

Ideas and Applications on Support Vector Machine Active Learning

Xing Xie

December 16, 2009

Contents

1	Introduction	1
2	SVM Active Learning	2
3	Experiments and Applications	3
3.1	Text Classification	3
3.2	Spam Filtering	4
4	Conclusion	5

1 Introduction

For most algorithms we studied from our machine learning course CS545 [1], we choose training samples randomly from a large pool of labeled data, which means we know the sample classes in advance while constructing the training data set. While there is another option for selection training data: *pool-based active learning*, which is first introduced by Lewis and Gale in 1994 [5]. The learner can access to a pool of unlabeled (unclassified) data and then request the true class label for a certain number of instances, in order to form a good training data set. In many applications like web search, text classification, and email filtering, pool-based active learning is a reasonable approach since a large quantity of unlabeled data is readily accessed in the pool. So the efficiency of query is the main issue we concern for this algorithm: minimizing the number of labeled samples during the request for training data.

Tong and Koller gave out *Support Vector Machines* (SVM) active learning algorithms in 2001 [6]. The SVM [7] method has been used in applications like handwritten digit recognition and text classification with excellent empirical successes more than two decades. The SVM active learning method first uses SVM to create hyper-planes to bisect the unclassified training data based on the feature space, then select one specific sample to query for the real class label. They claimed the SVM active learning method can reduce the number of necessary labeled training samples to get a nice model. Many experiments and applications in these years have provided solid proofs on the efficiency of this method. In this paper, we mainly discuss the performance of active learning on two applications: text classification and email spam filter.

Since SVM active learning method spends considerable time to construct optimal the model by selecting best training samples , it may not be the appropriate approach for online tasks requires quick response and high training speed [8]. While some researchers propose additional technology support for SVM active learning in order to improve efficiency on specific online applications. We cover this issue with an application on email spam filtering.

The rest of this paper is organized into four sections. We first introduce basic concepts on SVM active learning in Section 2. Then we discuss the two applications explored by current research on text classification and email spam filter in Section 3. Finally, conclusion on this report on SVM active learning method is provided in Section 4.

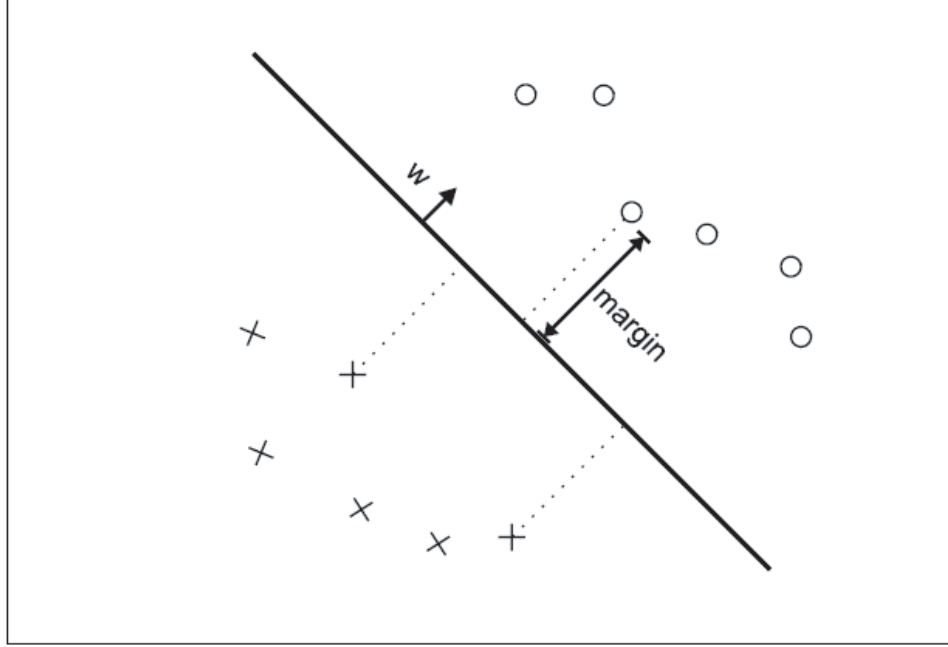


Figure 1: A simple linear support vector Machine

2 SVM Active Learning

In many cases, the data for classification have many features, which can be projected to multi-dimensions version space \mathbb{R}^d . So for some space $\mathbb{X} \subseteq \mathbb{R}^d$, all the data can be constructed as a vector like $\{x_1, \dots, x_n\}$. Then they consider SVMs in the binary classification setting by setting sample labels as $\{y_1, \dots, y_n\}$ where $y_i \in \{-1, 1\}$. SVMs are hyper-planes that separate the training data by a maximal margin. Vectors lying on one side of the hyperplane are labeled as -1, and all vectors lying on the other side are labeled as 1. The training instances that lie closest to the hyperplane are called support vectors. SVM allows one to project the original training data in space \mathbb{X} to a higher dimensional feature space \mathbb{F} . A simple SVM example is showed in 1:

