UB at TREC-11: Batch and Adaptive Filtering

M. Srikanth X. Wu R. Srihari

Department of Computer Science and Engineering
State University of New York at Buffalo,
Amherst, NY 14228-2567

1 Introduction

This is the first time we participated in TREC filtering track. We submitted four runs: two for adaptive filtering, and two for batching filtering. And these runs come from two separate efforts with very different approaches. One effort treats the filtering problems as standard text categorization problems and solves them using Support Vector Machines (SVM). The second effort is a Language Modeling approach to information filtering. Among other things we wanted to use filtering tasks as large scale test cases for two separate frameworks we have been working on for information retrieval. Significant time was spent on putting the components together and limited time on pre-submission performance evaluation.

2 Weighted Margin SVM for Information Filtering

2.1 SVM in text categorization

The standard text categorization problem can be stated as following: given a set of category-labeled documents, the goal is to classify a new document into the predefined categories. Typically each document can belong to multiple categories or no category at all. This is a supervised machine learning problem. And further more, when the categories form a flat structure, each category can be treated as a separated dichotomy problem.

SVM are based on Structural Risk Minimization (SRM) principle from statistical learning theory [7]. In contrast to the Empirical Risk Minimization (ERM) principle which try to find a hypothesis h from a structural complexity fixed hypothesis space, H, that minimizes the training error, the SRM try to find a hypothesis h where the true error of the classifier is minimized. To achieve this, SRM usually tries to shrink the complexity of the hypothesis space H while maintaining a fixed training error. Compare to ERM, SRM is more suited when the training data set is limited.

SVM, the simplest linear form of SRM, is nothing but a maximum margin linear classifier. Given an example (x_i, y_i), and decision hyperplane (w, b), the (functional) margin of example with respect to hyperplane is defined as

$$\gamma_i = y_i(< w \cdot x_i > +b). \tag{1}$$

Note γ_i implies correct classification of (x_i, y_i) if $\gamma_i > 0$. The (functional) margin of training set S (with l examples) with respect to decision hyperplane (w, b) is defined as

$$\gamma = \min_{0 \le i < l} y_i(< w \cdot x_i > +b), \tag{2}$$

Geometric margin is the functional margin derived by $\|w\|$. The maximum margin hyperplane given training set S is thus defined as the hyperplane with respect to which the training set has maximum geometric margin.

There are two major advantages of SVM. First the learning ability of SVM is independent of the dimensionality of the feature space thus immune from the so-called *curse of dimensionality*. SVM learning process typically focuses on these hard to classify patterns automatically and thus can ignore the extra noise introduced by each additional dimension. Thus by using SVM, the usually computational expensive feature selection steps are not needed. Second, with so-called *kernel tricks*, SVM can be used to learn different discriminant functions by using different kernel functions. So it can learn a linear classifier, polynomial classifier, or radial basis function with just one line change in your source code.

SVM has been used for text categorization [3, 2, 5] and shown to outperform all classical learning methods including neural networks, linear discriminant function and KNN. And this claim is further verified by Y. Yang in her well-cited comparison papers [9, 8]. The reason for its superior performance can be best explained by its compatibility to text categorization problems: the typical document representation method like term frequency and inverted document frequency (TFIDF) weighted vectors results in huge and sparse vector for each document and most text categorization problems are linearly separable (as suggested by the fact that all ohsumed categories and most Reuters categories are linearly separable).

2.2 Filtering as a Text Categorization Problem

While last year batch filtering track can be easily cast into a text categorization problem, it is not possible this year. For one, we don't have enough training examples available for most all the topics, and it is even worse in adaptive filtering track where we have only three positive examples for each topic. Second, the rich information contained in the topic description/narrative that help to shape topic boundary is not readily available for any statistical based machine learning methods. And actually it is our guess that the ability of approaches to make use of the topic description/narrative will differentiate their performance. In the final result, the fact that people can do much better in first 50 categories than in the second 50 categories can be considered as a manifestation of our guess. Since the quality of topic description/narrative is much better in first 50 categories is much better than these of the second 50 categories.

SVM is the winner of the last year batch filtering track. And it is considered as one of major component of our package, to really test out our SVM implementation, we use it to handle the filtering track problem. To do that, there are two problem we have to deal with. First, we have to turn the information stored in the topic description/narratives into some usable information for our SVM learner. Since the only information that SVM learner can make use of is the examples, we have to device a way to generate pseudo examples using topic description/narrative. Second, how our SVM learner make use of these pseudo examples?

