Online Active Learning Methods for Fast Label-Efficient Spam Filtering

D. Sculley
Department of Computer Science
Tufts University, Medford, MA USA
dsculley@cs.tufts.edu

ABSTRACT

Active learning methods seek to reduce the number of labeled examples needed to train an effective classifier, and have natural appeal in spam filtering applications where trustworthy labels for messages may be costly to acquire. Past investigations of active learning in spam filtering have focused on the pool-based scenario, where there is assumed to be a large, unlabeled data set and the goal is to iteratively identify the best subset of examples for which to request labels. However, even with optimizations this is a costly approach. We investigate an *online* active learning scenario where the filter is exposed to a stream of messages which must be classified one at a time. The filter may only request a label for a given message immediately after it has been classified. The goal is to achieve strong online classification performance with few label requests. This is a novel scenario for low-cost active spam filtering, fitting for application in large-scale systems. We draw from the label efficient machine learning literature to investigate several approaches to selective sampling in this scenario using linear classifiers. We show that online active learning can dramatically reduce labeling and training costs with negligible additional overhead while maintaining high levels of classification performance.

1. INTRODUCTION

Statistical spam filters require labeled examples of *spam* (unwanted or harmful electronic messages) and *ham* (legitimate electronic messages). Labeled data is used for training machine learning classifiers to distinguish between the two classes. This methodology has been largely successful when given large, fully labeled data sets [5] [4]. However, in practice it may be costly to acquire labels for training data. Indeed, for real-world spam filters, it is unreasonable to assume that a user will label every message: such a requirement defeats the purpose of the filter.

Active learning methods have been developed in the machine learning community to reduce labeling cost by identifying informative examples for which to request labels. It has been shown in practice that only a small portion of a large unlabeled data set may need to be labeled to train an active learner that achieves strong classification performance [13] [7] [17] [22] [2]. Thus, active learning is an appealing tool for real-world spam filtering.

The *pool-based* approach to active learning has previously been applied to spam filtering, with good results [14] [21] [4]. Similar to prior results in text classification [13], it has been shown that only a small subset of a larger unlabeled email data set needs to be labeled to achieve strong performance. Several active learning methods are appropriate for this task, including uncertainty sampling [13], version space reduction [22], and query by committee [7]. However, the iterative pool-based approach is computationally expensive, often requiring many passes through the entire unlabeled data set. Segal et al. introduced an efficient approximation that reduced this cost, but still required at least one full pass through an entire unlabeled data set before any labels could be requested [21].

We depart from the pool-based approach and investigate the novel use of online active learning methods for spam filtering. In this online case, the filter is exposed to a stream of messages and is asked to classify them one by one. At each point in time, the active filter may choose to request a label for the given example. The goal here is to create a strong classifier while requesting as few labels as possible. This approach has several advantages over the pool-based methodology. First, it reflects the actual application scenario of real-world spam filters, which are applied in online (and often real-time) settings. A user may be much more willing to label a new incoming message than one from the past, especially if label requests are made relatively infrequently. Second, online active learning enables solutions with only $O(1)$ additional computation and storage costs. The online active learning scenario involves no repeated passes over unlabeled data, which is the primary source of computational cost in pool-based active learning, and does not require storage of a large pool of unlabeled examples.

In the remainder of this paper, we review related work in both pool-based and online active learning. We then describe several online active learning active strategies for linear classifiers. We test these methods on spam filtering tasks using three linear classifiers: classical Perceptron, Perceptron with Margins, and linear Online Support Vector Machines. We find strong results with these methods, greatly reducing the number of labels needed to achieve strong classification performance on two large benchmark data sets. These results exceed the performance of uniform subsampling on both data sets, and also out-perform the pool-based active learning methods from the 2006 TREC spam filtering

CEAS 2007 - Fourth Conference on Email and Anti-Spam, August 2-3, 2007, Mountain View, California USA

competition by at least an order of magnitude. We conclude with a discussion on the implication of these results for spam filtering and user interface.

2. RELATED WORK

Active learning is a well-studied branch of machine learning. Although pool-based active learning has received more attention, there has also been significant work in online active learning (sometimes referred to as *label efficient* learning). In this section, we explore connections between this work and the goal of spam filtering with online active learning.

2.1 Pool-based Active Learning

As discussed in the introduction, there have been a variety of pool-based active learning methods proposed in the literature. There have been several examinations of pool-based active learning for spam filtering [21], including a task in the 2006 TREC spam filtering competition [4]. Pool-based active learning assumes that the learner has access to a pool of n unlabeled examples, and is able to request labels for up to $m << n$ examples on each labeling round.