SVM algorithm can learn a decision function in terms of $f(x) = w \cdot \phi(x)$, where $w = \sum_{i=1}^n \alpha_i y_i \phi(x_i)$, $\phi : X \rightarrow F$ a map from input space to some feature space, α s are coefficients of the maximal margin hyper-planes in F . Then a set of hyper-planes or hypophyses in the version space can be created. Hyper-plane H and version space V are defined as the following equations. Hyper-plane f is in the version space if for each training sample x_i with label y_i , $f(x_i) \leq 0$ if $y_i = -1$ and $f(x_i) \geq 0$ if $y_i = 1$.

$$H = \{f \mid f(x) = \frac{w \cdot \phi(x)}{\|w\|}\}$$

$$V = \{w \in F \mid \|w\| = 1, y_i(w \cdot \phi(x_i)) > 0, i = 1, \dots, n\}$$

We create two sets for the original data, U denotes the unlabeled data and L denotes the data with true label after query. then active learning begins in the following step:

1. The learner searches such an sample x from U for whose version space size is as equal as possible both if the sample is considered to be in the positive ($y = 1$) or the negative ($y = -1$) class.
2. The learner asks the user to provide the true label for the selected sample, and append it to L and the procedure is repeated for another sample.
3. The search stops when enough samples are collected depends on application requirements; In other words, these training samples should be able to construct a model with acceptable and stable performance.

How to pick the best sample is the main issue in active learning. Since an analytic solution is not feasible, three different approximations are proposed in their paper [6].

Simple Margin. Simple margin tests each of the unlabeled samples in the pool to see the distance between their corresponding hyper-planes and the central place w_i in F . The closer a hyperplane in F is to the point w_i , the more centrally it is placed in the version space, and the more it bisects the version space. Then the unlabeled one whose hyperplane in F comes closest to the vector w_i is selected.

MaxMin Margin. Each sample in the pool is separately added to both positive and negative class, after which the decision hyperplane is recalculated. m_i^+ and m_i^- represent the sizes of the corresponding SVM margins for the i th sample. Then we select the training sample by $\operatorname{argmax}_i \min(m_i^+, m_i^-)$. To be specific, for each unlabeled sample x_i , we can obtain the margins m_i^+ and m_i^- by labeling x_i as 1 and -1 respectively. After all samples are tested, we choose the one for which the quantity $\min(m_i^+, m_i^-)$ is greatest. Then means the choice for such sample would mostly bisect the version space.

Ratio Margin. Similar to maximizing $\min(m_i^+, m_i^-)$, it computes ratios of margin sizes $\min(\frac{m_i^+}{m_i^-}, \frac{m_i^-}{m_i^+})$. This method also halves version space as Simple and MaxMin margin.

3 Experiments and Applications

In this section, we discuss two applications with the active learning method. The first is text classification, we compare the active learning method with others in performance. The second is spam filtering, here we focus on the issue that how to improve original active learning method for better performance under the online requirements.

3.1 Text Classification

Novak et.al performed a text classification experiment in [2] for comparing the performance between SVM method and random sample. They run all of our experiments on the Reuters Corpus Vol. 1, 2002, which consists of about 810,000 news articles from 20 August 1996 to 19 August 1997, manually categorized into a shallow taxonomy of 103 categories. The following steps show how they prepare the data:

1. They selected training examples by splitting articles at 14 April 1997 into two sets, giving 504,468 articles for training and 302,323 articles for testing.
2. They removed the common words by using the standard English 523 stopword set. And they use Porter Stemming algorithm in order to simplify the data set: reducing the inflected or derived words to their term, root or base form.
3. All news articles were converted into TF-IDF vectors, which is a weight often used in information retrieval and text mining. This weight evaluates how important a word is to a document in a corpus.

