Active Learning Using Hint Information

Chun-Liang Li, Chun-Sung Ferng, and Hsuan-Tien Lin
{b97018, r99922054, htlin}@csie.ntu.edu.tw
Department of Computer Science, National Taiwan University

Keywords: Active Learning, Support Vector Machine

Abstract

The abundance of real-world data and limited labeling budget calls for active learning, which is an important learning paradigm for reducing human labeling efforts. Many recently developed active learning algorithms consider both uncertainty and representativeness when making querying decisions. However, exploiting representativeness with uncertainty concurrently usually requires tackling sophisticated and challenging learning tasks, such as clustering. In this paper, we propose a new active learning framework, called *hinted sampling*, which takes both uncertainty and representativeness into account in a simpler way. We design a novel active learning algorithm within the hinted sampling framework with an extended support vector machine. Experimental results validate that the novel active learning algorithm can result in a better and more stable performance than that achieved by state-of-the-art algorithms. We also show that the hinted sampling framework allows improving another active learning algorithm designed from the transductive support vector machine.

1 Introduction

Labeled data are the basic ingredients in training a good model in machine learning. It is common in real-world applications when one needs to cope with a large amount of data with costly labeling steps. For example, in the medical domain, a doctor may be required to distinguish (label) cancer patients from non-cancer patients according to their clinical records (data). In such applications, an important issue is to achieve high accuracy within a limited labeling budget. This issue demands active learning (Settles, 2009), which is a machine learning setup that allows iteratively querying the labeling oracle (doctor) in a strategic manner to label some selected instances (clinic records). By using a suitable query strategy, an active learning approach can achieve high accuracy within a few querying iterations i.e., only a few calls to the costly labeling oracle (Settles, 2009).

One intuitive approaches in active learning is called uncertainty sampling (Lewis and Gale, 1994). This approach maintains a classifier on hand, and queries the most uncertain instances, whose uncertainty is measured by the closeness to the decision boundary of the classifier, to fine-tune the boundary. However, the performance of uncertainty sampling becomes restricted owing to the limited view of the classifier. In other words, uncertainty sampling can be hair-splitting on the local instances that confuse the classifier, but not considering the global distribution of instances. Therefore, queries may not represent the underlying data distribution well, leading to unsatisfactory performance of uncertainty sampling (Settles, 2009).

As suggested by Cohn et al. (1996) as well as Xu et al. (2003), active learning can be improved by considering the unlabeled instances in order to query the instance that is not only uncertain to the classifier on hand but also "representative" to the global data distribution. There are many existing algorithms that use unlabeled information to improve the performance of active learning, such as representative sampling (Xu et al., 2003).

Representative sampling makes querying decisions by not only the uncertainty of each instance, but also the representativeness, which is measured by determining whether the instances reside in a dense area. Typical representative sampling algorithms (Xu et al., 2003; Nguyen and Smeulders, 2004; Dasgupta and Hsu, 2008) estimate the underlying data distribution via clustering methods. However, the performance of the algorithms depends on the result of clustering, which is a sophisticated and non-trivial task, especially when the instances are within a high dimensional space. Another state-of-the-art algorithm (Huang et al., 2010) models the representativeness by estimating the potential label assignment of the unlabeled instances on the basis of the min-max view of active learning (Hoi et al., 2008). The performance of this algorithm depends on the results of estimating the label assignments, which is also a complicated task. Yet another representative sampling algorithm makes potential label assignment of the unlabeled instances from the view of transductive learning, such as transductive SVM (TSVM; Wang et al., 2011).

In this work, we propose a novel framework of active learning, hinted sampling, which considers the unlabeled instances as hints (Abu-Mostafa, 1995) of the global data distribution, instead of directly clustering them or estimating their label assignments. This leads to a simpler active learning algorithm. Similar to representative sampling, hinted sampling also considers both uncertainty and representativeness. Somehow hinted sampling enjoys the advantage of simplicity by avoiding the clustering or label-assignment estimation steps. We demonstrate the effectiveness of hinted sampling by designing a novel algorithm with support vector machine (SVM; Vapnik, 1998). In the algorithm, we extend the usual SVM to a novel formulation, HintSVM, which is easier to solve than either clustering or label-assignment estimation. We then study a simple hint selection strategy to improve the efficiency and effectiveness of the proposed algorithm. Experimental results demonstrate that the simple HintSVM is comparable to the best of both uncertainty sampling and representative sampling algorithms, and results in better and more stable performance than other state-of-the-art active learning algorithms.

To demonstrate the generality of hinted sampling, we further extend the TSVM approach for active learning (Wang et al., 2011) to HintTSVM to show that the proposed

framework can benifit not only uncertainty sampling but also representative sampling. Experimental results confirm the promising performance of HintTSVM as well as the usefulness of the proposed hinted sampling framework.