There are several methods for selecting these m examples. Uncertainty sampling [13] requests the m examples for which the current hypothesis has the least confidence in classification. Another method is to request labels for those examples that will most reduce the size of the version space [22]. The Query by Committee algorithm is another approach to version space reduction, that relies on predictions from hypotheses sampled from the version space [7]. It is also possible to request labels for those examples which are estimated to most greatly reduce training error [17].

There are two issues with these pool-based active learners, as applied to spam filtering. First is cost. In an iterative pool-based scheme, each of the n examples must be re-evaluated on each iteration. Some methods, such as version space reduction, Query by Committee, and estimation of error rate reduction are expensive can incur expensive evaluation cost. But even with inexpensive evaluation methods such as uncertainty sampling, or the `simple` method of version space reduction [22], the cost of active learning is still $O(ni)$ for a pool of n examples over i iterations. For large email systems, this cost may be prohibitive. Segal et al. have proposed a method of reducing this cost for spam filtering with approximation to uncertainty sampling [21], but even here the entire pool of unlabeled examples must be examined at least once before labels may be requested. Thus, the main overhead in pool-based active learning is in choosing examples for which to request label.

The second issue with pool-based active learning is that in practical settings, spam filtering is most naturally viewed as an online task. Emails enter a system in a stream, not a pool. Online active learning enables clean user-interface for requesting labels from actual users in real time.

Pool-based active learning does have one potential advantage over online active learning. Because pool-based active learning considers an entire data set at once, it is possible that pool-based active learning may identify the *optimal* subset of training examples. Online active learning necessarily uses a greedy strategy that may not select the optimal subset. However, in practice, many pool-based active learning methods also employ a version of greedy selection, as constructing the optimal training set can be extremely expensive. Thus, the online approach may not necessarily suffer in this regard.

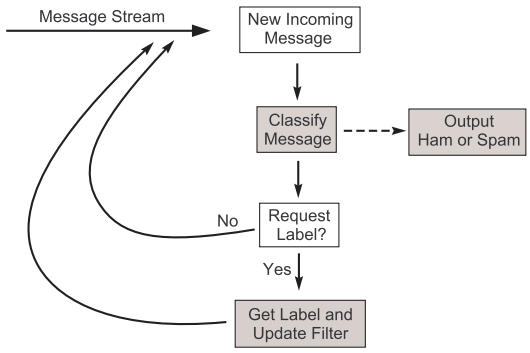

Figure 1: Online Active Learning.

2.2 Online Active Learning

To the best of our knowledge, this is paper is the first examination of online active learning methods to spam filtering. The first analysis of online active learning in machine learning was performed by Helmbold and Panizza [9], under the heading of Label Efficient Learning. This work examined the tradeoffs between the cost of label requests and the cost of errors in an online learning setting. There have since been several proposals for Label Efficient learning methods for linear classifiers [3] [6], but to our knowledge the b-Sampling approach is the only approach that has been analyzed that does not require either the number of total examples in the data stream or the maximum number of label requests to be specified in advance. Because practical spam filtering is performed in an essentially unbounded online setting, it is important not to have such restrictions.

We should also point out that Query by Committee [7] is essentially an online active learning algorithm. We do not examine it in this paper because the process of sampling hypotheses from the version space is too expensive for practical spam filtering.

3. ONLINE ACTIVE LEARNING METHODS

The basic online active learning framework is shown in Figure 2.1. Messages come to the filter in a stream, and the filter must classify them one by one. At each point, the filter has the option of requesting a label for the given message, and the goal is for the filter to achieve strong classification performance with as few label requests as possible.

The issue considered in this section is: given an example and a classification score, how should the filter decide to request a label? We examine four schemes for making this decision. The first is a randomized label efficient method first proposed for linear classifiers such as classical Perceptron [2]. The second is similar, but uses a logistic sampling rule. The third borrows from the idea of uncertainty sampling, and requests labels lying within a fixed distance from the classification hyperplane. The fourth method, uniform sub-sampling, is a non-active learning method used for baseline comparison.