2.3 Generating Pseudo Examples

The way we generate pseudo examples from the topic description and narrative is very simple. And there are two different steps. First, we use the description/narrative get the top 30 closest document in TFIDF sense, and label them as probable positive example. Note that they definitely can't be considered as 100% positive examples. Second, we randomly choose 90 documents from the whole training set excluding the existing training examples and top 1000 closest documents to topic description/narrative in TFIDF sense and label them as probable negative examples. This

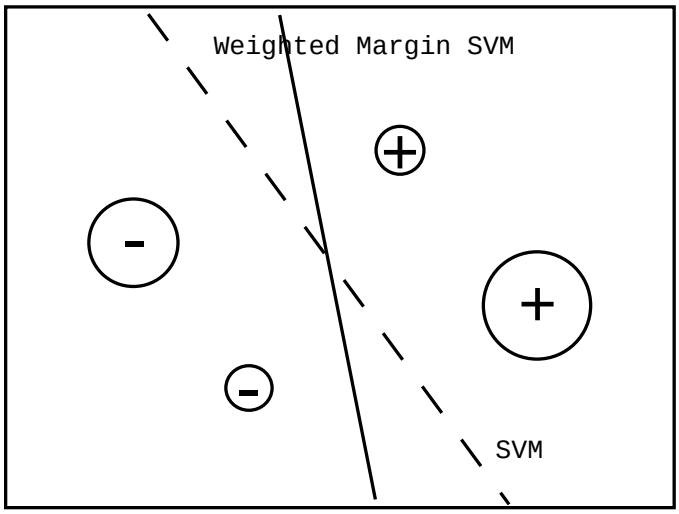

Figure 1: SVM vs Weighted Margin SVM

is extreme simple way to utilize the rich information contained in the topic description/narrative. And the number of probable positive and negative examples we use is set arbitrarily for all the topic without further investigation.

2.4 Weighted Margin SVM

With newly generated probable examples, the question now is how to use them in SVM learning. To be able to use the training set where label is associated with an observation weight, the SVM training and classification algorithms has been modified to handle these additional information available to the learner: and the result of such modification is the Weighted Margin SVM (WMSVM) machine.

The difficulty a classical SVM classifier faces when presented with weighted observation can be best illustrated by the figure 1. We have four pieces of labeled data: two positive and two negative. The size of the circle represents the label reliability. The dashed line represents the hyperplane found by SVM, and the solid line the hyperplane by the Weighted Margin SVM.

In the results submitted for the filtering track, we were using a weaker version of the WMSVM implementation that can handle the weighted soft margin. Since then, we have a stronger WMSVM implementation that can handle both hard margin and soft margin correctly, but we are still working the experiments with this strong version on filtering track. But even with the weaker version WMSVM, along with the simple pseudo example generation, we are able to apply SVM learning on a data set which for some cases, has only a few positive examples.

3 Language Modeling Approach

A model-based approach to information filtering was explored for the second set of submissions to TREC 2002 batch and adaptive filtering tasks. Language models are associated with both documents and queries. The initial query model was generated using the topic description. We viewed the training documents for a given query as relevance feedback documents. A new query model is estimated based on the initial query model and the language models estimated for the feedback documents. This method was used for both batch and adaptive filtering.

3.1 Language Modeling and Information Retrieval

Statistical Language Modeling (SLM) has been used in many Natural Language Processing (NLP) tasks including Speech Recognition, Machine Learning and Information Extraction. Recently, Ponte and Croft [6] proposed language modeling approach to information retrieval. Each document in a document collection is associated with a language model and given a query, documents are ranked based on the probability of their language model generating the query text. Alternatives to query likelihood model have been proposed. Of specific interest here is the method proposed by Lafferty and Zhai [4] where they associate language models to both documents and queries and rank documents based on their model's similarity with the query model. Model similarity was computed using the Kullback-Leibler divergence measure.

