attacks on classifiers. One of the common techniques is known as the *Good Word Attack*. Lowd and Meek [11] present and evaluate several variations of the good word attack that involves appending "good" words that are common to legitimate e-mails to spam messages. There are two different ways to carry out the attack: passively and actively. Active good word attacks use feedback obtained by sending test messages to a spam filter in order to determine which words are good. The active attacks were found to be more effective than the passive attacks. However, active attacks are generally more difficult than passive attacks because they require user-level access to the spam filter, which is not always possible. Passive good word attacks, on the other hand, do not involve any "feedback" from the spam filter and guesses are made as to what words are considered good. Lowd and Meek identify three common ways that good words can be chosen for passive attacks. First, dictionary attacks involve selecting random words from a large collection of words, such as a

dictionary. In testing, this method did not prove to be effective; in fact, it actually increased the chances that the email would be classified as spam. Next, frequent word attacks involve the selection of words that occur most often in legitimate messages, such as news articles. This method was more effective than the previous one, but it still required as many as 1,000 words to be added to the original message, on average. Finally, frequency ratio attacks involve the selection of words that occur very often in legitimate messages but not in spam messages. The authors' tests showed that this technique was quite effective, resulting in the average spam message being passed off as legitimate by adding as few as 150 words to the e-mail.

In this paper, we demonstrate that an effective counter-attack strategy can be developed in the framework of multiple instance learning, provided that a single instance can be properly transformed into a bag of instances. Our experiments also verify earlier observations, discussed in [11, 17], that re-training on e-mails modified during adversarial attacks may improve the performance of the filters.

## 3. Problem Definition

Consider a standard supervised learning problem with a set of training data  $D = \{ \langle X_1, Y_1 \rangle, \dots, \langle X_m, Y_m \rangle \}$ , where  $X_i$  is an instance represented as a single feature vector,  $Y_i = C(X_i)$  is the target value of  $X_i$ , where  $C$  is the target function. Normally, the task is to learn  $C$  given  $D$ . The learning task becomes more difficult when there are adversaries who could alter some instance  $X$  so that  $X \rightarrow X'$  and  $C(X) \neq C(X')$ . Let  $\Delta X$  be the difference between  $X$  and  $X'$ , i.e.  $X' = X + \Delta X$ . There are two cases that need to be studied separately:

1. the training set is clean; no training instances are altered. Adversaries can only modify examples in the test set.
2. both the training set and the test set are compromised.

In the first case, the classifier is trained on a clean training set. Predictions made for altered instances are highly unreliable. In the second case, the classifier may capture some adversarial patterns as long as the adversaries consistently follow a particular pattern.

In both cases, the problem becomes trivial if we know how the instances are altered. We could recover the original data and solve the problem as if no instances were altered by the adversary. In reality, knowing exactly how the instances are altered is impossible. Instead, we seek to approximately separate  $X$  and  $\Delta X$  and treat them as separate instances in a bag. We apply multiple instance learning to learn a hypothesis defined over a set of bags.

## 4. Multiple Instance Bag Creation

We formulate the spam filtering problem as a multiple instance binary classification problem in the context of adversarial attacks. Note that the adversary is only interested in altering positive instances, i.e. spam, by appending sets of “good” words that are commonly encountered in negative instances, i.e. legitimate e-mails, often referred to as ham in practice. We propose three different approaches to creating multiple-instance bags. We call them split-half (split-H), split-term (split-T), and split-projection (split-P).

### 4.1. Split-H

In our first splitting method, *split-half*, we split every e-mail right down the middle into approximately equal halves. Formally, let  $B = \{B_1, \dots, B_i, \dots, B_m\}$  be a set of bags (e-mails), where  $B_i = \{X_{i1}, X_{i2}\}$  is the  $i^{\text{th}}$  bag,  $X_{i1}$  and  $X_{i2}$  are the two instances in the  $i^{\text{th}}$  bag, created from the upper half and the lower half of the e-mail respectively. This splitting approach is reasonable in practice because spammers usually append a section of good words to either the beginning or the end of an e-mail to ensure the legibility of the spam message.

