Incorporating Diversity in Active Learning with Support Vector Machines

Klaus Brinker KBRINKER@UNI-PADERBORN.DE
International Graduate School of Dynamic Intelligent Systems, University of Paderborn, 33098 Paderborn, Germany

Abstract

In many real world applications, active selection of training examples can significantly reduce the number of labelled training examples to learn a classification function. Different strategies in the field of support vector machines have been proposed that iteratively select a single new example from a set of unlabelled examples, query the corresponding class label and then perform retraining of the current classifier. However, to reduce computational time for training, it might be necessary to select batches of new training examples instead of single examples. Strategies for single examples can be extended straightforwardly to select batches by choosing the $h > 1$ examples that get the highest values for the individual selection criterion. We present a new approach that is especially designed to construct batches and incorporates a diversity measure. It has low computational requirements making it feasible for large scale problems with several thousands of examples. Experimental results indicate that this approach provides a faster method to attain a level of generalization accuracy in terms of the number of labelled examples.

1. Introduction

The standard setting in classification learning assumes that a previously labelled set of examples is available. While this assumption holds for a large number of real world applications, there are some applications in which we have only access to an initially unlabelled set of examples. Since labelling these examples can be expensive in terms of both time and money, we naturally try to minimize the number of labelled examples that are necessary to learn a classification function at a certain accuracy level. Actively selecting new training examples from the set of unlabelled examples, then querying their class labels and incrementally learning a classification function is an efficient strategy to control the labelling effort and accelerate the learning process.

Support vector machines (Vapnik, 1998) have received ample treatment being both theoretically well founded and showing excellent generalization performance in practice. Several publications discuss active learning strategies for support vector machines that select training examples from a finite set of unlabelled examples. It has been shown empirically that active selection outperforms learning by randomly adding training examples in the field of character recognition (Campbell et al., 2000), document classification (Schohn & Cohn, 2000; Tong & Koller, 2000) and computational chemistry (Warmuth et al., 2002). Since it is time consuming to retrain the classifier whenever a new example is added to the training set, it is more efficient from a computational point of view to select and label a set of examples before repeatedly running the training algorithm. Furthermore, if a parallel labelling instance is available, e.g. a number of labels can be determined at the same time by an experimental test procedure, we want to take advantage of it. Previously studied strategies for single examples have been extended to select batches in a straightforward manner by choosing the $h > 1$ examples that get the highest values for the individual selection criterion.

可见主动学习算法都是从头重新模型

这句话没懂什么意思

We present an approach that is especially designed to construct batches of new training examples and enforces selected examples to be diverse with respect to their angles. Our approach has low computational requirements making it feasible for large scale problems with several thousands of examples. Compared to previous approaches, the experimental results presented in section 5 indicate that this approach provides a faster method to attain a level of generalization accuracy in terms of the number of labelled examples.

The remainder of this paper is structured as follows: In the subsequent section, we recapitulate fundamental properties of support vector machines that are relevant in the field of active learning. In the first part of section 3, we discuss previous approaches to active learning with support vector machines, while the second part introduces our new selection strategy. Section 5 shows experimental results supporting the efficiency of our strategy. Finally, we summarize our results and discuss open research topics.

2. Support Vector Machines

We consider a standard binary classification problem consisting of n training examples $\{(x_1, y_1), \ldots, (x_n, y_n)\} \subset (\mathcal{X} \times \{-1, +1\})^n$ with $\mathcal{X}$ denoting a nonempty input space. Support vector machines constitute kernel classifiers which can be expanded in terms of the training examples (Schölkopf & Smola, 2002):

$$f(x) = \operatorname{sign}\big(\underbrace{\textstyle\sum_{i=1}^{n} \alpha_i \, k(x_i, x)}_{=: g(x)}\big) \tag{1}$$

with $\alpha = (\alpha_1, \ldots, \alpha_n) \in \mathbb{R}^n$ and k being a kernel function.[1] If the kernel k satisfies Mercer's condition, there exists a (nonunique) feature space $\mathcal{F}$ and a map ϕ from the input space $\mathcal{X}$ to the feature space $\mathcal{F}$ such that k corresponds to a dot product in $\mathcal{F}$ by $k(x, x') = \langle \phi(x), \phi(x') \rangle$. Therefore, the classification function can be rewritten as

$$\begin{aligned} f(x) &= \operatorname{sign}\Big(\big\langle \underbrace{\textstyle\sum_{i=1}^{n} \alpha_i \phi(x_i)}_{=: w_{\text{svm}}}, \phi(x) \big\rangle\Big) \\ &= \operatorname{sign}\big(\langle w_{\text{svm}}, \phi(x) \rangle\big). \end{aligned}$$

Thus, the classification boundary is a hyperplane in feature space $\mathcal{F}$ with normal vector w_{svm}.

