Less is More: Active Learning with Support Vector Machines

Greg Schohn GCS@JUSTRESEARCH.COM
David Cohn COHN@JUSTRESEARCH.COM
Just Research, 4616 Henry Street, Pittsburgh, PA 15213 USA

Abstract

We describe a simple active learning heuristic which greatly enhances the generalization behavior of support vector machines (SVMs) on several practical document classification tasks. We observe a number of benefits, the most surprising of which is that a SVM trained on a well-chosen subset of the available corpus frequently performs better than one trained on *all* available data. The heuristic for choosing this subset is simple to compute, and makes no use of information about the test set. Given that the training time of SVMs depends heavily on the training set size, our heuristic not only offers better performance with fewer data, it frequently does so in less time than the naive approach of training on all available data.

1. Introduction

There are many uses for a good document classifier — sorting mail into mailboxes, filtering spam or routing news articles. The problem is that *learning* to classify documents requires manually labelling more documents than a typical user can tolerate. This makes it an obvious target for active learning, where we can let the system ask for labels only on the documents which will most help the classifier learn.[1]

In this paper, we describe the application of active learning to a support vector machine (SVM) document classifier. Although one can define an "optimal" (but greedy) active learner for SVMs, it is computationally impractical to implement. Instead, we use the simple, computationally efficient heuristic of labeling examples that lie closest to the SVM's dividing hyperplane. Testing this heuristic on several domains, we observe a number of results, some of which are quite surprising. Compared with a SVM trained on randomly selected examples, the active learning heuristic provides significantly better generalization performance for a given number of training examples.

[1] See Tong and Koller (2000) in this volume for parallel research on this topic.

When we train the SVM on a small subset of the available data chosen by our heuristic, its performance is frequently *greater* than the performance of an SVM trained on all available data. As such, it provides a win on two fronts: it provides better generalization and requires using fewer data than a passive learner trained on the entire data set. Lewis and Gale (1994) observed similar phenomena with their probabilistic classifiers. In the case of SVMs however, there is an added bonus: given the superlinear training cost, the smaller training set frequently yields this improved performance using less computation *time* than needed to achieve this performance from random examples. A final contribution of this paper is the identification of a heuristic stopping criterion that appears to reliably indicate when peak generalization performance has been reached.

In the remainder of this section, we describe and motivate the process of active learning, and briefly review our target learning architecture, the support vector machine.

Section 2 discusses several active learning heuristics for support vector machines. Sections 3 and 4 describe a series of experiments and results with one of those heuristics. We consider the implications of our results in Section 5.

1.1 Active Learning

Formally, active learning studies the closed-loop phenomenon of a learner selecting actions or making queries that influence what data are added to its training set. Examples include selecting joint angles or torques to learn the kinematics or dynamics of a robot arm, selecting locations for sensor measurements to identify and locate buried hazardous wastes, or querying a human expert to label a new document in a document classification problem.

The primary motivation for active learning comes from the time or expense of obtaining labeled training examples. In some domains, such as industrial process modeling, a single training example may require several days and cost thousands of dollars to generate. In other domains, such as learning to classify and filter email, obtaining examples is not expensive, but may require the user to spend hours of tedium labeling them.

The promise of active learning is this: when the examples to be labeled are selected properly, the data requirements for some problems decrease drastically. In special cases, even the computational requirements decrease, and some NP-complete learning problems become polynomial in computation time (Angluin, 1988; Baum & Lang, 1991).

In this paper, we will focus on a form of active learning called *selective sampling*. In selective sampling, the learner is presented with a large corpus of unlabeled examples, and is given the option of labeling some subset of them. Since each label "costs" us something, we wish to choose a small subset that maximizes our classification accuracy on the entire corpus. In our case, this corpus is a collection of documents, which we wish to classify, perhaps into a Yahoo-like hierarchy. The architecture which we will apply to this problem is the Support Vector Machine.