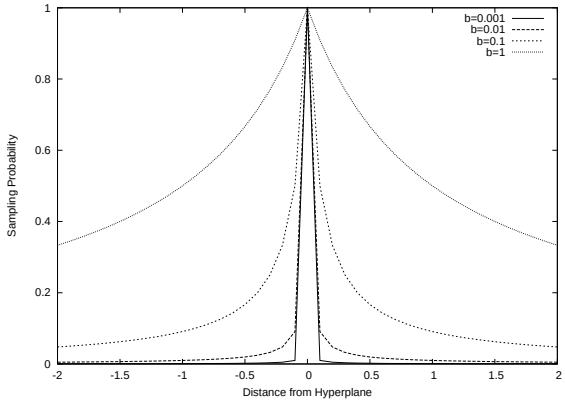

Figure 2: b-Sampling Probabilities

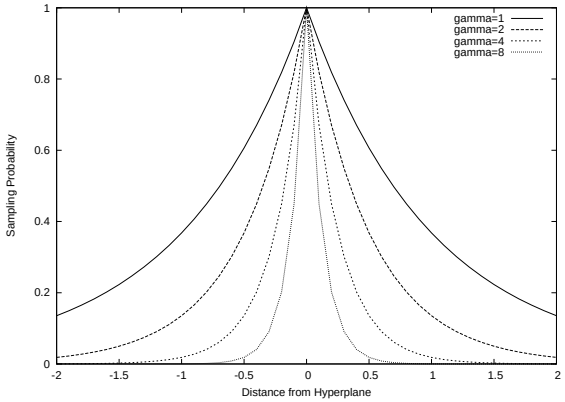

Figure 3: Logistic Margin Sampling Probabilities

The notation in this section assumes that the online active filter is built using a linear classifier, several of which are reviewed in Section 4. Individual emails are treated as *examples*, in the machine learning terminology. Assume that examples are drawn from $X \subseteq R^n$, and that each example $\mathbf{x}_i \in X$ has an associated label y_i drawn from $\{-1, 1\}$, representing ham and spam, respectively. Assume that L is a linear classifier with weight vector $\mathbf{w}$, that a value p_i is defined for each $\mathbf{x}_i$ by $p_i =< \mathbf{w}, \mathbf{x}_i >$, and the prediction by L for example $\mathbf{x}_i$ is given by $\text{sign}(p_i)$.

3.1 Label Efficient b-Sampling

Cesa-Bianci et al. introduced a label efficient method of selective sampling for linear classifiers such as classical Perceptron and Winnow, and gave theoretical mistake bounds and expected sampling rates [2]. We refer to this method as b-sampling, and describe it here.

The b sampling rule [2] is: given a sampling parameter $b > 0$, request a label for example $\mathbf{x}_i$ with probability

$$P_i = \frac{b}{b + |p_i|}$$

As $|p_i|$ approaches zero, the probability of a label request for $\mathbf{x}_i$ approaches 1. This makes intuitive sense: the closer an example is to the hyperplane, the less confidence we have in L's prediction. There is always some non-zero probability of requesting a label for examples far from the hyperplane to ensure that the hypothesis is performing well across the entire data space.

The parameter b defines a function relating the sampling probability P_i to the classification confidence $|p_i|$. To help illustrate the effects of varying b, we have mapped P_i against $|p_i|$ for several values of b as shown in Figure 2. Note that when $|p_i| = 0$, then $P_i = 1$ for all values of $b > 0$. That is, a label is always requested when the hypothesis has zero confidence.

3.2 Logistic Margin Sampling

The b-Sampling method of online active learning used one particular method of mapping $|p|$ confidence values to P sampling probabilities, in part because this method allowed clean theoretical analysis. For comparison, we propose and test another natural mapping from $|p|$ to P based on a logistic model of confidence probabilities (without theoretical analysis) that we call Logistic Margin Sampling.

For an example $\mathbf{x}_i$, let probability q_i represent the probability that the predicted label given by $\text{sign}(p_i)$ matches the true label y_i. (As before, p_i is the signed distance of $\mathbf{x}_i$ to the classification hyperplane.) Thus, q_i is the confidence that our label is correct.

In this Logistic Margin approach, we model the confidence value q_i using a logistic function:

$$q_i = \frac{1}{1 + e^{-\gamma|p_i|}}$$

This approximation is reasonable given the work of Platt [15], who showed that a similar sigmoid function is a good model for confidence values for linear classifiers.

Following the intuition behind uncertainty sampling, we request a label for $\mathbf{x}_i$ with probability $P_i = 1 - q_i$. Substituting and simplifying gives the Logistic Margin sampling probability:

$$P_i = e^{-\gamma|p_i|}$$

Like b-Sampling, Logistic Margin sampling gives the highest sampling probability to those examples lying closest to the classification hyperplane. Those examples $\mathbf{x}_i$ with $p_i = 0$ are always sampled, and every example has a non-zero sampling probability. The difference is in the shape of the distribution, shown in Figure 3 for different values of γ.

3.3 Fixed Margin Sampling

The previous two methods of online active learning are probabilistic, mapping classification confidence values to sampling probabilities. For comparison, we also investigate a deterministic variant that we call fixed margin sampling. Fixed margin sampling is a sampling heuristic that can reduce the total number of label requests needed for a given performance level, but offers no theoretical guarantees.