There are two possible ways for spammers to attack spam filters with splitting methods of this nature. One is to create a visual pattern with good words so that the original spam message is still legible after the attack, but the spam is fragmented in such a way that “spammy” words are well separated and become less indicative of spam in the presence of a number of good words. The following example demonstrates how this idea can work effectively to deceive this type of filter.

From: foo@internet.org
To: foo-foo@email.org
Subject: meeting agenda

goodwords	...	<b>low</b>	...	goodwords
goodwords	...	<b>mortgage</b>	...	goodwords
goodwords	...	<b>rate</b>	...	goodwords

The second way to defeat the split-H method is to append a very large block of good words so that after the split, good words outweigh spam-indicative words in all instances.

### 4.2. Split-T

The second splitting method, *split-term*, partitions a message into three groups of terms (words) based on whether the word is an indicator of spam, an indicator of legitimate, or neutral, i.e.  $B_i = \{X_{is}, X_{in}, X_{il}\}$ , where

$X_{is}$  is the spam-likely instance,  $X_{in}$  is the neutral instance, and  $X_{il}$  is the legit-likely instance. The determination for each word is based on a weight generated for it during pre-processing. These weights are calculated using word frequencies obtained from the spam and legitimate messages. More specifically, the weight of a term  $W$  is given as follows:

$$weight(W) = \frac{s(W)}{s(W) + h(W)},$$

where

$$s(W) = \frac{\text{number of spam e-mails containing the word } W}{\text{total number of spam e-mails}}$$

$$h(W) = \frac{\text{number of ham e-mails containing the word } W}{\text{total number of ham e-mails}}.$$

We considered any word with a weight less than  $thresh_s$  to be spammy, any word with a weight greater than  $thresh_\ell$  to be legitimate, and any word with a weight in between to be neutral.  $thresh_s$  is selected such that 20% of the terms have a weight less than or equal to it.  $thresh_\ell$  is selected so that 50% of the terms have a weight greater than or equal to it. Since most legitimate messages have at least a few terms that are more or less spam-indicative, the threshold values are chosen asymmetrically so that relatively fewer words are considered “spammy” in order to reduce the risk of false positives.

### 4.3. Split-P

The third splitting method, *split-projection*, transforms each message into a bag of two instances by projecting the message vector onto the spam and ham prototype vectors. The prototype vectors are computed from the spam cluster and the ham cluster in the training set. More specifically, let  $C_s$  be the set of instances that are spam and  $C_\ell$  be the set of instances that are legitimate. The prototypes are computed using Rocchio’s algorithm [15] as follows:

$$P_s = \beta \cdot 1/|C_s| \cdot \sum_{i=1}^{|C_s|} C_{s_i} - \gamma \cdot 1/|C_\ell| \cdot \sum_{i=1}^{|C_\ell|} C_{\ell_i}$$

$$P_\ell = \beta \cdot 1/|C_\ell| \cdot \sum_{i=1}^{|C_\ell|} C_{\ell_i} - \gamma \cdot 1/|C_s| \cdot \sum_{i=1}^{|C_s|} C_{s_i}$$

where  $C_{s_i}$  is the  $i^{\text{th}}$  spam instance in  $C_s$  and  $C_{\ell_i}$  is the  $i^{\text{th}}$  ham instance in  $C_\ell$ ,  $\beta$  is a fixed constant suggested to be 16 and  $\gamma$  is a fixed constant suggested to be 4. Given a message  $M$ , the two instances are formed by projecting  $M$  onto  $P_s$  and  $P_\ell$  respectively. The rationale of this splitting approach rests on the assumption that a message is close to the spam prototype in terms of cosine similarity if it is indeed spam, and a ham is close to the ham prototype.

To attack split-T and split-P, adversaries would need to initiate active attacks through trial and error, which is only feasible if they have access to the spam filters at the user level.

Once we create bags of instances, we can transform a standard supervised learning problem into a multiple instance learning problem under the standard MI assumption. In this paper, we adopt the multiple instance logistic regression model to train a spam filter that is more robust to good word attacks than traditional spam filters.