1.2 Support Vector Machines

Given a domain X, a linear support vector machine (Schölkopf et al., 1999) is defined in terms of the hyperplane

$$w \cdot x + b = 0 \tag{1}$$

corresponding to the decision function

$$f(x) = sgn\left(w \cdot x + b\right), \tag{2}$$

for $w \in \Re^N$ and $b \in \Re$. Given a set of labeled data $D = \{(x_1, y_1), (x_2, y_2), \ldots, (x_m, y_m)\}$, where $x_i \in \Re^N$ and $y_i \in \{-1, +1\}$, the optimal hyperplane is the unique hyperplane that separates positive and negative examples for which the margin is maximized:[2]

$$\max_{w,b} \left\{ \min_{x_i} \left\{ ||x - x_i|| : x \in \Re^N, w \cdot x + b = 0 \right\} \right\}. \tag{3}$$

When the data are not separable, a *soft margin* classifier is used. This introduces a misclassification cost C, which is assigned to each misclassified training example.

Equation 3 is usually optimized by introducing Lagrange multipliers α_i and recasting the problem in terms of its Wolfe dual:

$$\text{maximize:} \quad L_D = \sum_i \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j x_i x_j \tag{4}$$

$$\text{subject to:} \quad 0 \leq \alpha_i \leq C, \text{ and } \sum_{i=0} \alpha_i y_i = 0 \tag{5}$$

The x_i for which α_i are non-zero have a special meaning. They are the training examples which fall *on* the margin, and thus limit the position of the optimal hyperplane. These x_i are the *support vectors*. The x_i for which $\alpha_i = C$ also have special meaning — these are *bound* examples, examples which are incorrectly classified or are within the margin of the hyperplane.

Support vector machines have demonstrated excellent performance in many domains, particularly those involving text classification (Joachims, 1998b; Dumais et al., 1998). Recent advances (Joachims, 1998a; Platt, 1998) have also sped up the optimization problem such that it is practical to solve support vector problems involving tens of thousands of documents in a reasonable amount of time.

The complexity of finding the optimal hyperplane and its support vectors involves a form of quadratic programming and, as such, is NP-complete in the worst case (Vavasis, 1991), with typical running times of $\Omega(m^2)$ in the size of the training set. Given the superlinear time dependence on the number of training examples, as well as the cost of obtaining labels for the examples in the first place, it is reasonable to try to minimize the number of labeled examples needed to achieve good performance.

2. Active Learning for Support Vector Machines

Learners traditionally attempt to minimize error on future data. In a probabilistic framework, an active learner can do this by estimating expected future error, then selecting training examples that are expected to minimize this expected future error. SVMs, however, are a discriminant classifier; only recently have attempts been made to provide probabilistic estimates of a label's confidence.

In this section, we describe two active learning criteria for selecting data. The first is a probabilistically "correct," but impractical approach, while the second is a simple, efficient heuristic inspired by the first. We find empirically that the heuristic achieves remarkable performance at little cost.

2.1 A Greedy Optimal Strategy

Platt (1999) describes an intuitive means of assigning probabilities to points in the space classified by a support vector machine: project all examples onto an axis perpendicular to the dividing hyperplane, and perform logistic regression on them to extract class probabilities. By integrating the probability of error over the volume of the space, weighted by some assumed distributions of test examples, we can estimate the expected error of the classifier.[3]

[2] This notion of "optimality" is not directly tied to the performance of the classifier. There is, however, evidence that maximizing the margin acts as a form of structural risk minimization (Vapnik, 1998).

[3] This integration would probably be done via Monte Carlo methods, on a random sample of unlabeled examples sampled or generated according to the test distribution. See Cohn et al. (1996) for details.

From there, it is straightforward to compute the expected effect of adding an arbitrary unlabeled example x:

1. Use logisic regression to compute class probability $P(y = 1|x)$ and $P(y = -1|x)$.
2. Add $(x, 1)$ to the training set, retrain, and compute the new expected error $E_{(x,1)}$.
3. Remove $(x, 1)$, add $(x, -1)$ to the training set, retrain, and compute the new expected error $E_{(x,-1)}$
4. Estimate expected error after labeling and adding example x as
$E_x = P(y = 1|x) \cdot E_{(x,1)} + P(y = -1|x) \cdot E_{(x,-1)}$.

The learner can then consider all available candidate points and select from them the one that minimizes E_x.

A non-probabilistic analogue of this approach would use a "best worst-case" model: Define $E_{(x,1)}$ and $E_{(x,-1)}$ as the volume spanned by the margin. Then define $E_x = \max(E_{(x,1)}, E_{(x,-1)})$, providing a lower bound on the decrease in uncertainty that labelling and adding a training example will produce.