If we assume that the training set is linearly separable in feature space, (hard margin) support vector machines calculate the coefficient vector α such that the classifier is consistent with the training set and that the margin between training examples and the classification boundary in feature space is maximized. Since scaling with a positive constant does not have an effect on the classification outcome, we can normalize w_{svm} by requiring that the closest example has unit functional distance to the classification hyperplane: $\min_{i=1,\ldots,n} |\langle w_{\text{svm}}, \phi(x_i) \rangle| = 1$ (which is called the canonical form of the hyperplane with respect to $x_1, \ldots, x_n$). Calculating the coefficient vector α amounts to solving a quadratic programming problem. Highly efficient algorithms have been developed to accomplish this task in the special case of support vector machines (Platt, 1999).

If the training set is not linearly separable in feature space (e.g. noisy data), we can modify any kernel by adding some constant $\nu > 0$ to the diagonal elements of the kernel matrix, $k(x_i, x_j) + \delta_{ij}\,\nu$, such that the training set becomes linearly separable (Shawe-Taylor & Cristianini, 1999).

3. Active Learning

3.1. Selection Strategy for Single Examples

Let us consider a linearly separable problem in feature space, i.e. we assume that there exists a linear classifier $\tilde{f}(x) = \operatorname{sign}(\langle \tilde{w}, \phi(x) \rangle)$ satisfying $\tilde{f}(x_i) = y_i$ for each training example x_i. The nonempty set

$$\begin{aligned} \mathcal{V} := \{ w \in \mathcal{F} \mid y_i \langle w, \phi(x_i) \rangle > 0 \text{ for } i = 1, \ldots, n \\ \text{and } \|w\| = 1 \} \end{aligned}$$

is called *version space* (Mitchell, 1982). $\mathcal{V}$ consists of all (normalized[2]) weight vectors corresponding to linear classifiers in feature space which separate the training set without errors. We can view learning as a search problem with the version space $\mathcal{V}$ containing the solution we are looking for. Each training example limits the volume of the version space because consistent solutions can only lie on one side of the hyperplane with normal vector $\phi(x_i)$, depending on the class y_i. Therefore, $\mathcal{V}$ is the intersection of n halfspaces (a convex polyhedral cone) with the unit sphere in feature space $\mathcal{F}$.

If all support vectors have the same length in feature space, $w_{\text{svm}}/||w_{\text{svm}}||$ is a reasonable approximation of the center of mass of $\mathcal{V}$. Furthermore, the center of mass approximates the Bayes point which is the center of the region of intersection of all hyperplanes bisecting the version space into two halves of equal volume (Ruján & Marchand, 2000; Herbrich et al., 2001). Therefore, if we select a new training example with minimal distance to the classification hyperplane, the

[1]Although we have omitted a bias term in (1) and, therefore, consider only hyperplanes passing through the origin in feature space $\mathcal{F}$, we can nevertheless express classification functions that do not pass through the origin in input space $\mathcal{X}$ by using appropriate kernel functions such as inhomogeneous polynomial kernels.

[2]In this section we use the data-independent normalization to unit length for theoretical analysis, while in the rest of this paper we assume normal vectors to be in canonical form with respect to the training set.

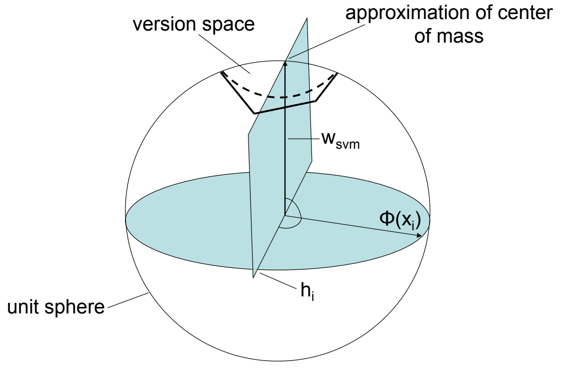

Figure 1. *$\phi(x_i)$ is perpendicular to w_{svm}, in other words, its distance to the classification hyperplane in feature space is zero. The corresponding hyperplane h_i approximately bisects the version space into two halves of equal volume.*

corresponding hyperplane with normal vector $y_i\,\phi(x_i)$ approximately bisects the version space into two halves of equal volume (see figure 1). Thus reducing the initial search problem to a space of half the volume. We can approximate the new center of mass by training a support vector machine based on the augmented training set and repeat these steps as often as necessary. Apart from the version space model which has been considered in (Tong & Koller, 2000) there are other theoretical justifications for this approach (Campbell et al., 2000; Schohn & Cohn, 2000). We refer to this selection strategy as the distance strategy.