They want to compare the performance among SVM Simple margin algorithm, random sample, and error reduction sampling algorithm by providing some query sample freedom on training sample selections. Error reduction sampling algorithm [9] basically chooses next sample while attempting to reduce future generalization error probability. Since true future error rates are unknown, the learner estimates them by a “self-confidence” heuristic for probability measurements. In the experiment, each of the three methods ran for 30 times, which use a random sub-sample of the data that consists of 5,000 training and 10,000 testing examples. For each run, one positive and one negative labeled sample were provided initially and then a total of 100 queries were made. The three algorithms had 200 randomly chosen examples from the unlabeled pool in each iteration for efficiency reasons. The size of the evaluation pool for the Error Reduction Sampling algorithm was also set to 200.

For the random sample algorithm, they selected next examples for “querying” just by random choice. They expected the random sample method to set a baseline for the performance of SVM methods.

The experiment result shows Simple margin always achieves the best choice and error reduction sampling has consistently a bit worse performance than Simple margin. A surprising result they found is one of the experiments show Error reduction sampling after 100 examples consistently performs worse than random sampling. This might implicate the performance of error reduction is not as stable as SVM Simple margin. So they recommended SVM method for large-scale text classification task in practice.

3.2 Spam Filtering

Spam filtering is another popular research topic in machine learning: the system picks a set of samples and query users to identify whether it is a spam or ham. After the training, the learner can classify the unlabeled data with some acceptable accuracy in future. So how to select the training data for optimal model plays an important role in spam filtering.

Intuitively, active learning is a nice solution for picking up training data. However, spam filter will not only use the existing data in the pool. There is an online feature on this issue: Since the behavior of emails changes continuously along the time, so learner should has a strong timely generalization and have a high processing speed to reduce training time.

Liu et.al developed an SVMEL (SVM Ensemble learning) method [3] to handle the email spam filtering with the online feature. SVMEL method aims to decrease the training time and increase the performance of the model by selecting the best training data. Figure 2 captured from [3] shows the SVMEL structure, and we describe it as follows:

1. Sample Selection and labeled decision by active learner are divided in two individual process. The result from final decision will affect the next sample selection by active learners.
2. the active learner first picks up the best unlabeled email based on previous knowledge in pool.
3. After the email sample is selected, it is first handled by several different spam detecting filters in paralle. These filters handle the email based on different concerns: for example, email text like spam keywords, and email behavior like receivers, sending times, etc. Each filter will return a spam confidence score (SCS) which is a real number between 0 and 1.
4. The SVMEL filter makes the final decision based on the a set of SCS values. To be more specific, let the object space of the SVM classifier is spam, ham, each training vector is $(SCS_1, SCS_2, , SCS_n)$ which SCS_i denotes the SCS output by the i th simple filter.
5. According to the final ensemble SCS of the email, active learner makes a decision. If the learner thinks the email can improve training performance then it actively queries user for a response and sends the response to each simple learners and ensemble one. This reinforcement behavior can refine the knowledge of those individual filters for next mail.
6. SCS can be improved more by comparing new email content with memorized labeled email one. But memorizing all labeled emails is impractical. So They use *cache* technology during the learning process, based on the assumption of the bulk feature for email spam: Similar spam emails are likely repeating in a certain of time. So they use a cache to memorize an amount of latest been labeled ones. Those cached emails are used for reinforcement learning in the filters.

Under the SVMEL structure, They performed an experiment on comparing SVMEL method and Bogofilter-0.93.4 system. The Bogofilter system implements a fast Bayesian spam filter to classify email as spam or ham by a statistical analysis of the message's header and content [4]. The Bogofilter system is well known by high performance on TREC06 spam track. TREC email data set is held by The Public Chinese corpus, which used data provided by the *CERNET Computer Emergency Response Team (CCERT)* at Tsinghua University, Beijing. The ham messages consisted of those send by to a mailing list; the spam messages were those sent to a spam trap in the same internet domain. Headers and bodies of spam messages were modified to make them appear to have been delivered to the same servers as the ham messages, in the same time interval. In this experiment, they use the TREC07 for comparison. The TREC07p corpus contains 75,419 messages: 25,220 ham emails and 50,199 spam emails.

They did 2 groups experiments and both are using the SVMEL and Bogofilter. For Group 1, is only 10,000 queries are allowed to make for both systems. For Group 2, 30,388 queries are allowed to make. First, the learning speed using active learning is faster than Bogofilter, especially from the beginning to the stable level. Besides, before 30,000 messages, the cache improvement is obvious, while after 30,000 messages the cache improvement is not so obvious because less repeated emails after 30,000 messages. Finally, the cache technique really help improve the pure active filter. In summary, The results show the active filter with cache is better than Bogoflter in performance on decreasing computation complexity, saving training time, resisting useless samples, and increasing filtering accuracy.