While this is arguably the best one can do in a greedy active learning setting, both algorithms are impractical. Evaluating each candidate point requires solving two QP problems; in a domain with thousands of available candidates, even a single complete evaluation of each is expensive.

2.2 A Simple Heuristic

What we would like is a heuristic which will estimate the expected change in error from adding an example without requiring us to actually add the example and recompute the resulting SVM. We investigate this by considering the ways a new example may change the SVM to which it is added.

SVMs have shown the greatest promise in high-dimensional domains such as text classification, where the number of problem dimensions may be an order of magnitude larger than the number of examples. In these cases, the subspace spanned by a given a set of training examples will cover only a fraction of the available dimensions. One heuristic active learning criterion would be to search for examples that are orthogonal to the space spanned the the current training set, effectively giving the learner information about dimensions it has not yet explored.

An alternative approach is to try to improve confidence in dimensions about which we already have information. We can accomplish this by attempting to maximally narrow the existing margin. If we assume the "best worst-case" model, examples that lie along the dividing hyperplane will on average divide the space up most quickly. Note that a point's location on the hyperplane with have a large effect on how labeling it influences the hyperplane (see Figure 1). But labeling an example that lies on or close to the hyperplane is guaranteed to have an effect on the solution, and appears to be a simple and effective form of divide and conquer.[4]

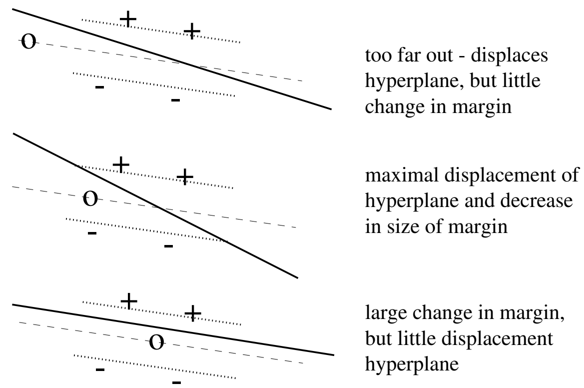

Figure 1. The location of an example on the dividing hyperplane affects its displacement. The dashed line represents the old hyperplane, dotted lines are the old margin; 'o' denotes the example being queried, and the solid line denotes the new hyperplane, given a labeling of '-'.

Selecting training examples by their proximity to the dividing hyperplane is computationally inexpensive: if we explicitly compute the dividing hyperplane, evaluating each candidate requires only a single dot product computation. As we discover in the next section, this criterion is also surprisingly effective.

3. Experiments

In this section, we evaluate the effectiveness of selecting training examples according to their proximity to the dividing hyperplane for a "linear" SVM. We describe experiments in two text classification domains, and compare the performance of our heuristic with that of training on randomly selected examples.

3.1 Document Classification and the Vector Space Model

One of the most successful approaches to document classification is the *vector space model*, which ignores word location and context, and treats the document as an unordered "bag of words." The space of the model has one dimension for each word in the corpus vocabulary, and the coordinate of an example in that dimension is the number of times the corresponding word appears in the document. Frequently a list of "stopwords" — common but non-content-bearing words such as "the," "if," and "it" are removed.

The vector space model can lead to a domain with more

[4] See Baum and Lang (1991) for another example of applying divide and conquer to identify hyperplanes in an active learning setting.

than 100,000 dimensions, making many traditional machine learning approaches infeasible. The model however, has been shown to work very well with naive Bayes classifiers (Nigam et al., 1998) and SVMs (Joachims, 1998b).

3.2 Experimental setup

We ran experiments on two text domains: binary classification of four newsgroup pairs from the "20 Newsgroups" data set (Nigam et al., 1998), and topic classification on a subset of five topics from Reuters news articles (Lewis, 1997). Each document was normalized for document length, but no other weighting (such as TFIDF) was performed on the vectors.