## 5. Multiple Instance Logistic Regression

Given a set of training bags

$$B = \{ \langle B_1, Y_1 \rangle, \dots, \langle B_i, Y_i \rangle, \dots, \langle B_m, Y_m \rangle \},$$

let  $Pr(Y_i = 1 | B_i)$  be the probability that the  $i^{th}$  bag is positive, and  $Pr(Y_i = 0 | B_i)$  be the probability that it is negative. The bag-level binomial log-likelihood function is:

$$L = \sum_{i=1}^m [Y_i \log Pr(Y_i = 1 | B_i) + (1 - Y_i) \log Pr(Y_i = 0 | B_i)]$$

We do not have direct measure of bag-level probabilities in the log-likelihood function. However, we can compute  $Pr(Y_i = 0 | B_i)$  with instance-level class probabilities  $Pr(Y_j = 0 | X_{ij}) = \frac{1}{1 + \exp(p \cdot X_{ij})}$ , where  $X_{ij}$  is the  $j^{th}$  instance in the  $i^{th}$  bag, and  $p$  is the parameter vector that needs to be estimated. A bag is negative if and only if all the instances in the bag are negative. Thus,

$$\begin{aligned} Pr(Y_i = 0 | B_i) &= \prod_{j=1}^n Pr(Y_j = 0 | X_{ij}) \\ &= \exp\left(-\sum_{j=1}^n (\log(1 + \exp(p \cdot X_{ij})))\right) \end{aligned}$$

where  $n$  is the number of instances in the  $i^{th}$  bag, and

$$Pr(Y_i = 1 | B_i) = 1 - Pr(Y_i = 0 | B_i).$$

We can now apply maximum likelihood estimation (MLE) to maximize the bag-level log-likelihood function, and estimate the parameter vector  $p$  that maximizes the probability of observing the bags in  $B$ .

## 6. Experimental Results

We test our multiple instance learning counter-attack strategy on e-mails from the 2006 TREC Public Spam Corpus<sup>1</sup>. We simulate good word attacks by generating a list of

<sup>1</sup><http://plg.uwaterloo.ca/~gvcormac/treccorpus06/>

good words from the corpus and randomly selecting some to add to the spam messages in the data sets. We compare our multiple instance logistic regression classifiers to their single instance learning counterpart – logistic regression (LR), support vector machine (SVM), and the multinomial naïve Bayes (MNB) classifier.

### 6.1. Experimental Data

Our experimental data consists of 36,674 spam and legitimate e-mail messages from the 2006 TREC spam corpus. We preprocessed the entire corpus by stripping HTML and attachments, and applying stemming and stop-list to all terms. Messages that became empty after preprocessing were discarded.

### 6.2. Experiment Setup

For our experiments we sorted the messages in the corpus by their receiving date and evenly divided them into 11 subsets  $\{D_1, \dots, D_{11}\}$ . In other words, the messages in subset  $i$  come chronologically before the messages in  $i + 1$ . Experiments were run in the online fashion, i.e. training on subset  $i$  and testing on subset  $i + 1$  for  $i = 1, \dots, 10$ . Each subset contains approximately 3300 messages. The percentage of spam messages in each subset varies as in the operational settings (see Figure 1). We used the Multiple Instance Learning Tool Kit (MILK) [19] and Weka 3.4.7, a well known open-source machine learning tool [18] in our experiments. We reduced the feature space to the top 500 words ranked using information gain. Attribute values for each message are calculated using the *tf-idf* (term frequency inverse document frequency) weighting scheme.

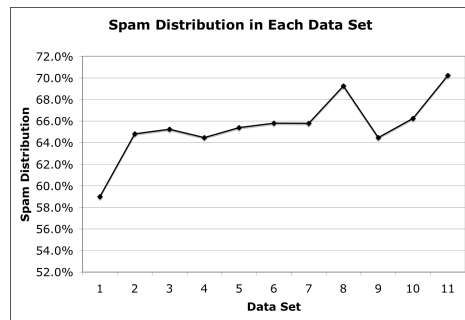


Figure 1. Percentage of spam in each dataset.