The rest of the paper is organized as follows. Section 2 introduces the formal problem definition and reviews the related works. Section 3 describes our proposed hinted sampling framework as well as the HintSVM algorithms with the simple hint selection strategy, and reports experiment results and comparisons. Section 4 discusses TSVM and HintTSVM with experimental justifications. Finally, we conclude in Section 5.

A short version of the paper appeared in the 2012 Asian Conference on Machine Learning (Li et al., 2012). The paper was then enriched by discussing more related works in Section 2, refining the hint sampling strategies along with broader experiments in Section 3, and the novel extension of TSVM (Wang et al., 2011) to hinted sampling for active learning in Section 4.

2 Problem Definition and Related Works

In this work, we focus on pool-based active learning for binary classification, which is one of the most common setups in active learning (Lewis and Gale, 1994). At the initial stage of the setup, the learning algorithm is presented with a labeled data pool and an unlabeled data pool. We denote the labeled data pool by $\mathcal{D}_l = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), ..., (\mathbf{x}_N, y_N)\}$ and the unlabeled data pool by $\mathcal{D}_u = \{\widetilde{\mathbf{x}}_1, \widetilde{\mathbf{x}}_2, ..., \widetilde{\mathbf{x}}_M\}$, where the input vectors $\mathbf{x}_i, \widetilde{\mathbf{x}}_j \in \mathbf{R}^d$ and the labels $y_i \in \{-1, 1\}$. Usually, the labeled data pool $\mathcal{D}_l$ is relatively small or even empty, whereas the unlabeled data pool $\mathcal{D}_u$ is assumed to be large. Active learning is an iterative process that contains R iterations of querying and learning. That is, an active learning algorithm can be split into two parts: the querying algorithm $\mathcal{Q}$ and the learning algorithm $\mathcal{L}$.

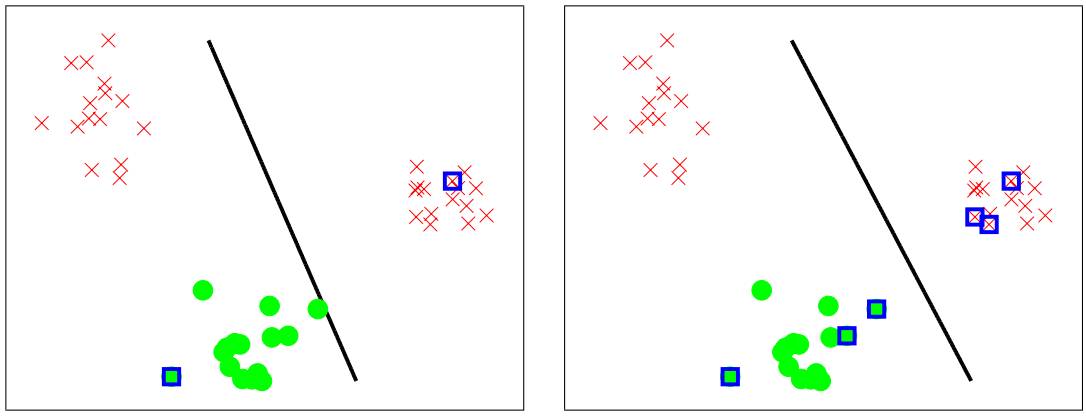
Using the initial $\mathcal{D}_l \cup \mathcal{D}_u$, the learning algorithm $\mathcal{L}$ is first called to learn a decision function $f^{(0)} \colon \mathbf{R}^d \to \mathbf{R}$, where the function $\text{sign}(f^{(0)}(\mathbf{x}))$ is taken for predicting the label of any input vector $\mathbf{x}$. Then, in iteration r, where $r = 1, 2, ..., R$, the querying algorithm $\mathcal{Q}$ is allowed to select an instance $\widetilde{\mathbf{x}}_s \in \mathcal{D}_u$ and query its label y_s from a labeling oracle. After querying, $(\widetilde{\mathbf{x}}_s, y_s)$ is added to the labeled pool $\mathcal{D}_l$ and $\widetilde{\mathbf{x}}_s$ is removed from the unlabeled pool $\mathcal{D}_u$. The learning algorithm $\mathcal{L}$ then learns a decision function $f^{(r)}$ from the updated $\mathcal{D}_l \cup \mathcal{D}_u$. The goal of active learning is to use the limited querying and learning opportunities properly to obtain a decent list of decision functions $[f^{(1)}, f^{(2)}, ..., f^{(R)}]$ that can achieve low out-of-sample (test) error rates.