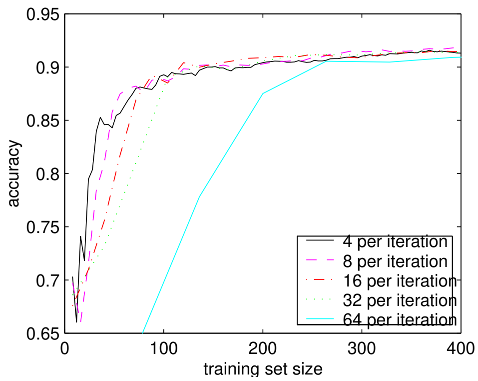

Figure 2. Effect of sample granularity of active learning for the "earn" Reuters group. There is a large initial difference, but little asymptotic effect from the number of training examples added on each iteration.

For each class of two domains, we performed five runs with independently-drawn 50%/50% test/training splits for both the heuristic and the default strategy of adding example labels at random. Each experiment began with four randomly chosen positive examples and four randomly chosen negative examples. Successive examples were added in batches of size b, chosen from the training set either randomly or in order of proximity to the current dividing hyperplane. We performed runs for values of $b = 4, 8, 16$ and 32 new examples per iteration. As expected, finer sampling granularity leads to a sharper increase in performance for small training set sizes (Figure 2). Computationally, this increase must be traded off against the cost of re-solving a new QP problem more frequently. The optimal time-vs.-label cost tradeoff depends heavily on the domain; for simplicity, all experiments discussed for the rest of the paper use $b = 8$.

We used a QP solver based on Joachims (1998a) to train the SVMs at each iteration for the USENET data. The solver used the working set strategy (Osuna et al., 1997) with four elements and the PR_LOQO solver (Smola, 1998) without shrinking to solve each QP sub-problem. We used a version of Platt's SMO algorithm (Keerthi et al., 1999) to train the SVMs for the Reuters data. Both algorithms produce comparable results on all data sets; the different methods were chosen in the interest of computational efficiency.

3.3 USENET – "20 Newsgroups"

We chose four pairs of newsgroups from the 20 Newsgroups data set, selected to span a wide range of perceived difficulties:

alt.atheism vs. talk.religion.misc	difficult
comp.graphics vs. comp.windows.x	moderately difficult
comp.os.ms-windows.misc vs. comp.sys.ibm.pc.hardware	moderately easy
rec.sports.baseball vs. sci.cryptography	easy

The goal was to learn the correct newsgroup that a message belongs to, analogous to an email filter. The 20 newsgroups dataset does include a small number of cross-posted articles, which makes some datasets inseparable under any domain representation. All of the newsgroups have approximately the same size (1000 documents). Articles with uuencoded content were ignored and headers were removed before classification.

3.4 Reuters

The second data set involved articles on five topics from Reuters news articles. These articles were taken from the Reuters-21578 Distribution 1.0 dataset (Lewis, 1997). The dataset contains 21,578 Reuters newswire articles from 1987. Of those, the subset of 10,711 articles that have topic labels were used. On each run, the articles were randomly divided into equally sized test and train sets. We chose the five topics with the largest number of "positive" examples: **acquisitions**, **earn**, **grain**, **crude oil**, and **money-fx**. The remainder of the 10,711 documents not bearing the selected topic were labeled negative. The earn set had 3802 "positive" documents (those with the earn label), acq had 2327, money-fx had 743, crude had 592 and grain had 589.

4. Results

4.1 USENET

In Figure 3 we plot test accuracy of the active learner and random sampler as a function of training set size. The performance of the active learner is slightly, but consistently better than that of the random sample, and all learners reach their asymptotes after relatively few documents. It is worth noting that in these experiments, the number of positive and negative examples (in both training and test sets) are approximately equal. A relatively large number of the training examples (40%-70%) become support vectors in the final model, leaving relatively few insignificant examples.

It should also be noted that the USENET dataset contain crosspostings; approximately 20% of the atheism/religion newsgroup articles are posted to both newsgroups, limiting a perfect learner to 90% accuracy. The atheism/religion newsgroup exhibits a slight downward slope after half the documents have been inspected. In this case, the dip may be due to an increasing number of conflicts between labels of the crossposted articles, we will encounter the phenomenon again with the Reuters groups, which contains no conflicting labels.