### 6.3. Good Word List

To simulate the good word attacks, we generated a list of good words using all the messages in the TREC spam cor-

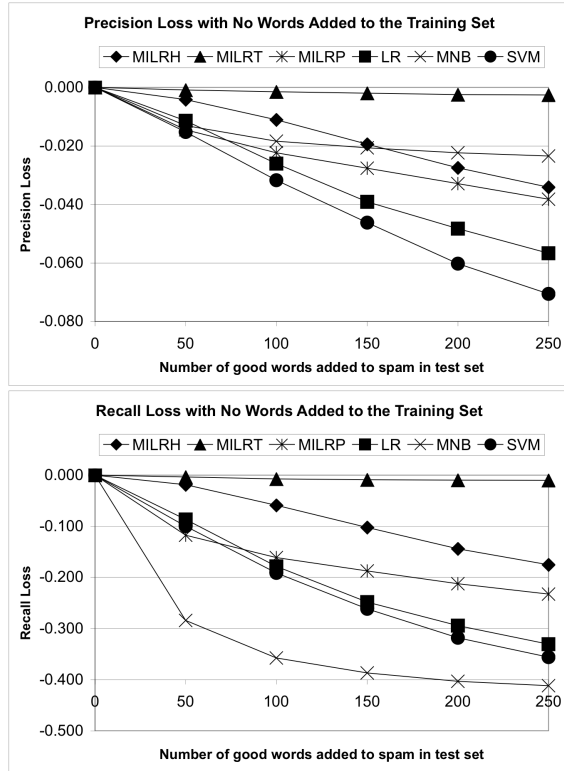
pus. We ranked each word according to the ratio of its frequency in the legitimate messages over its frequency in the spam messages. Any words that appeared only in the legitimate messages and not in the spam messages were not included in the ranking as described by Lowd and Meek [11]. We then chose the top 1,000 words from this ranking to use as the good word list in our experiments. In the real world, it is highly unlikely that a spammer would have access to the set of e-mails from which the training set for the spam filter was obtained; however, we generated our good word list in this way so that the good words would have an observable effect on the accuracy of the classifiers.

### 6.4. Experiment 1

In the first experiment, we tested the MILR-based classifiers on e-mail injected with varied amounts of good words. We also tested the standard logistic regression, support vector machine and multinomial naïve Bayes classifiers for comparison. The classifiers were each tested on the eleven chronologically sorted data sets in an on-line fashion, i.e. all of the classifiers were trained on the same unaltered data set  $D_i$ , and then tested on the data set  $D_{i+1}$ , for all  $i$  from 1 to 10. Six variations of each test set were created to test the susceptibility of the classifiers to good word attacks of varying strength. The first version of each test set was left unmodified, i.e. no good words were added to the test set. With each successive version of the test set, the number of good words added to half of the spam messages increased in increments of 50 words up to 250. The good words added to each message were randomly selected from our good word list, without replacement or regard to rank, on a message by message basis. The precision and recall on each version of the test set for all 10 tests were averaged and recorded for each classifier. MILR was tested using the split-H, split-T and split-P methods to split the messages into multiple instances. In our results, we use “MILRH”, “MILRT” and “MILRP” when split-H, split-T and split-P are used with MILR, respectively.

Figure 2 shows how the precision and recall values are affected by the number of good words added to the test set. MILRT is barely affected by the attacks. MILRH and MILRP apparently suffer less precision and recall loss compared to LR and SVM, and suffer much less recall loss compared to MNB. More specifically, the average recall value for MILRH, MILRT, and MILRP dropped 17.9% (from 0.98 to 0.80), 1.1% (from 0.93 to 0.92), and 24.4% (from 0.95 to 0.72) respectively, as a result of the good word attack while the average recall value of LR, SVM and MNB dropped by 34.2% (from 0.97 to 0.64), 36.6% (from 0.97 to 0.62), and 45.1% (from 0.91 to 0.50) respectively. In the same experiment, the drop in precision for MILRH, MILRT and MILRP was 4.0%, 0.3%, and 4.3% while the drop in

precision for LR, SVM and MNB was 6.2%, 8.0% and 2.4% respectively.



**Figure 2. Precision & Recall loss when good words are added to 50% of the test set, and no good words are added to the training set.**

Figure 3 shows the change in F-measure, which is the harmonic mean of the precision and recall values, as the number of good words added to the test set increases. As can be observed, the MILR-based classifiers stand up much better to the good word attack than do LR, SVM and MNB. Among MILRH, MILRT and MILRP, MILRT appears to be most effective against the attacks.