3.2. Selection Strategy for Batches

Consecutively selecting that unlabelled example which is closest to the classification boundary can theoretically be well motivated for batch size $h = 1$ in the version space model as stated above. This strategy can be extended for batch sizes $h > 1$ in a straightforward manner by selecting those h unlabelled examples whose functional distances to the classification hyperplane in feature space $\mathcal{F}$ are minimal (Warmuth et al., 2002; Schohn & Cohn, 2000). However, the theoretical motivation for each selected example to approximately bisect the version space into two halves of equal volume does not hold in this case. Adding a batch of new examples to the training set that is selected exclusively based on the distances to the classification hyperplane does not necessarily yield a much greater reduction of the volume of the version space than simply adding one such example in general. As illustrated in figure 2, where the enclosed angles between selected examples are small, there might only be a small additional reduction of volume induced by the second and third

为什么不直接选函数距离最小的

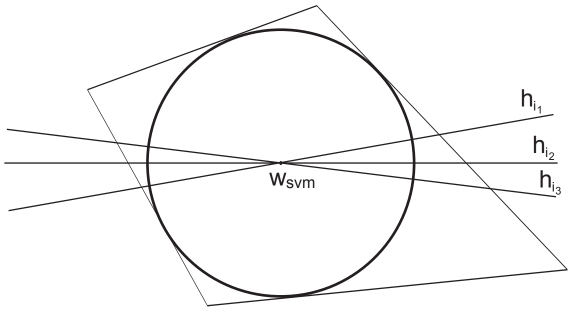

Figure 2. The angles between the hyperplanes that are induced by the three examples are small. Therefore, the additional reduction of version space resulting from examples two and three can be rather small, despite the fact that their distances to the classification hyperplane are zero.

example.

A straightforward approach to deal with this potential problem is to select batches yielding minimum worst-case version space volume (as an extension of (Tong & Koller, 2000)). However, this method requires an exhaustive search in the space of all possible label assignments for all batches of size h and expensive volume estimation techniques making it unfeasible in practice. Our new heuristic selection strategy tries to overcome this problem by incorporating a diversity measure that considers the angles between the induced hyperplanes. Calculation of the (undirected) angle between two hyperplanes h_i and h_j which correspond to examples x_i and x_j (with normal vectors $\phi(x_i)$ and $\phi(x_j)$) can be written in terms of the kernel function:

这个角度应该是基于法线和超平面的交点

$$|\cos(\angle(h_i, h_j))| = \frac{|\langle\phi(x_i), \phi(x_j)\rangle|}{\|\phi(x_i)\|\|\phi(x_j)\|} = \frac{|k(x_i, x_j)|}{\sqrt{k(x_i, x_i)k(x_j, x_j)}}.$$

To maximize the angles within a set of hyperplanes, we employ the following incremental strategy: Let I denote the set of indices of unlabelled examples that have not been selected for training yet. Starting with an initial hyperplane h_{i_1}, we add that unlabelled example x_{i_2} to our set $S = \{x_{i_1}\} \cup \{x_{i_2}\}$ whose corresponding hyperplane h_{i_2} maximizes the angle to h_{i_1}. We continue by adding further examples x_{i_j} that minimize

$$\max_{l\in\{1,\ldots,j-1\}} \underbrace{\frac{|k(x_{i_l}, x_{i_j})|}{\sqrt{k(x_{i_l}, x_{i_l})k(x_{i_j}, k_{i_j})}}}_{=:k^*(x_{i_l}, x_{i_j})}. \qquad (2)$$

Figure 3 illustrates this strategy for the three-dimensional case by projecting the version space,