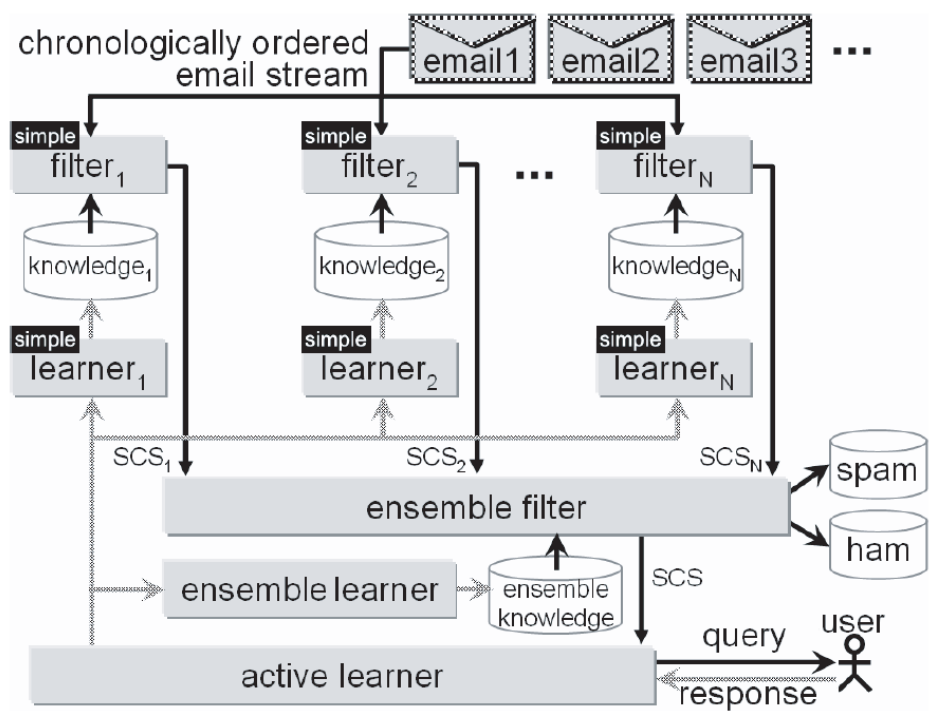


Figure 2: SVMEL Structure

4 Conclusion

From this report assignment, I have learned that SVM active learning method is a nice approach for selecting unlabeled samples as training data set. The model constructed by the selected samples give out better performance than random selecting method, in the cost of selection time. I think the reward is good if we have a large scale of data in the pool with unlabeled data is readily accessible. The model trained by SVM active learning is faster to become stable according to numerous experiments and applications.

On the other hand, the online tasks set obstacles for active learning with requirements like limited training time and undetermined future samples. Some technical tricks may be applied on specific tasks like cache for email spam filtering; but there is still no general solutions for active learning on this issue.

The most difficult part in this report is to understand the SVM active learning method. I have to refer to more previous paper for background reading. Also, summarizing the concepts in an easy understood way costs me lot of time. From this report, I've improved the learning skill for new research topics.

References

- [1] Anderson C., *CS545 Website*, <http://www.cs.colostate.edu/~anderson/cs545>, 2009.
- [2] Novak B., Mladenič D., Grobelnik M., *Text Classification with Active Learning*, Proceedings of Gesellschaft fuer Klassifikation (GfKI), 2005.
- [3] Liu W., Wang T., *Active Learning for Online Spam Filtering*, pp 555-560, Asia Information Retrieval Symposium (AIRS), 2008.
- [4] BGOFILTER, *Bogo Project*, <http://bogofilter.sourceforge.net/>.

- [5] Lewis D., Gale W., *A Sequential Algorithm for Training Text Classifiers*, In Proceedings of the Seventeenth Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval, pp 3-12. Springer-Verlag, 1994.
- [6] Tong S., Koller D., *Support Vector Machine Active Learning with Applications to Text Classification*, Journal Machine Learning Research, pp 999-1006, 2001.
- [7] Vapnik V., *Estimation of Dependences Based on Empirical Data*, Springer Verlag, 1982.
- [8] Monteleoni C., Kaariainen M., *Practical Online Active Learning for Classification*, In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Online Learning for Classification Workshop, (CVPR), 2007.
- [9] Roy, N., McCallum A., *Toward Optimal Active Learning Through Sampling Estimation of Error Reduction*, International Conference on Machine Learning, pp 441-448, 2001.