Figure 4 shows the success rate of MILRH, MILRT, MILRP, LR, SVM and MNB as the number of good words added to the test set increases. Again, MILR classifiers outperform LR, SVM and MNB as more and more good words are added.

### 6.5. Experiment 2

In the second experiment, our goal was to observe the effect that training on messages injected with good words had on the susceptibility of the classifiers to attacks on the test set. We again tested each of the classifiers on the 11 chronologically sorted data sets in an on-line fashion. This

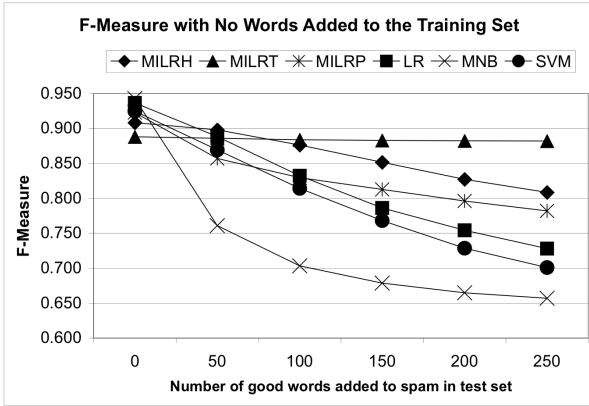


Figure 3. F-measure when good words are added to 50% of the test set.

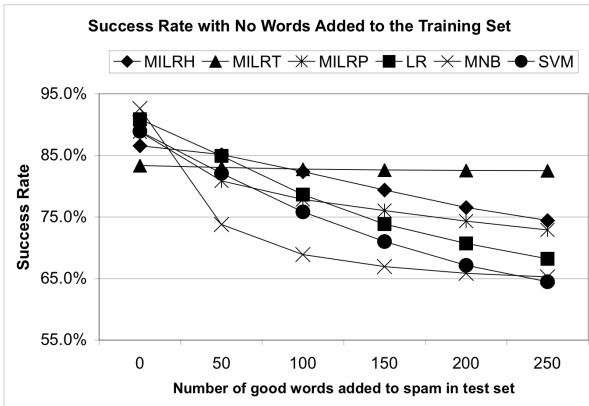


Figure 4. Success rate when good words are added to 50% of the test set.

on each version of the test set were averaged and recorded for each classifier.

Figures 5, 6, 7, 8, and 9 show the success rate and the F-measure of the six algorithms as the number of good words added to 50% of the spam messages in the training set increases from 10 to 50.

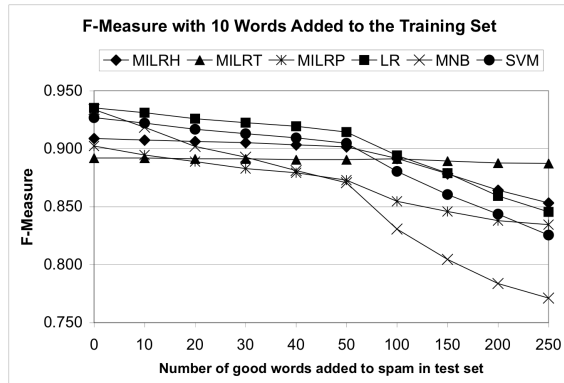
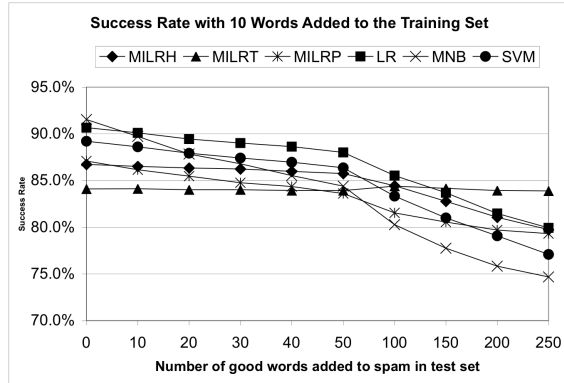


Figure 5. Success rate and F-measure when 10 good words are added to 50% of the spam in the training set.