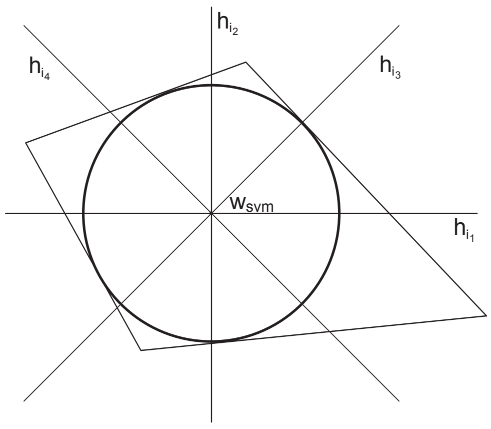

Figure 3. We assume that there exist unlabelled examples $h_{i_1}, \ldots, h_{i_4}$ that take the global minimum of (2). Therefore, each newly chosen example corresponds to a hyperplane which maximizes the minimum angle to previous hyperplanes.

which is a subset of the unit sphere, to the plane. From a more abstract point of view, this strategy ensures that the chosen unlabelled examples are diverse in terms of their angles to each other in feature space.

Finally, in order to combine both requirements, viz. minimal distance to the classification hyperplane and diversity of angles, we build the convex combination of both measures and proceed in the following way to construct a new training batch: Let I^* denote the set of indices of unlabelled examples that have not yet been selected for training and which have a distance to the classification hyperplane that is less than one. The additional distance restriction ensures that hyperplanes corresponding to (normalized) examples in I^* intersect with the version space. We incrementally construct a new training batch S:

```
1: $S = \emptyset$
2: **repeat**
3:     $t = \underset{i \in I^* \setminus S}{\operatorname{argmin}} \big(\lambda\, |g(x_i)| + (1-\lambda) \max_{j \in S} k^*(x_i, x_j)\big)$
4:     $S = S \cup \{x_t\}$
5: **until** card$(S) = h$
```

(With g denoting the real-valued output of the classifier f before thresholding as defined in section 2.)

The individual influence of each requirement can be adjusted by the trade-off parameter λ. Setting $\lambda = 1$ restores the distance strategy, whereas for $\lambda = 0$ the algorithm focuses exclusively on maximizing the angle diversity.

This combined strategy can be implemented very efficiently and is almost as fast as the distance strategy (see next section for details). Reevaluating the second part of the sum in line 3 in a naive way for every single example that is added to the training batch results in a quadratic dependence of computational time on the size of the new batch h. It is more efficient to cache the values of the second part for all card$(I \backslash S)$ unlabelled examples and perform an update if the cosine of the angle between an unlabelled example and a newly added example is greater than the stored maximum. For each newly added example, this requires three kernel evaluations for all card$(I \backslash S)$ unlabelled examples.

To distinguish between previously labelled and still unlabelled examples, we store a permutation $I = (i_1, \ldots, i_n)$ of $\{1, \ldots, n\}$. The first part $(i_1, \ldots, i_{s-1})$ denotes the indices of previously labelled examples, while $(i_s, \ldots, i_n)$ denotes the indices of unlabelled examples. The complete pseudo code of an efficient implementation of the combined strategy is given below.

4. Computational Complexity

Experimental results indicate that support vector machines typically require $\mathcal{O}(m^2)$ time for training, where m denotes the number of examples (Platt, 1999). Therefore, actively learning m examples by adding h (with $1 \leq h < m$ and $h|m$ for notational convenience) examples to the training set before performing retraining requires an accumulated computational time of order $\mathcal{O}(\frac{m^3}{h})$, excluding the selection steps.

To select new examples, we need to calculate the distances of all unlabelled examples to the classification hyperplane once within each iteration, i.e. for every h examples. Denoting the initial number of unlabelled examples by n, we obtain a total time for distance calculations of order $\mathcal{O}(n\,\frac{m^2}{h})$ taking into account that the computational time for one distance calculation is proportional to the number of labelled examples. In addition, for each newly added example, three kernel evaluations for all unlabelled examples are necessary to update the angle values. This amounts to $\mathcal{O}(n\,m)$ time in total.

Summing up training time and selection time, actively learning m examples from a pool of n examples and batch size h with the combined strategy requires computational time of order

$$\mathcal{O}\Big(\underbrace{\frac{m^3}{h}}_{\text{training}} + \underbrace{n\,\frac{m^2}{h}}_{\text{distance calculations}} + \underbrace{n\,m}_{\text{angle calculations}} \Big).$$

In comparison to the distance strategy, the combined strategy requires additional computational time of order $\mathcal{O}(n\,m)$.