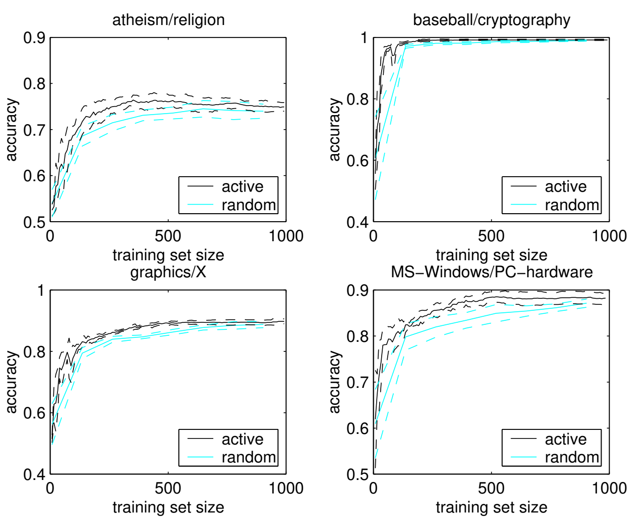

Figure 3. Accuracy on pairwise newsgroup classification is improved slightly by the active learning heuristic. All learners reach their asymptotes after relatively few training examples. Lines indicate mean accuracy over five runs; dashed lines indicate standard error.

4.2 Reuters

The difference between active learning and random sampling is much more pronounced on all of the Reuters sets (Figures 4–8). The accuracies reported are the macro-averaged[5] accuracies of both classes. The accuracy of the active learner reaches a maximum long before that of the random learner, which reaches its maximum using all of the documents.

It is worth noting that the active learner's performance is strongest over other methods when the split between categories is most uneven, where the smaller class can

[5]The macro-average (Yang, 1999) is the average over each class instead of each document. In other words, the average of each classes accuracy.

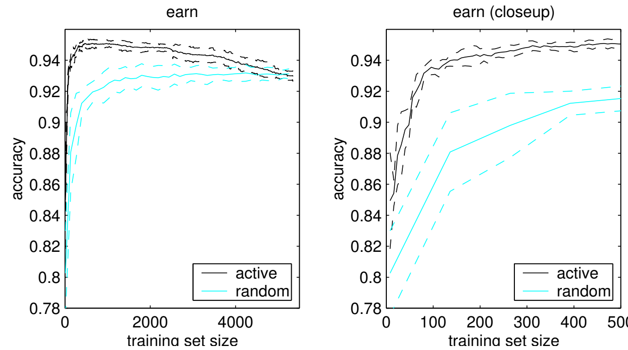

Figure 4. Accuracy curves for Reuters "earn" topic. Left is full curve; right is close-up of initial segment.

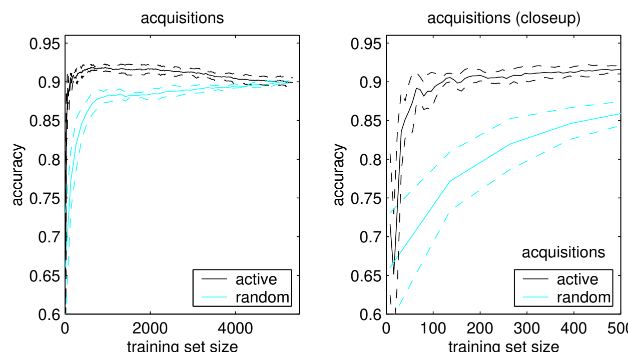

Figure 5. Accuracy curves for Reuters "acquisitions" topic. Left is full curve; right is close-up of initial segment.

be quickly exhausted. This is consistent with the relative parity between random and active learners on the USENET data, where positive and negative classes were evenly matched.

5. Discussion

In the previous section, we observed an unusual phenomenon with the learning curves. When training examples were added at random, generalization increased monotonically until all available examples were added. When training examples were added via the heuristic, generalization peaked to a level above that achieved by using all available data, then slowly degraded to the level achieved by the random learner when all data had finally been added. We are actually achieving *better* performance from a small subset of the data that we can achieve using all available data. In the name of cost and accuracy, we would like to estimate when we have achieved this peak performance, so we can stop adding examples.