In fixed margin sampling, a confidence threshold c is set as a parameter. The sampling rule is straightforward: request a label for an example $\mathbf{x}_i$ when (and only when) $|p_i| < c$. Fixed margin sampling is thus visualized as a step function. Unlike the two prior methods, fixed margin sampling does not assign a non-zero sampling probability to all examples. Examples $\mathbf{x}_i$ with $|p_i| \geq c$ will never have their labels requested. Thus, the theoretical guarantees of b-Sampling do not apply to fixed margin sampling: it is possible that a bad initial hypothesis will continually make mistakes with high confidence. A learner with this hypothesis, "never in doubt,

but never correct," will not receive any label information and thus will never update.

However, in practice, we have found that fixed margin sampling can be effective for spam filtering. This is because the online linear classifiers tend to make low-confidence mistakes before they make high-confidence mistakes, due to the incremental nature of online updates for the linear classifiers we examined. Thus, they avoid this theoretical problem in actual tests. Furthermore, because fixed margin sampling does not request labels for examples the learner is confident for, this approach may require fewer labels in the long run. This is especially true when the learner is able to achieve a strong hypothesis, such as is the case in spam filtering where filters can achieve extremely high classification performance.

3.4 Baseline: Uniform Subsampling

For completeness, we also perform tests with uniform, random subsampling. This is not an active learning method, but is presented here for baseline comparison. With this method, a fixed probability value q is set as a parameter, and a label is requested for each example with probability q. Using uniform subsampling as a baseline comparison is common practice in active learning research, including active learning for spam filtering [4]. This allows us to examine the difference between a learner trained on n examples drawn at random versus a learner trained on n examples selected by an active learning method.

4. LINEAR CLASSIFIERS

Each of the online active learning methods described in the previous section assumes only that the classifier provides a distance $|p_i|$ showing how far an example $\mathbf{x}_i$ is from the classification hyperplane. Thus, these methods can be used with any linear classifier, so long as the classifier's hypothesis can be expressed as a weight vector $\mathbf{w} \in R^n$ for an n-dimensional feature space. In this paper, we examine three online linear classifiers: classical Perceptron [16], Perceptron with Margins [12] [8], and linear Online SVMs. All of these methods will share the same classification rule: the label of an unseen example $\mathbf{x}_i$ is predicted with $\text{sign}(p_i)$, where $p_i =< \mathbf{w}, \mathbf{x}_i >$. The differences between the methods lie in how they construct the hypothesis $\mathbf{w}$.

4.1 Classical Perceptron

Classical Perceptron is an online linear classifier that has been part of machine learning literature since the 1960's [16], and has received extensive theoretical analysis. The online training method for classical Perceptron is straight forward:

- Accept learning rate η as a parameter.
  Initialize $\mathbf{w} \leftarrow 0$.
- For each new unseen example $\mathbf{x}_i$, compute $p_i \leftarrow < \mathbf{w}, \mathbf{x}_i >$ and predict y_i' as $\text{sign}(p_i)$. If $y_i' \neq y_i$, update: $\mathbf{w} \leftarrow \mathbf{w} + y_i \eta \mathbf{x}_i$.

With normalized, noiseless data, linearly separable with margin γ, classical Perceptron has been shown to converge to a correct hypothesis with at most $1/\gamma^2$ mistakes. When data is non-linearly separable, perhaps because of noise, convergence is not guaranteed and performance may suffer. Online training for classical Perceptron is fast, requiring only $O(m)$ updates, where m is the number of mistakes made in training.

4.2 Perceptron with Margins

Perceptron with Margins is a variant of the classical algorithm that attempts to maintain an approximate margin between the data classes [12] [8], and has been shown to give good tolerance to noise in practice [11], and has given strong previous results on spam filtering [20]. The online training method for Perceptron with Margins is as follows.

- Accept parameters: margin m, learning rate η.
  Initialize $\mathbf{w} \leftarrow 0$.
- For each new unseen example $\mathbf{x}_i$, compute $p_i \leftarrow < \mathbf{w}, \mathbf{x}_i >$ and predict y_i' as $\text{sign}(p_i)$. If $y_i p_i \leq m$, update: $\mathbf{w} \leftarrow \mathbf{w} + y_i \eta \mathbf{x}_i$.

Perceptron with Margins requires more online training updates than classical Perceptron, needing at most $8/\gamma^2$ updates for separable data with margin γ [1]. The benefit of Perceptron with Margins is that it tends to give improved performance with noisy data, non-separable data, or data separable with a very small margin γ [12] [8] [11].

4.3 Linear Online SVMs

Linear Support Vector Machines (SVMs) are a current machine learning method that give state of the art performance on text classification. SVMs are maximum margin classifiers, finding a separating hyperplane that maximizes the distance between two data classes. See [18] for a complete discussion.