Algorithm 1 Select training examples

```
input:
$\lambda$          (trade-off between distance and diversity)
$h$                                          (batch size)
$s$                                      (start position)
$I = (i_1, \ldots, i_n)$         (permutation of $\{1,\ldots,n\}$)
$g : \mathcal{X} \rightarrow \mathbb{R}$    (unthresholded classification function)

output:
$I$                              (permutation of $\{1,\ldots,n\}$)

distance = array[$n - s + 1$] of double
maxCos = array[$n - s + 1$] of double

for $j = 0$ to $n - s$ do
  distance($j$) = $g(x_{i_{s+j}})$
  maxCos($j$) = 0
end for

for $k = 0$ to $h - 1$ do
  minIndex = $k$
  minValue = $+\infty$
  for all $j = k$ to $n - s$ do
    if distance($j$) < 1 then
      value = $\lambda$ distance($j$) + $(1 - \lambda)$ maxCos($j$)
      if value < minValue then
        minIndex = $j$
        minValue = value
      end if
    end if
  end for

  swap($i_{s+\text{minIndex}}, i_{s+k}$)

  for all $j = k + 1$ to $n - s$ do
    cos = $k^*(x_{i_{s+k}}, x_{i_{s+j}})$
    if cos > maxCos($j$) then
      maxCos($j$) = cos
    end if
  end for
end for
```

5. Experiments

5.1. Experimental Setting

To evaluate the combined selection strategy we have conducted several experimental studies on data sets that are publicly available from the UCI repository of machine learning databases (Blake & Merz, 1998) and from the STATLOG collection (Michie et al., 1994). Our experiments include the distance and the combined strategy. Additionally, we compare both strategies to random selection of new training batches which serves as a base line.

Each of the data sets was randomly split 100 times into a training set and a test set of equal size. Active selection was restricted to the training set. The initial training batches always consist of 8 examples which are randomly drawn from the entire training set and contain at least one example from class -1 and class $+1$. The generalization accuracy is evaluated on the test set after each iteration (selection and training) and the results are averaged over all 100 runs. In our experiments, we have used a modified version of bsvm (Hsu & Lin, 2002) that trains support vector machines without bias to stay consistent with our theoretical motivation. All experiments were performed on a single processor pentium 4 with 1.8 ghz and 1 gb of memory.

h	1	2	4	8	16	32	64
time	565s	367s	210s	135s	73s	54s	34s

Figure 4. Running times (selection+training) for the combined strategy on the shuttle *data set for different choices of* h*. The final number of training examples was* 456 *from a pool of size* 21750*.*

We chose the *shuttle* data set from STATLOG as our first experimental problem. It contains 43500 examples[3] with 9 continuous attributes. Approximately 80% of the examples belong to class 1 of the 7 classes. We tested all strategies on the binary classification problem class 1 against the rest and used an RBF kernel with $\sigma = 2$.

Our second data set is the *waveform-5000* problem from UCI which contains 5000 examples with 21 continuous attributes and 3 classes. The class distribution is $\frac{1}{3}$ for each of the 3 classes. We carried out our test runs on the class 0 against the rest problem using an inhomogeneous polynomial kernel of degree 5.

[3]The original data set contains 58000 examples that are usually split into 43500 training and 14500 test examples. For computational reasons, we only used the first 43500 examples.

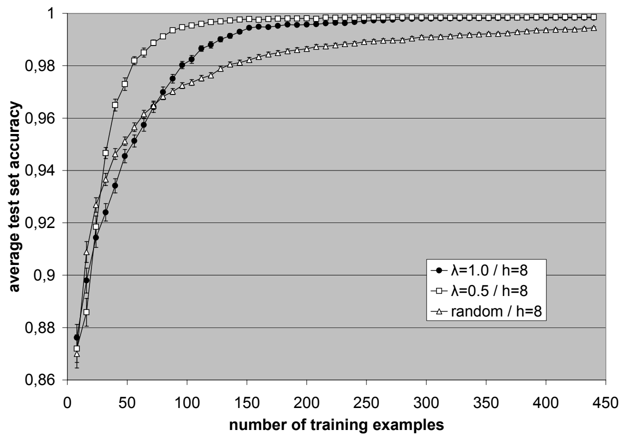

Figure 5. Learning curves for the distance*,* combined *and* random *selection strategy on the* shuttle *data set. The generalization accuracy was averaged over 100 random splits into training and test sets. Error bars indicate one standard error of the mean. Within each iteration $h = 8$ new examples are added to the training set. The* combined *strategy outperforms both the* distance *and the* random *selection strategy.*