time, we also created five versions of the training set. We injected 10 good words into the first version of the training set and then increased the number of injected good words by 10 for each successive version, up to 50 good words for the fifth version. We also tried injecting larger numbers of good words, but after exceeding 50 words, the effect was minimal; therefore, the corresponding results are not reported here. For each version of the training set we tested the classifiers on 10 versions of the corresponding test set. The first version of each test set was left unmodified. With each successive version of the test set, half of the spam messages were injected with an increasingly larger number of good words, first in increments of 10 words for the first five versions and then in increments of 50 words for the next four versions. As before, good words were selected from the good word list randomly and without replacement or regard to rank, on a message by message basis. For all ten tests on the chronologically sorted data sets, the precision and recall

The general observations are:

- when some of the spam messages in the training set are modified with a small number of good words, the performance of all classifiers except MILRT worsens as the number of good words added to the spam in the test set increases, as shown in Figure 5.
- but as the training spam is inflated with more good words, the attacks become less effective as shown in Figure 6, 7, 8, 9. In our experiments, we observed that after the number of good words added to the training spam exceeds 50, the added good words actually help all classifiers correctly identify spam, which is in line with the results reported by some other researchers [11, 17]. Due to space limit, we do not show all the plots with more than 50 words added to the training spam.

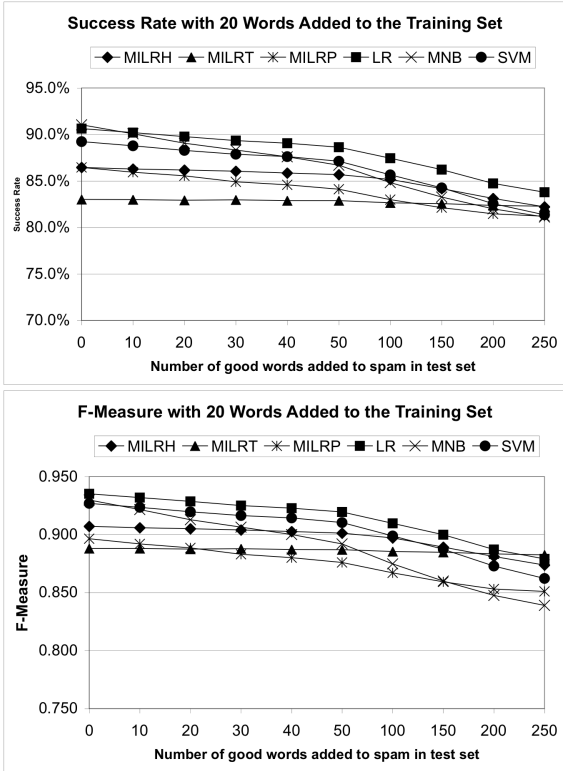


Figure 6. Success rate and F-measure when 20 good words are added to 50% of the spam in the training set.

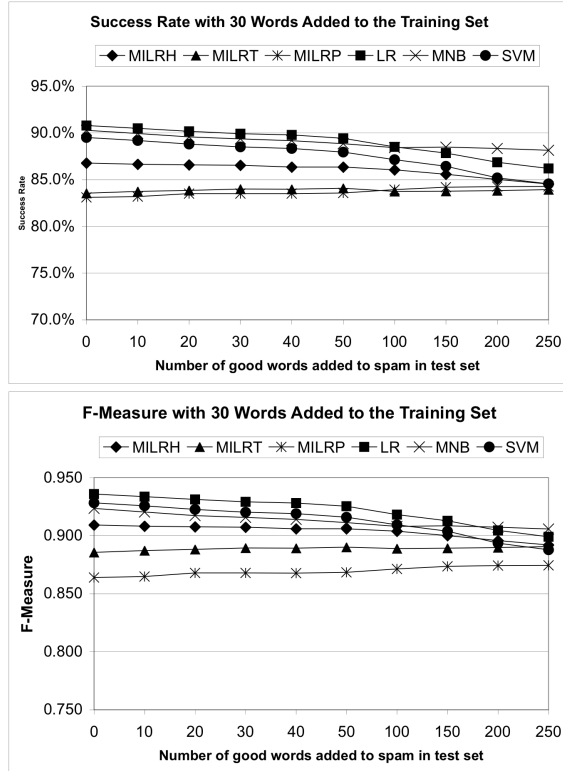


Figure 7. Success rate and F-measure when 30 good words are added to 50% of the spam in the training set.