In the soft-margin case (allowing for noise), the hypothesis $\mathbf{w}$ found by SVM training is the one that solves the following quadratic programming problem. Given m training examples in n-dimensional feature space, and a set value of tradeoff parameter $C > 0$, find the hypothesis vector $\mathbf{w}$ and slack vector ξ to minimize:

$$\tau(\mathbf{w}, \xi) = \frac{1}{2}||\mathbf{w}||^2 + C \sum_{i=1}^{m} \xi_i$$

subject to the constraints [18] that $\xi_i \geq 0$ and $y_i p_i \geq 1 - \xi_i$ for $i = \{1, ..., m\}$.

A naive conversion of this batch-mode Linear SVM to an online learning method is straightforward [19], if expensive. This is accomplished by re-training on the entire set of seen data each time an example is classified poorly: that is, with $y_i p_i < 1$. For iterative solvers such as SMO, this process can be made less expensive by using the old hypothesis as the starting point for re-training. These linear Online SVMs have been shown to give excellent performance on spam filtering [19], with appropriate setting of the C parameter.

The primary drawback to SVMs is that they are computationally expensive, requiring training time that is roughly quadratic in the size of the data set.[1] However, it has been shown that online updates for linear SVMs in spam filtering are relatively inexpensive, because the hypothesis tends to be relatively stable [19]. Furthermore, online training cost can be greatly reduced by optimizing over only the q most recent examples while maintaining strong classification performance, and relaxing the update and convergence requirements (See [19] for details). We use this approach in this paper.

[1]Note that Joachims has proposed a linear solution for linear SVMs [10], which will be investigated for spam filtering in future work.

5. EXPERIMENTS

In this section, we report results from experiments testing the effectiveness of the online active learning methods from Section 2 with the learning methods described in Section 3 on spam filtering. These results show strong support for the use of online active learning in spam filtering.

5.1 Data Sets

We use two large, publicly available benchmark data sets first developed for the TREC spam filtering competitions: `trec05p-1` containing 92,189 total messages [5] and `trec06p` containing 37,822 total messages [4]. These data sets each have a canonical ordering for online classification, which we employ here for repeatability. Initial testing and parameter tuning was performed on a separate tuning data set, the publicly available `spamassassin_corpus`.

5.2 Feature Space

For all of our tests, we use a simple binary 4-gram feature mapping drawn from the first 3,000 characters of each message. Formally, the first 3,000 characters of each message (including all headers, content, and any attachments) is treated as a single string s_i with label $y_i \in \{-1, 1\}$. Assume that s_i consists of characters from alphabet Σ, and let examples be represented as feature vectors in space $\{0,1\}^{|\Sigma|^4}$, with a unique feature for each possible string of 4 contiguous characters, referred to as a 4-gram. Map s_i to $\mathbf{x}_i$ by assigning a 1 for each contiguous 4-gram feature appearing in s_i, and a 0 for all other features. Finally, normalize $\mathbf{x}_i$ by its Euclidean norm. We choose this feature space because it is simple, language independent, and has given strong results on spam classification in previous work [19].

5.3 Classification Performance

We tested each of the active learning methods with each of the base machine learning methods on both `trec05p-1` and `trec06p`. We varied the parameter values of the active learning methods to assess performance with different total numbers of label requests. For b-Sampling, b was varied between 0.001 and 1, for Logistic Margin sampling γ was varied between 1 and 16, for Fixed Margin sampling the value m was varied from .001 to 2.4, and for uniform subsampling p was varied from .001 to .512. These parameter values were chosen to keep the total number of label requests within the range of a minimum of 0 to a maximum of roughly 30,000 requested labels. Probabilistic tests were repeated 10 times each, with mean results reported.

The results are given in Figures 4-9, using the standard (1-ROCA)% measure [5] as the performance measure. Each graph shows the (1-ROCA)% score (on the vertical axis) achieved over the entire online test by an active learner requesting a given number of labels (on the horizontal axis) during that test.

These results show a clear win for online active learning methods, compared to random subsampling. In all cases, the online active learning methods dominated random subsampling in the full range of label requests tested, and achieved equivalent performance levels using as little as 10% of the labels needed by random subsampling to achieve the same performance level. Note also that although the different learning methods achieve different performance levels, with classical Perceptron being the weakest of the methods and Online SVMs being the strongest, the online active learning

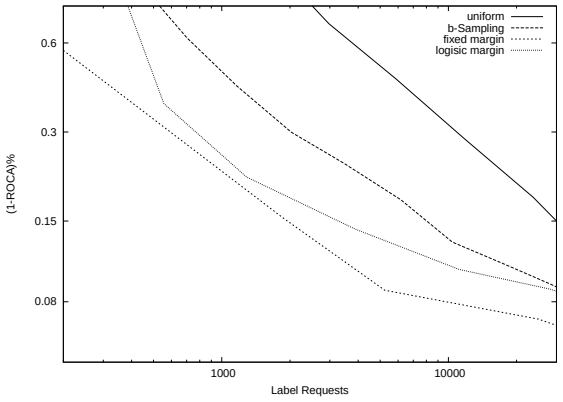

Figure 4: Classical Perceptron on trec05p-1. Results reported as (1-ROCA)%, by number of labels requested. Score using all labels is 0.071.