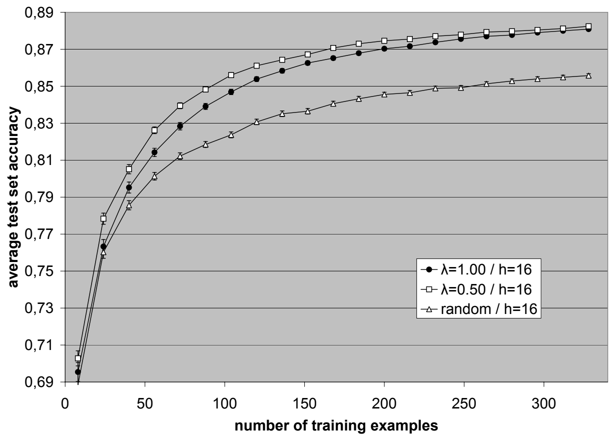

Figure 6. Learning curves for the distance*,* combined *and* random *selection strategy on the* waveform-5000 *data set. The generalization accuracy was averaged over 100 random splits into training and test sets. Error bars indicate one standard error of the mean. Within each iteration $h = 16$ new examples are added to the training set. The* combined *strategy outperforms both the* distance *and the* random *selection strategy.*

The *krkpa7* data set from UCI contains 3196 examples with 36 discrete attributes taking either 2 or 3 different values. There are 2 different classes to predict with approximately 52% belonging to class 1 and 48% belonging to class 2. On the *krkpa7* problem we used an inhomogeneous polynomial kernel of degree 3.

The first part of our experiments has been set up to explore the influence of the batch size on the efficiency of the selection strategies. For all data sets described above we fixed $\lambda = 0.5, 1.0$ and tested batch sizes $h = 8, 16, 32, 64$. For $\lambda = 1.00$ the combined strategy corresponds to the distance strategy. In the second part, we focused on the influence of the parameter λ. Therefore we fixed the batch size $h = 16$ and compared the combined strategy for $\lambda = 0, 0.25, 0.50, 0.75, 1.00$ to the random strategy.

5.2. Experimental Results

With fixed $\lambda = 0.5$, the estimated generalization accuracy of the combined strategy is consistently superior to the distance strategy ($\lambda = 1.00$), independent of the batch size (see figure 5 for a typical learning curve for batch size $h = 8$ on the *shuttle* data set and figure 6 for batch size $h = 16$ on the *waveform-5000* data set). Furthermore, the efficiency of both the combined and the distance decreases if the batch size h increases (figure 7 and figure 8 show learning curves for the *krkpa7* data set and *waveform-5000* data set for different batch sizes). With the number of training examples increasing we observed higher generalization accuracy for the combined and the distance strategy compared to the random strategy in all our experiments. However, at the beginning the random strategy tends to perform better with the turning point increasing with the batch size h. For batch size $h = 8, 16$, the turning point is typically reached after less than 50 training examples, whereas for $h = 64$ it can take several hundred examples. Therefore, our experiments suggest that with respect to generalization accuracy it is preferable to choose h as small as possible, while from a computational point of view increasing h allows to control computational time. Hence, h has to be chosen carefully as a trade-off between accuracy and computational complexity.

batch-size影响拐点

batch-size越小越好

从计算角度越大越好

In our experiments the learning curves for $\lambda = 0.25, 0.50, 0.75$ were very similar (see figure 9 for the *shuttle* data set and figure 10 for the *waveform-5000* data set) and consistently superior to the distance strategy ($\lambda = 1.00$). Contrary to this, the behavior of the combined strategy for $\lambda = 0.00$ is rather unstable ranging from the best (for the *shuttle* data set) to the worst strategy (for the *waveform-5000* data set) apart from the random strategy. Thus, except for the extreme choice of $\lambda = 0.00$, the combined strategy seems to be fairly robust with respect to λ.