The motivation for our language modeling approach to TREC-11 batch and adaptive filtering is Zhai and Lafferty's paper on incorporating relevance feedback in language modeling approaches to information retrieval [10]. Their proposal was in the Query-Document Model similarity approach to information retrieval Given a set of feedback documents $F = \{d_1, d_2, \cdots, d_n\}$ for query Q, they estimate a feedback model $\hat{\theta}_F$ based on the feedback document F and use it to update the query model $\hat{\theta}_Q$ to $\hat{\theta}_{Q'}$ by

$$\hat{\theta}_{Q'} = (1 - \alpha)\ \hat{\theta}_Q + \alpha\ \hat{\theta}_F \tag{3}$$

where α is the interpolation parameter. Two different strategies were proposed for feedback model estimation: a generative model of feedback documents and a model with minimum divergence over feedback documents. We used the later in our language modeling approach to information filtering. Zhang and Callan [11] used a method similar to the generative model for feedback documents in their TREC 2001 adaptive filtering submission. They used language modeling techniques in updating terms and term weights in their query representation.

In the divergence minimization approach, the feedback model is estimated to satisfy two conditions: (1) that it is "closer" to the feedback documents and (2) it is "farther" from the corpus model. The second condition ensures that the effect of language and domain characteristics common to feedback documents do not generalize the new query model and move it off topic. The feedback model is selected to be the one which minimizes

$$D_e(\theta; F, C) = \frac{1}{|F|} \sum_{i=1}^{n} D(\theta || \hat{\theta}_{d_i}) - \lambda\ D(\theta || p(\cdot|C)) \tag{4}$$

where $p(\cdot|C)$ is the corpus probability distribution and λ is the feedback parameter.

3.2 Language Modeling approach to Information Filtering

Given a query Q, two language models are estimated: (1) positive or on-topic language model, $\hat{\theta}_P$, and (2) negative or off-topic language model, $\hat{\theta}_N$. The initial positive and negative models are estimated from the topic description. These models are updated based on the training data available for each query. The positive examples $F_p = \{d_{p_1}, d_{p_2}, \cdots, d_{p_{|F_p|}}\}$ are used to update the positive model $\hat{\theta}_P$ using the feedback model generated by minimizing (5) which is similar to (4).

$$D_n(\theta_{F_p}; F_p, C) = \frac{1}{|F_p|} \sum_{d \in F_p} D(\theta_{F_p} || \hat{\theta}_d) - \lambda\ D(\theta_{F_p} || p(\cdot|C) \tag{5}$$

While the negative language model $\hat{\theta}_N$ can be used instead of the corpus probabilities in (4), we have used the corpus model since it is a better representation of what is not in topic. A negative

feedback model is generated by minimizing

$$D_n(\theta_{F_n}; F_n, C) = \frac{1}{|F_n|} \sum_{d \in F_n} D(\theta_{F_n} || \hat{\theta}_d) - \lambda \; D(\theta_{F_n} || \hat{\theta}_P) \tag{6}$$

where $F_n = \{d_{n_1}, d_{n_2}, \cdots, d_{n_{|F_n|}}\}$ is the set of negative examples. Here one is interested in a negative feedback model that is "closer" to negative examples by "farther" from positive language model. Unlike [10] who used a Dirichlet smoothing in estimating document language model, $\hat{\theta}_d$, we used a mixture model with fixed weights for document and corpus statistics.

$$p(w|\hat{\theta}_d) = \gamma \; p(w|d) + (1 - \gamma) \; p(w|C) \tag{7}$$

where γ was set to 0.6.

The positive and negative topic models are updated by

$$\hat{\theta}_{P'} = (1 - \alpha_1) \; \hat{\theta}_P + \alpha_1 \; \theta_{F_p} \tag{8}$$

$$\hat{\theta}_{N'} = (1 - \alpha_2) \; \hat{\theta}_N + \alpha_2 \; \theta_{F_n}. \tag{9}$$

Given a test document, its language model is first estimated using (7). Its score is determined by the ratio of its divergence from positive and negative models

$$score(\hat{\theta}_d; \hat{\theta}_P, \hat{\theta}_N) = D(\hat{\theta}_d || \hat{\theta}_P) / D(\hat{\theta}_d || \hat{\theta}_N). \tag{10}$$

Document scores are thresholded to make the binary classification decision. Thresholds were estimated based on score distribution in the training set similar to the method used by [1]. The score of relevant documents are assumed to be normally distributed and the top non-relevant documents are exponentially distributed. The utility score is optimized to obtain a closed form solution for the threshold.