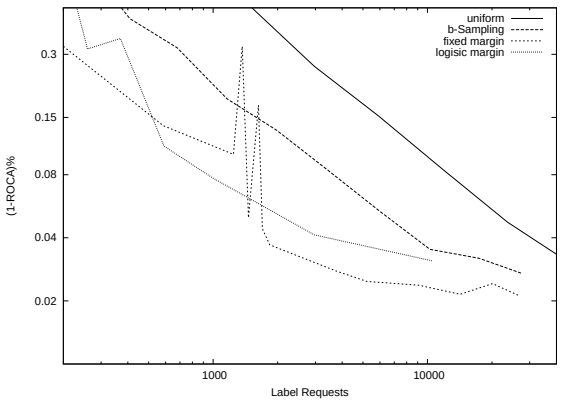

Figure 5: Perceptron with Margins on trec05p-1. Results reported as (1-ROCA)%, by number of labels requested. Score using all labels is 0.021.

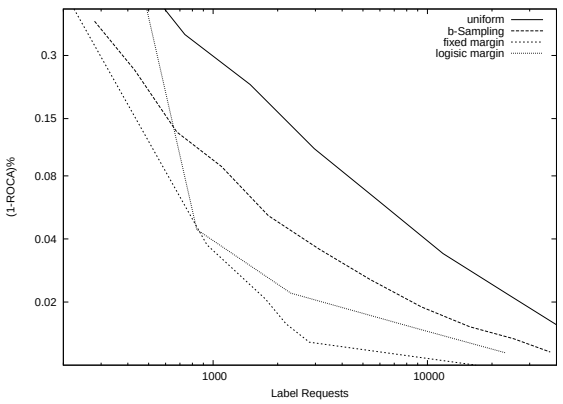

Figure 6: Online SVM with trec05p-1. Results reported as (1-ROCA)%, by number of labels requested. Score using all labels is 0.010.

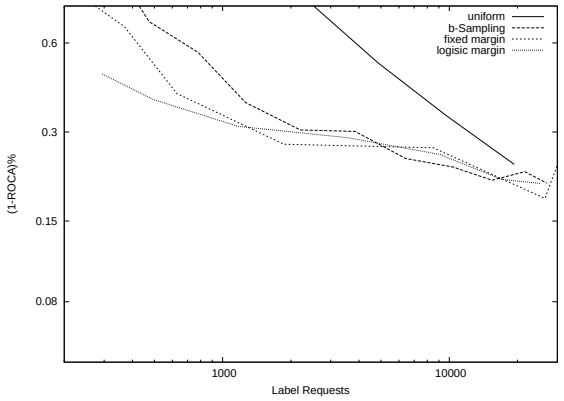

Figure 7: Classical Perceptron, trec06p Results reported as (1-ROCA)%, by number of labels requested. Score using all labels is 0.189.

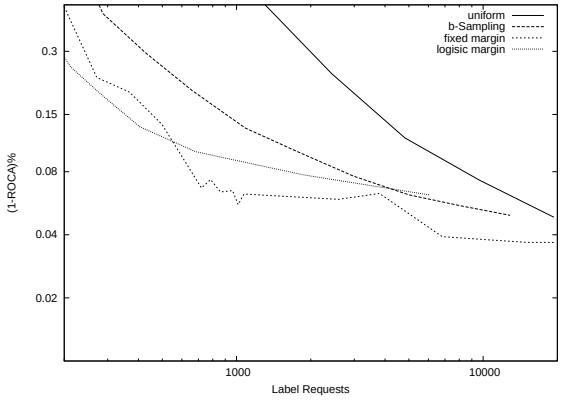

Figure 8: Perceptron with Margins on trec06p. Results are reported as (1-ROCA)% by number of labels requested. Score using all labels is 0.037.