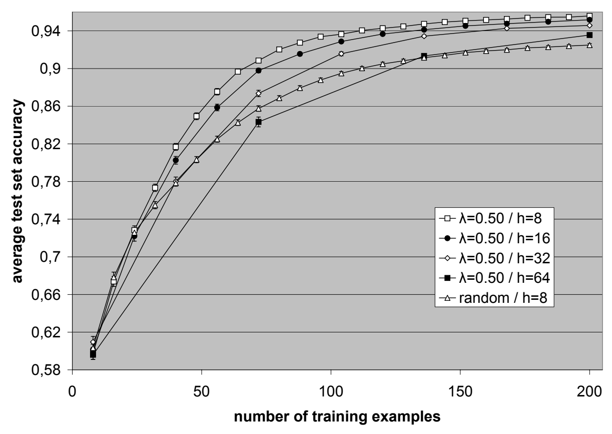

Figure 7. Learning curves for the combined *selection strategy ($\lambda = 0.50$) and for the* random *strategy on the* krkpa7 *data set. The generalization accuracy was averaged over 100 random splits into training and test sets. Error bars indicate one standard error of the mean. Within each iteration $h = 8, 16, 32, 64$ new examples are added to the training set. For smaller batch sizes h the* combined *strategy is more efficient.*

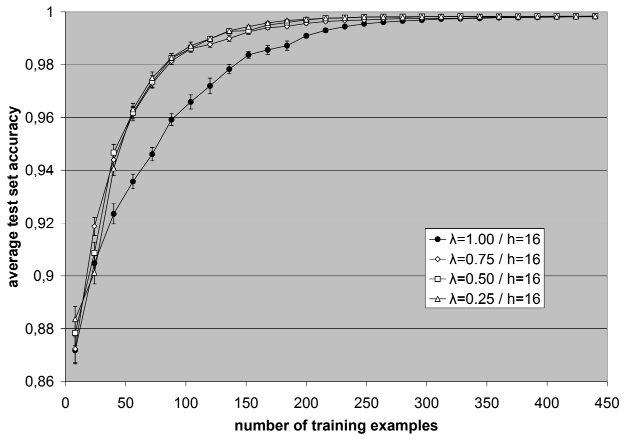

Figure 9. Learning curves for the distance *and* combined *selection strategy for $\lambda = 0.25, 0.50, 0.75$ on the shuttle data set. The generalization accuracy was averaged over 100 random splits into training and test sets. Error bars indicate one standard error of the mean. Within each iteration $h = 16$ new examples are added to the training set. For all choices of λ the* combined *strategy outperforms the* distance *selection strategy.*

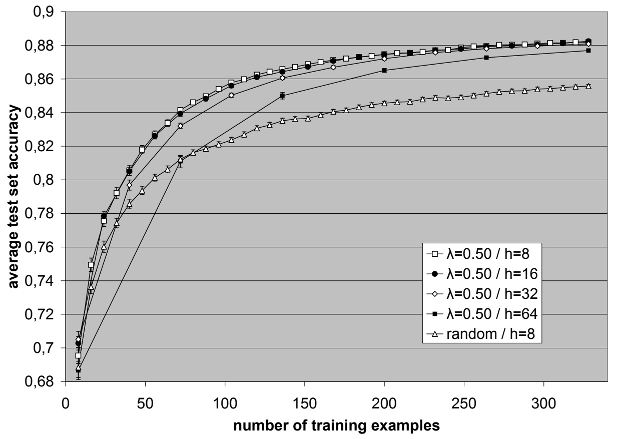

Figure 8. Learning curves for the combined *selection strategy ($\lambda = 0.50$) and for the* random *strategy on the* waveform-5000 *data set. The generalization accuracy was averaged over 100 random splits into training and test sets. Error bars indicate one standard error of the mean. Within each iteration $h = 8, 16, 32, 64$ new examples are added to the training set. For smaller batch sizes h the* combined *strategy is more efficient.*

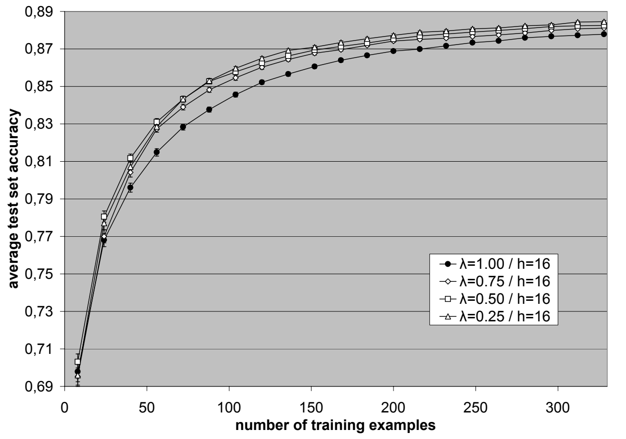

Figure 10. Learning curves for the distance *and* combined *selection strategy for $\lambda = 0.25, 0.50, 0.75$ on the waveform data set. The generalization accuracy was averaged over 100 random splits into training and test sets. Error bars indicate one standard error of the mean. Within each iteration $h = 16$ new examples are added to the training set. For all choices of λ the* combined *strategy outperforms the* distance *selection strategy.*