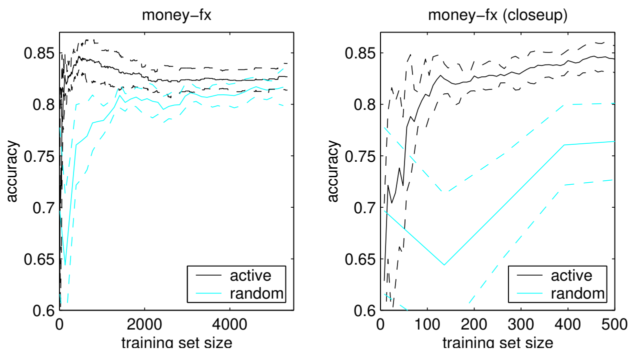

Figure 6. Accuracy curves for Reuters "moneyfx" topic. Left is full curve; right is close-up of initial segment.

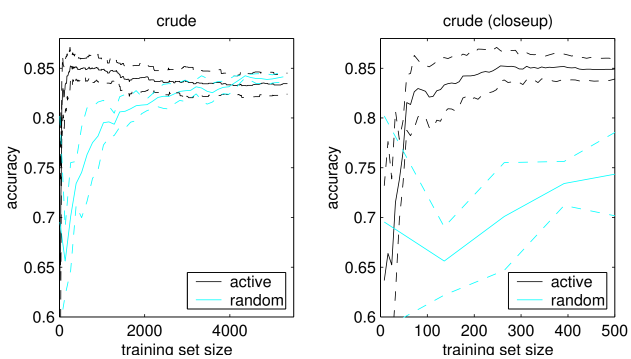

Figure 7. Accuracy curves for Reuters "crude" topic. Left is full curve; right is close-up of initial segment.

5.1 Stopping Criteria

One obvious way of estimating when we have reached peak generalization performance is the use of crossvalidation, or of a hold-out set. Crossvalidation on the training set is impractical, given the time needed to re-solve the SVM for each crossvalidation split. Even if time were not an issue, crossvalidation assumes that the training set distribution is representative of the test set distribution — an assumption violated by the active learning approach. Given that the motivation of active learning is to use as few labeled examples, the alternative of requiring a held-out validation set is counterproductive. Instead, we look for a self-contained statistic.

Let us make the (unreasonable) assumption that our data is linearly separable. If this is the case, then only unlabeled examples within the margin will have any effect on our learner. Labelling an example in the margin may shift the margin such that examples that were previously "outside" are now "inside," but once all unlabeled examples in the margin have been exhausted, no future labelings will affect the learner in the slightest.

Working from this assumption, it would be reasonable to stop labeling data once the margin has been exhausted. We can compute this stopping criterion as a byproduct of evaluating candidates for labeling – if the best one (closest to the hyperplane) is no closer than any of the support vectors, our margin has been exhausted. Empirically, this stopping criterion performs well; when the margin has been exhausted, the number of new support vectors drops (recall that it should go to zero only if our data is linearly separable). While this point generally lags the true peak, accuracy at margin exhaustion appears to closely approximate peak accuracy (see Figure 9 and Table 1).

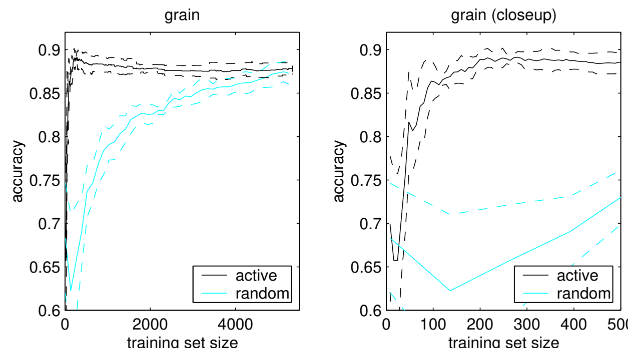

Figure 8. Accuracy curves for Reuters "grain" topic. Left is full curve; right is close-up of initial segment.

Table 1. Accuracy of active and random learners on Reuters topics, averaged over five randomized runs. 'Peak' is the number of training examples at which the active learner reached its mean peak accuracy. The maximum accuracy for random learners was achieved when all training data were used. 'Cutoff' indicates the number of examples at which the stopping criterion was satisfied.

label	peak at	max accuracy		cutoff	accuracy at cutoff	
		active	random		active	random
earn	504	0.95	0.93	875	0.95	0.922
acq	752	0.92	0.90	1050	0.917	0.882
money	456	0.846	0.817	675	0.842	0.771
crude	256	0.852	0.842	550	0.849	0.750
grain	272	0.89	0.87	580	0.883	0.741