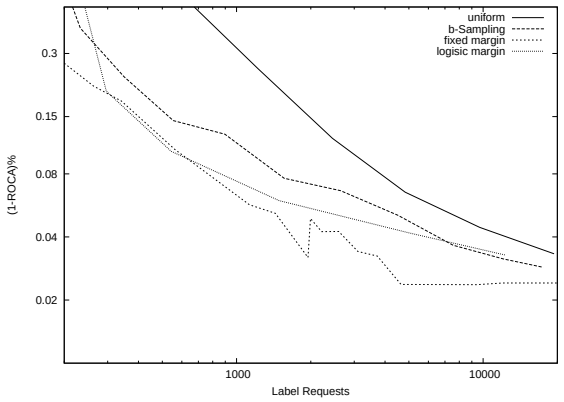

Figure 9: Online SVM on trec06p. Results reported as (1-ROCA)%, by number of labels requested. Score using all labels is 0.025.

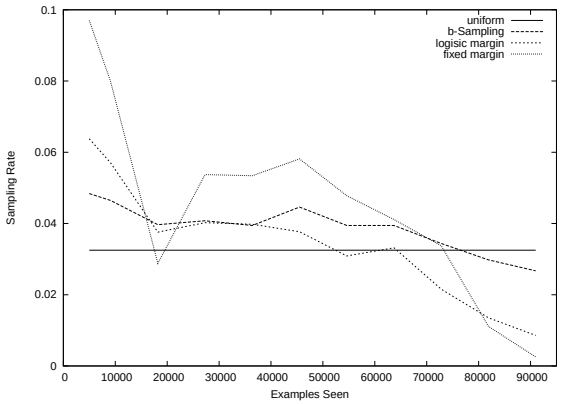

Figure 10: Perceptron with Margins, sampling rate over time, trec05p-1

methods give the same improvement over uniform random subsampling in all cases.

There are interesting comparisons among the online active learning methods, as well. Both Fixed Margin and Logistic Margin sampling tended to out-perform b-Sampling, often requiring a fraction of the labels needed by b-Sampling to achieve equivalent performance levels. This is most often true when the total number of labels requested is less than 10,000. With higher numbers of label requests, the performance of the different methods tends to converge. Note that the Fixed Margin sampling, which offers no theoretical guarantees, experiences some volatility between 1000 and 2000 label requests for the `trec05p-1` tests with Perceptron with Margins.

Finally, we find that the results reported here with online active learning methods for the `trec06p` data set are a dramatic improvement over the results reported for pool-based active learning methods tested in the 2006 TREC spam filtering competition [4]. For example, the best pool-based active learning method achieved a (1-ROCA)% score of roughly 0.3 with 10,000 label requests. The best online active learning methods achieved scores between 5 and 10 times better than this with the same number of label requests. (To our knowledge, pool-based active learning tests for the `trec05p-1` data set were not performed as part of this competition.)

Note that this is not strictly an apples-to-apples comparison: the 2006 TREC competition divided the `trec06p` data set into a pool of unlabeled examples from which examples selected (90% of the original data set), and a separate set of examples for testing (10% of the original data). However, it is still fair to compare the 2006 TREC results with our online active learning results because the 2006 TREC experimental setup is actually more favorable than the online active learning setup. In the TREC active learning experiments, the active filter was allowed to fully train on the given number of n labeled examples before testing and evaluation, while in the online case the filter was forced to classify new messages without the benefit of having seen all n labeled examples. The online active learning methods out-performed the pool-based methods with the same number of label requests, and did so at greatly reduced computational cost.

5.4 Online Sampling Rates

Aside from examining overall performance levels, it is useful to consider how the sampling rates for the online active learners changes over time. In Figure 10, the sampling rate for each active learning method is plotted against the number of examples seen, where parameter values for each method were set to each request roughly 3,000 labels during the entire set. (Results are shown for Perceptron with Margins on trec05p-1. Other results are similar.)

Over time, the number of labels requested by the active learning methods tends to decrease, with the Logistic Margin and Fixed Margin methods requesting labels for less than 1% of the examples by the end of the trial, compared to an overall sampling rate of 3.2% on these tests. The sampling rate of b-Sampling decreases steadily, but less slowly over time. Naturally, the sampling rate of uniform subsampling remains constant.

The decrease in sampling rate over time by the active learners is due to the fact that the quality of the hypothesis for each learner improves with additional labeled examples. This allows the learner to make more predictions with high confidence over time, reducing the number of label requests.

This observation is of practical value in large-scale email systems. Online active learning methods not only reduce the number of labeled examples needed to make a spam filtering system operable, but will also greatly reduce the number of labels needed to maintain strong classifiers over time.

6. CONCLUSIONS

We have proposed an online active learning framework for spam filtering, and have explored several reasonable approaches to determining when to request labels for new examples. We believe that online active learning is the most appropriate form of active learning for spam filtering. These methods give improved results over uniform subsampling and over prior pool-based active learning methods, reducing the number labels needed to achieve high performance with negligible computational cost. Furthermore, the online active learning approach is well suited to this domain, because spam filtering is an inherently online task.