For adaptive filtering, there are no negative examples and hence the initial negative model is the corpus model. The above method is followed to classify documents. When a document is deemed relevant for a query by the system, its relevance judgment is fetched to update the language models. If the document was judged relevant to the topic, the positive model is updated and if it was deemed not relevant the negative model was updated. While the models can be updated irrespective of the relevance of the document, for our TREC submission, we only updated one model at a time. The score threshold is updated before moving on to the next document.

Some implementation specific details and observation on our system's performance are given here. In our implementation,

- Document and queries were stemmed and stop words were removed.
- Only the topic description was used in generating the initial topic model generation
- While computing the score, terms with probability less than 0.0001 were ignored.
- Instead of top non-relevant document scores, all non-relevant document scores were used in computing the threshold.
- In adaptive filtering, the corpus statistics were not updated as test documents are processed. The training document collection was used to estimate the corpus model.
- In adaptive filtering, documents whose relevance is not known is assumed to be not relevant to the topic and is used in updating the negative topic model.

4 Observations and Conclusion

The results we submitted to filtering track using either methods is not impressive. There are couple reasons for that. With regards to our Weighted SVM submission, besides the fact we didn't use a stronger version WMSVM, the naive way of making use of topic description/narrative probably killed us. This is partially supported by the different performance difference between us and best performance for topic: in the first 50 topic where description/narrative is rich in content, we lag far behind the best performer. But on the second 50 topic, where the topic description/narrative is not as good, we are a little bit closer to the best performer overall.

With regards to the language modeling approach, our initial analysis suggests that the thresholding technique used seems to favor high recall taking our system closer to an "allow-all" classifier. This could have been due to the characteristics of the measure we used for scoring documents. Using all non-relevant documents in our threshold computation seems to have affected the thresholding processes. This was compounded by the assumption of documents with unknown relevance as non-relevant.

Either methods, we believe, have scope for further improvement with respect to their application in information filtering. We expect the stronger version of WMSVM to perform better. In addition we are exploring better ways to generate pseudo examples from topic description/narratives. At same time, a couple parameter, such as the observation weight for each pseudo examples and the number of pseudo examples, can be tuned to improve filtering performance.

References

[1] A. Arampatzis, J. Beney, C Koster, and T van der Weide. Incrementality, half-life, and threshold optimzation for adaptive document filtering. In E. M. Voorhees and D. K. Harman, editors, *TREC 2000*, Gaithersburg, MD, 2000.

[2] R. Cooley. Classification of news stories using support vector machines. In *Proceedings of IJCAI'99 Workshop on Text Mining*, Stockholm, Sweden, 1999.

[3] Thorsten Joachims. Text categorization with support vector machines: learning with many relevant features. In Claire Nédellec and Céline Rouveirol, editors, *Proceedings of ECML-98, 10th European Conference on Machine Learning*, pages 137–142, Chemnitz, DE, 1998. Springer Verlag, Heidelberg, DE.

[4] J. Lafferty and C. Zhai. Document language models, query models, and risk minimization for information retrieval. In *SIGIR01*, pages 111–119, 2001.

[5] David Lewis. Applying support vector machines to the trec-2001 batch filtering and routing tasks. In *NIST Special Publication 500-250: The Tenth Text REtrieval Conference*, pages 286–292, 2001.

[6] J. M. Ponte and W. B. Croft. A language modeling approach to information retrieval. In *SIGIR98*, pages 275–281. ACM, New York, 1998.

[7] Vladimir N. Vapnik. *The nature of statistical learning theory, 2nd Edition.* Springer Verlag, Heidelberg, DE, 1999.

[8] Y. Yang and X. Liu. A re-examination of text categorization methods. In *22nd Annual International SIGIR*, pages 42–49, Berkley, August 1999.

[9] Yiming Yang. An evaluation of statistical approaches to text categorization. *Information Retrieval*, 1(1-2):69–90, 1999.

[10] C. Zhai and J. Lafferty. Model-based feedback in the language modeling approach to information retrieval. In *CIKM*, pages 403–410, 2001.

[11] Y. Zhang and J. Callan. The bias problem and language models in adaptive filtering. In E. M. Voorhees and D. K. Harman, editors, *TREC 2001*, pages 78–83, Gaithersburg, MD, 2001.