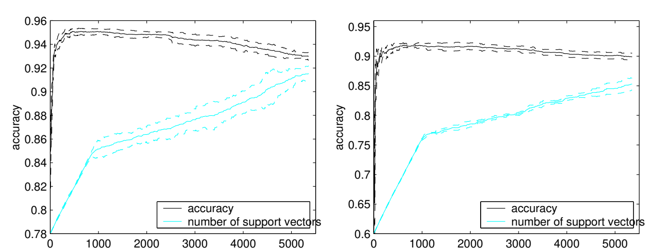

Figure 9. As examples are labeled and added to the training set, there is a "knee" in the fraction of new points that become support vectors. This point corresponds to the point where all unlabeled examples within the margin have been labeled. Accuracy is plotted along with the number of support vectors (not to scale) as a function of training set size for "earn" (left) and "acq" (right).

5.2 Timing Considerations

As we remarked earlier, training an SVM requires time $\Omega(m^2)$ in the number of training examples. Training with all available data requires solving a single QP problem with a large value of m. Active learning requires iteratively solving QP problems for small but increasing values of m. The superlinear time penalty for adding training data combines well with the increased generalization performance from active learning in our experiments. When the cutoff point is $O(n^{\frac{2}{3}})$ in the total number of examples, active learning will be more efficient than batch learning on the entire set. Though not always superior, the running times for the active learning experiments on large datasets were always competitive with, and in some cases clearly faster than training just once on all available data (Figure 10).

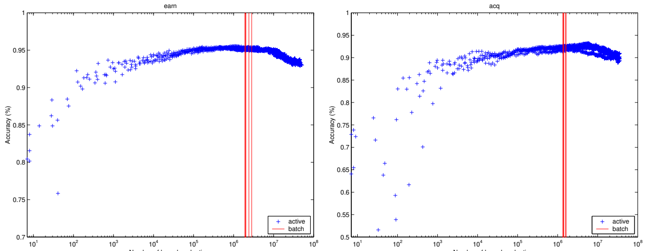

Figure 10. Accuracy as a function of time (in terms of kernel evaluations) for earn and acq. Points represent results of five independent runs on log scale; vertical lines represent the number of kernel evaluations required by the five batch runs. Peak performance for active learning requires, in general, no more time than batch learning on all available data; achieving performance comparable to that of batch learning requires an order of magnitude less.

5.3 Comparison with Bound Support Vector Removal

The peak in performance before the training data has been exhausted is not an uncommon phenomenon for other learning techniques. However, given the nature of a linear support vector machine, generalization should never be lost as data increases.[6] However, as inconsistent examples (those whose label differs from the model's prediction) are added, the hyperplane shifts, possibly to a position that increases both the training and test error. Though (soft-margin) SVMs are resilient to overfitting, they are susceptable to error introduced by noise.

Following the active learning heuristic, documents nearest the hyperplane are added first. When an example that lies inside the margin of a separable model is added to the training set, the resultant model must remain separable and will contain no bound support vectors. Therefore, when adding one example per training iteration, the active heuristic cannot add an example which will be misclassified — not until all examples within the margin have been exhausted. Examples which make the model inconsistent will not be added until no more examples remain within the margin.

One hypothesis then, would be that it is these inconsistent examples which lead to the degradation in performance, and that the degradation could be eliminated by removing inconsistent examples. Empirically, however, removing the misclassified examples and retraining does not recover the peak performance of the active learner. For instance, the "earn" group from the Reuters experiments reaches 95% accuracy through the active learner, but attains 87% accuracy on the entire set with bound support vector removal. Although bound support vector removal *sometimes* helps, it has shown no systematic improvement over traditional soft-margin classifiers, while active learning has consistently outperformed soft-margin classifiers built over all examples. Since the active learning heuristic we describe will not reach misclassified examples until after the cutoff, it may be benefitting from a better heuristic to remove noise.