## 7. Conclusions and Future Work

We propose a multiple instance counter-attack strategy for combating good word attacks on statistical e-mail spam filters. We treat each e-mail as a bag of multiple instances. A logistic model at the instance level is learned indirectly by maximizing the bag-level binomial log-likelihood function. We have demonstrated that a multiple instance learner stands up better to good word attacks than the standard single instance learners: logistic regression, support vector machine, and the often practiced naïve Bayes model.

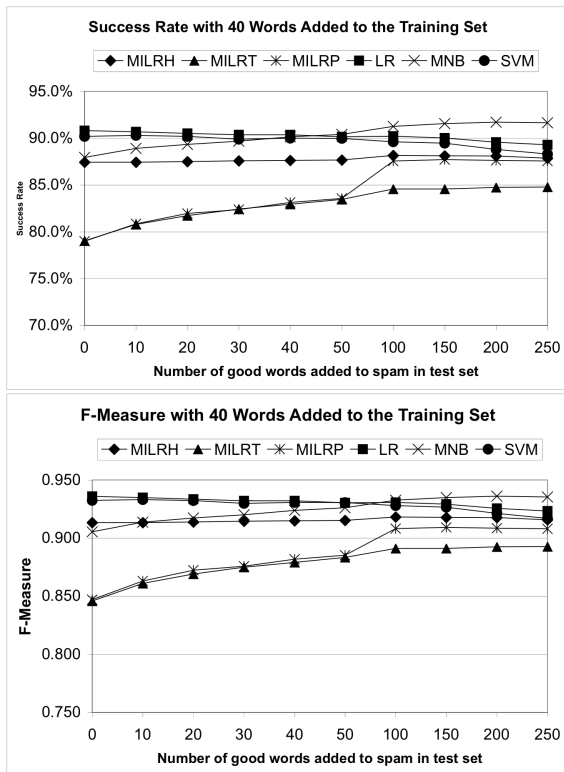
Our preliminary results demonstrate the great potential of the proposed multiple instance learning strategy against good word attacks on spam filters. However, due to the complexity of the problem nature, we need to carefully investigate the influences of various parameters, such as the distribution of spam in the corpus, the size of the training set, the percentage of spam in either the training or the test set being inflated with good words, and the number of good words appended to spam messages.

In addition, we plan to extend the proposed multiple instance learning strategy to handle adversarial attacks in general. Since it is an arms race between spammers and the

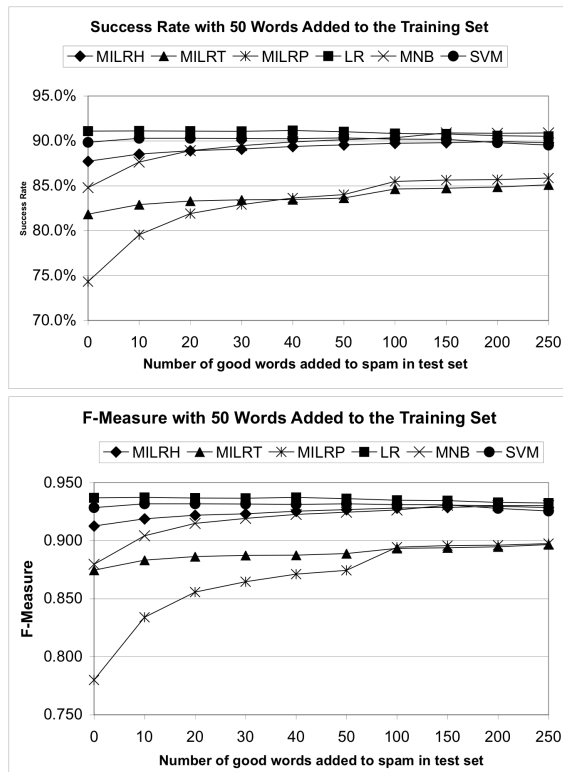
filter designer, we also plan to make our MI strategy adaptive as new spam techniques are devised, and online as the concept of spam drifts over time.