Not only do online active learning methods reduce the number of email labels needed for strong performance, they also reduce computational cost of training. Training updates in the online active learning framework are only able to occur when a label request has been made. Because these methods require only a fraction of the total possible labels, training cost is necessarily reduced. This finding is of key importance for large email systems filtering millions or billions of messages each day.

These results have implications not only for the statistical side of spam filtering, but for user interface as well. Because online active learners require only few label requests, and the rate of label requests decreases over time, it is possible to envision an email system that asks a user to label perhaps one or two messages per day. Such a system would have strong filtering performance while requiring very little feedback from the user. Given the low cost and high performance of this approach, we recommend online active learning methods as a new, general strategy for real-world spam filtering.

7. REFERENCES

[1] M. Balcan and A. Blum. On a theory of learning with similarity functions. In *ICML '06: Proceedings of the Twenty-Third International Conference on Machine Learning*, pages 73–80, 2006.

[2] N. Cesa-Bianchi, C. Gentile, and L. Zaniboni. Worst-case analysis of selective sampling for linear classification. *Journal of Machine Learning Research*, 7:1205–1230, 2006.

[3] N. Cesa-Bianchi, G. Lugosi, and G. Stoltz. Minimizing regret with label efficient prediction. In *COLT*, pages 77–92, 2004.

[4] G. V. Cormack. TREC 2006 spam track overview. In *The Fifteenth Text REtrieval Conference (TREC 2006) Proceedings*, 2006.

[5] G. V. Cormack and T. R. Lynam. TREC 2005 spam track overview. In *The Fourteenth Text REtrieval Conference (TREC 2005) Proceedings*, 2005.

[6] S. Dasgupta, A. T. Kalai, and C. Monteleoni. Analysis of perceptron-based active learning. In *COLT*, pages 249–263, 2005.

[7] Y. Freund, H. S. Seung, E. Shamir, and N. Tishby. Selective sampling using the query by committee algorithm. *Machine Learning*, 28(2-3):133–168, 1997.

[8] C. Gentile. A new approximate maximal margin classification algorithm. *Journal of Machine Learning Research*, 2:213–242, 2001.

[9] D. Helmbold and S. Panizza. Some label efficient learning results. In *COLT '97: Proceedings of the tenth annual conference on Computational learning theory*, pages 218–230, 1997.

[10] T. Joachims. Training linear svms in linear time. In *KDD '06: Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 217–226, 2006.

[11] R. Khardon and G. Wachman. Noise tolerant variants of the perceptron algorithm. Technical report, Tufts University, 2005. To appear in Journal of Machine Learning Research; available at http://www.cs.tufts.edu/tr/techreps/TR-2005-8.

[12] W. Krauth and M. Mézard. Learning algorithms with optimal stability in neural networks. *Journal of Physics A*, 20(11):745–752, 1987.

[13] D. D. Lewis and W. A. Gale. A sequential algorithm for training text classifiers. In *SIGIR '94: Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 3–12, 1994.

[14] K. Li, K. Li, H. Huang, and S. Tian. Active learning with simplified svms for spam categorization. In *International Conference on Machine Learning and Cybernetics*, volume 3, pages 1198–1202, 2002.

[15] J. Platt. Probabilities for sv machines. In A. Smola, P. Bartlett, B. Scholkopf, and D. Schuurmans, editors, *Advances in Large Margin Classifiers*, pages 61–74. MIT Press, 1999.

[16] F. Rosenblatt. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, 65:386–407, 1958.

[17] N. Roy and A. McCallum. Toward optimal active learning through sampling estimation of error reduction. In *ICML '01: Proceedings of the Eighteenth*

International Conference on Machine Learning, pages 441–448, 2001.

[18] B. Schölkopf and A. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimizati on, and Beyond*. MIT Press, 2001.

[19] D. Sculley and G. Wachman. Relaxed online SVMs for spam filtering. In *The Thirtieth Annual ACM SIGIR Conference Proceedings*, 2007.

[20] D. Sculley, G. Wachman, and C. Brodley. Spam filtering using inexact string matching in explicit feature space with on-line linear classifiers. In *The Fifteenth Text REtrieval Conference (TREC 2006) Proceedings*, 2006.

[21] R. Segal, T. Markowitz, and W. Arnold. Fast uncertainty sampling for labeling large e-mail corpora. In *CEAS 2006: Conference on Email and Anti-Spam*, 2006.

[22] S. Tong and D. Koller. Support vector machine active learning with applications to text classification. *Journal of Machine Learning Research*, 2:45–66, 2002.