5.4 Relation to Previous Work

Most SVM solvers reduce the size of original problem by disregarding dormant examples in the training set. Chunking (Boser et al., 1992) and shrinking (Joachims, 1998a) use heuristics to reduce the size of the training set. Chunking solves sub-problems by iteratively building a set of examples, using those that violate the optimality coniditions the most. Shrinking, which may be viewed as chunking in reverse, temporarily removes examples from the training set that are not likely to become support vectors, trains the model, then adds the examples and does a final optimization. Since an active learner begins with a small training set and iteratively increases its size, its computational performance parallels that of chunking.[7]. An active learner attempts to minimize number of labels for non-support vectors (since they have no affect on improving the model), active learning may also be thought of as a priori shrinking: the only examples included in the labeled training set are support vectors. It is important, however, not to forget a key distinction: in chunking and shrinking, we disregard already-labeled examples that are unlikely to be support vectors, purely for computational gain. In active learning, we avoid asking for those (expensive) labels in the first place — saving not only computation, but the time and expense of labelling unneeded data.

[6] Vapnik (1998) discusses similar effects when the problem is ill-posed. In such cases minimizing an "obvious" resolution may not yield results as good as the optimization of a "corrupted" function. Terminating the active learner early is analogous to solving a corrupted function.

[7] Though the sub-problems are solved with an SVM solver instead of a QP solver, which in turn may or may not use pruning techniques on the training set

References

Angluin, D. (1988). Queries and concept learning. *Machine Learning*, *2*, 319–342.

Baum, E., & Lang, K. (1991). Neural network algorithms that learn in polynomial time from examples and queries. *IEEE Transactions on Neural Networks*, *2*.

Boser, B. E., Guyon, I. M., & Vapnik, V. (1992). A training algorithm for optimal margin classifiers. *Proceedings of the Fifth Annual ACM Workshop on Computational Learning Theory* (pp. 144–152). Pittsburgh, PA: ACM Press.

Cohn, D., Ghahramani, Z., & Jordan, M. (1996). Active learning with statistical models. *Journal of Artificial Intelligence Research*, *4*, 129–145.

Dumais, S., Platt, J., Heckerman, D., & Sahami, M. (1998). Inductive learning algorithms and representations for text categorization. *Proceedings of the ACM CIKM International Conference on Information and Knowledge Management*.

Joachims, T. (1998a). Making large-scale SVM learning practical. In *Advances in kernel methods: Support vector learning*. Cambridge: MIT Press.

Joachims, T. (1998b). Text categorization with support vector machines: Learning with many relevant features. *Proceedings of the Tenth European Conference on Machine Learning*.

Keerthi, S., Shevade, S., Bhattacharyya, C., & Murthy, K. (1999). *Improvements to Platt's SMO algorithm for SVM classifier design* (Technical Report CD-99-14). Department of Mechanical and Production Engineering, National University of Singapore, Singapore.

Lewis, D. (1997). The Reuters-21578 text categorization test collection. *Available at http://www.research.att.com/ lewis/reuters21578.html*.

Lewis, D., & Gale, W. (1994). A sequential algorithm for training text classifiers. *Proceedings of the Twenty-first Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval*.

Nigam, K., McCallum, A., Thrun, S., & Mitchell, T. (1998). Learning to classify text from labeled and unlabeled documents. *Proceedings of the Fifteenth National Conference on Artificial Intelligence*.

Osuna, E., Freund, R., & Girosi, F. (1997). An improved training algorithm for support vector machines. *Neural Networks for Signal Processing VII – Proceedings of the 1997 IEEE Workshop* (pp. 276–285).

Platt, J. (1998). Fast training of support vector machines using sequential minimal optimization. *Advances in kernel methods: Support vector learning*. Cambridge: MIT Press.

Platt, J. (1999). Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. *Advances in large margin classifiers*. Cambridge: MIT Press.

Schölkopf, B., Burges, C., & Smola, A. (1999). *Advances in kernel methods : Support vector learning*. Cambridge: MIT Press.

Smola, A. (1998). Quadratic optimizer for pattern recognition. *Unpublished manuscript, German National Research Center for Information Technology. Available at http://svm.first.gmd.de/software/loqosurvey.html*.

Tong, S., & Koller, D. (2000). Support vector machine active learning with applications to text classification. *Seventeenth International Conference on Machine Learning*.

Vapnik, V. (1998). *Statistical learning theory*. New York: John Wiley.

Vavasis, S. A. (1991). *Nonlinear optimization: Complexity issues*. New York: Oxford Science.

Yang, Y. (1999). An evaluation of statistical approaches to text categorization. *Information Retrieval Journal, May 1999* (pp. 67–68).