6. Conclusions and Future Research

Our experiments indicate that the combined selection strategy is an efficient method to construct batches of new training examples outperforming previous approaches in active learning with support vector machines. Involving only a small amount of additional computational time, this approach provides a faster method to attain a level of generalization accuracy in terms of the number of labelled examples. This makes our approach feasible for data sets with thousands of examples. Although the combined selection strategy is fairly robust with respect to the trade-off parameter λ, the question of how to choose an optimal value for λ which may depend on the progress of the training process is yet unanswered. Beyond active learning with support vector machines, the proposed batch selection technique can be adapted to other base strategies which aim at selecting examples halving the volume of version space such as query-by-committee (Seung et al., 1992). Particularly, it is possible to apply our combined selection strategy without modifications to Bayes point machines, which are able to more accurately approximate the center of mass if the version space is not symmetrical (Herbrich et al., 2001). Preliminary experiments show very promising results.

Acknowledgements

We would like to thank Eyke Hüllermeier, Hans Kleine Büning and the anonymous reviewers for fruitful suggestions. Additionally, we have to thank Chih-Len Lin for helpful comments on the implementation.
This work was supported by the International Graduate School of Dynamic Intelligent Systems, University of Paderborn.

References

Blake, C. L., & Merz, C. J. (1998). UCI repository of machine learning databases. Data available at `http://www.ics.uci.edu/~mlearn/MLRepository.html`.

Campbell, C., Cristianini, N., & Smola, A. (2000). Query learning with large margin classifiers. *Proceedings of the Seventeenth International Conference on Machine Learning*, 111–118.

Herbrich, R., Graepel, T., & Campbell, C. (2001). Bayes point machines. *Journal of Machine Learning Research*, *1*, 245–279.

Hsu, C.-W., & Lin, C.-J. (2002). A simple decomposition method for support vector machines. *Machine Learning*, *46*, 291–314. Implementation available at `http://www.csie.ntu.edu.tw/~cjlin/bsvm/`.

Michie, D., Spiegelhalter, D. J., & Taylor, C. C. (1994). *Machine learning, neural and statistical classification.* Ellis Horwood. Data available at `ftp.ncc.up.pt/pub/statlog/`.

Mitchell, T. M. (1982). Generalization as search. *Artificial Intelligence*, *18*, 203–226.

Platt, J. (1999). Fast training of support vector machines using sequential minimal optimization. *Advances in Kernel Methods — Support Vector Learning* (pp. 185–208). Cambridge, MA: MIT Press.

Rufján, P., & Marchand, M. (2000). Computing the bayes kernel classifier. *Advances in Large Margin Classifiers* (pp. 329–348). Cambridge, MA: MIT Press.

Schohn, G., & Cohn, D. (2000). Less is more: Active learning with support vector machines. *Proceedings of the Seventeenth International Conference on Machine Learning* (pp. 839–846). Morgan Kaufmann, San Francisco, CA.

Schölkopf, B., & Smola, A. J. (2002). *Learning with kernels: Support vector machines, regularization, optimization, and beyond.* Cambridge, MA: MIT Press.

Seung, H. S., Opper, M., & Sompolinsky, H. (1992). Query by committee. *Proceedings of the Fifth Workshop on Computaional Learning Theory* (pp. 287–294). San Mateo, CA: Morgan Kaufmann.

Shawe-Taylor, J., & Cristianini, N. (1999). Further results on the margin distribution. *Proceedings of the twelfth annual conference on Computational learning theory* (pp. 278–285). Santa Cruz, CA: ACM Press.

Tong, S., & Koller, D. (2000). Support vector machine active learning with applications to text classification. *Proceedings of the Seventeenth International Conference on Machine Learning* (pp. 999–1006). Morgan Kaufmann, San Francisco, CA.

Vapnik, V. (1998). *Statistical learning theory.* N.Y.: John Wiley.

Warmuth, M. K., Rätsch, G., Mathieson, M., Liao, J., & Lemmen, C. (2002). Active learning in the drug discovery process. *Advances in Neural information processings systems.*