## References

- [1] P. Auer. On learning from multi-instance examples: Empirical evaluation of a theoretical approach. In *Proceedings of the 14th International Conference on Machine Learning*, pages 21–29, San Francisco, CA, 1997. Morgan Kaufmann.
- [2] A. Blum and A. Kalai. A note on learning from multiple-instance examples. *Machine Learning*, 30(1):23–30, 1998.
- [3] J. Carpinter and R. Hunt. Tightening the net: A review of current and next generation spam filtering tools. *Computers and Security*, 25(8):566–578, 2006.
- [4] N. Dalvi, P. Domingos, Mausam, S. Sanghai, and D. Verma. Adversarial classification. In *Proceedings of the 2004 ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 99–108. ACM Press, 2004.
- [5] T. Dietterich, R. Lathrop, and T. Lozano-Pérez. Solving the multiple-instance problem with axis-parallel rectangles. *Artificial Intelligence Journal*, 89(1-2):31–71, 1997.
- [6] T. Gärtner, P. Flach, A. Kowalczyk, and A. Smola. Multi-instance kernels. In *Proceedings of the 19th International*



**Figure 8. Success rate and F-measure when 40 good words are added to 50% of the spam in the training set.**



**Figure 9. Success rate and F-measure as 50 good words are added to 50% of the spam in the training set.**

*Conference on Machine Learning*, pages 179–186, San Francisco, CA, 2002. Morgan Kaufmann.

- [7] R. Jennings. The global economic impact of spam. Technical report, Ferris Research, 2005.
- [8] J. Z. Kolter and M. A. Maloof. Using additive expert ensembles to cope with concept drift. In *Proceedings of the Twenty-second International Conference on Machine Learning*, pages 449–456, New York, NY, 2005. ACM Press.
- [9] P. Long and L. Tan. PAC learning axis-aligned rectangles with respect to product distribution from multiple-instance examples. *Machine Learning*, 30(1):7–21, 1998.
- [10] D. Lowd and C. Meek. Adversarial learning. In *Proceedings of the 2005 ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 641–647. ACM Press, 2005.
- [11] D. Lowd and C. Meek. Good word attacks on statistical spam filters. In *Proceedings of the 2nd Conference on Email and Anti-Spam*, 2005.
- [12] O. Maron and T. Lozano-Pérez. A framework for multiple-instance learning. *Advances in Neural Information Processing Systems*, 10:570–576, 1998.
- [13] J. Ramon and L. Raedt. Multi instance neural networks. In *Proceedings of ICML-2000 workshop on Attribute-Value and Relational Learning*, 2000.
- [14] S. Ray and M. Craven. Supervised versus multiple instance learning: An empirical comparison. In *Proceedings of the*

*22nd International Conference on Machine Learning*, pages 697–704, New York, NY, 2005. ACM Press.

- [15] J. Rocchio Jr. Relevance feedback in information retrieval. In *The SMART Retrieval System: Experiments in Automatic Document Processing*, pages 68–73. Prentice Hall, 1971.
- [16] J. Wang and J. Zucker. Solving the multiple-instance learning problem: A lazy learning approach. In *Proceedings of the 17th International Conference on Machine Learning*, pages 1119–1125, San Francisco, CA, 2000.
- [17] S. Webb, S. Chitti, and C. Pu. An experimental evaluation of spam filter performance and robustness against attack. In *The 1st International Conference on Collaborative Computing: Networking, Applications and Worksharing*, pages 19–21, 2005.
- [18] I. Witten and E. Frank. *Data Mining: Practical machine learning tools with Java implementations*. Morgan Kaufmann, San Francisco, CA, USA, 2000.
- [19] X. Xu. Statistical learning in multiple instance problems. Master’s thesis, University of Waikato, 2003.
- [20] Q. Zhang and S. Goldman. EM-DD: An improved multiple-instance learning technique. In *Proceedings of the 2001 Neural Information Processing Systems (NIPS) Conference*, pages 1073–1080, Cambridge, MA, 2002. MIT Press.
- [21] Z. H. Zhou, K. Jiang, and M. Li. Multi-instance learning based web mining. *Applied Intelligence*, 22(2):135–147, 2005.