As discussed in a detailed survey (Settles, 2009), there are many active learning algorithms for binary classification. In this paper, we shall review some relevant and representative ones. One of the most intuitive families of algorithms is called uncertainty sampling (Lewis and Gale, 1994). As the name suggests, the querying algorithm $\mathcal{Q}$ of uncertainty sampling queries the most uncertain $\widetilde{\mathbf{x}}_s \in \mathcal{D}_u$, where the uncertainty for each input vector $\widetilde{\mathbf{x}}_j \in \mathcal{D}_u$ is usually computed by re-using the decision function $f^{(r-1)}$ returned from the learning algorithm $\mathcal{L}$. For instance, Tong and Koller (2000) take the support vector machine (SVM; Vapnik, 1998) as $\mathcal{L}$ and measure the uncertainty of $\widetilde{\mathbf{x}}_j$ by the distance between $\widetilde{\mathbf{x}}_j$ and the boundary $f^{(r-1)} = 0$. In other words, the algorithm in Tong and Koller (2000) queries the $\widetilde{\mathbf{x}}_s$ that is closest to the boundary.

Uncertainty sampling can be viewed as a greedy approach that queries instances from the viewpoint only of the decision function $f^{(r-1)}$. When the decision function is not close enough to the ideal one, however, this limited viewpoint can hinder the performance of the active learning algorithm. Thus, Cohn et al. (1996) suggest that the viewpoint of the unlabeled pool $\mathcal{D}_u$ should also be included. Their idea leads to another family of active learning algorithms, called representative sampling (Xu et al., 2003), or density-weighted sampling (Settles, 2009). Representative sampling takes both the uncertainty and the representativeness of each $\widetilde{\mathbf{x}}_j \in \mathcal{D}_u$ into account concurrently in the querying algorithm $\mathcal{Q}$, where the representativeness of $\widetilde{\mathbf{x}}_j$ with respect to $\mathcal{D}_u$ is measured by the density of its neighborhood area. For instance, Xu et al. (2003) employ the SVM as the learning algorithm $\mathcal{L}$ as do Tong and Koller (2000). They use a querying algorithm $\mathcal{Q}$ that first clusters the unlabeled instances near the boundary of $f^{(r-1)}$ by a K-means algorithm, and then queries one of the centers of those clusters. In other words, the queried instance is not only uncertain for $f^{(r-1)}$ but also representative for $\mathcal{D}_u$. Some other works estimate the representativeness with a generative model. For instance, Nguyen and Smeulders (2004) propose a querying algorithm $\mathcal{Q}$ that uses multiple Gaussian distributions to cluster all input vectors $\mathbf{x}_i \in \mathcal{D}_l$, $\widetilde{\mathbf{x}}_j \in \mathcal{D}_u$ and estimate the prior probability $p(\mathbf{x})$; $\mathcal{Q}$ then makes querying decisions based on the product of the prior probability and some uncertainty measurement. The idea of estimating the representativeness via clustering is a core element of many representative sampling algorithms (Xu et al., 2003; Nguyen and Smeulders, 2004; Dasgupta and Hsu, 2008). Nevertheless, clustering is a challenging task and it is not always easy to achieve satisfactory clustering performance. When the clustering performance is not satisfactory, it has been observed (Donmez et al., 2007; Huang et al., 2010) that representative sampling algorithms could fail to achieve decent performance. In other words, the clustering step is usually the bottleneck of representative sampling.

Huang et al. (2010) propose an improved algorithm that models representativeness without clustering. In the algorithm, the usefulness of each $\widetilde{\mathbf{x}}_j$, which implicitly contains both uncertainty and representativeness, is estimated by using a technique in semi-supervised learning (Hoi et al., 2008) that checks approximately all possible label assignments for each unlabeled $\widetilde{\mathbf{x}}_j \in \mathcal{D}_u$. The querying algorithm $\mathcal{Q}$ proposed (Huang et al., 2010) is based on the usefulness of each $\widetilde{\mathbf{x}}_j$; the learning algorithm $\mathcal{L}$ is simply a stand-alone SVM. While the active learning algorithm (Huang et al., 2010) often achieves promising empirical results, its bottleneck is the label-estimation step, which is rather sophisticated and thus not always leading to satisfactory performance.

Another improvement of representative sampling is presented by Donmez et al. (2007), who report that representative sampling is less efficient than uncertainty sampling for later iterations, in which the decision function is closer to the ideal one. To combine the best properties of uncertainty sampling and representative sampling, Donmez et al. (2007) propose a mixed algorithm by extending representative sampling (Nguyen and Smeulders, 2004). The proposed query algorithm $\mathcal{Q}$ (Donmez et al., 2007) is split into two stages. The first stage performs representative sampling (Nguyen and Smeulders, 2004) while estimating the expected error reduction. When the expected reduction is smaller than a given threshold, the querying algorithm $\mathcal{Q}$ switches to uncertainty sampling for fine-tuning the decision boundary. The bottleneck of the algorithm (Donmez et al., 2007) is still the clustering step in the first stage.



(a) the decision function (black) obtained from two labeled (blue) instances

(b) when using the decision function in (a) for uncertainty sampling, the top-left cluster keeps being ignored

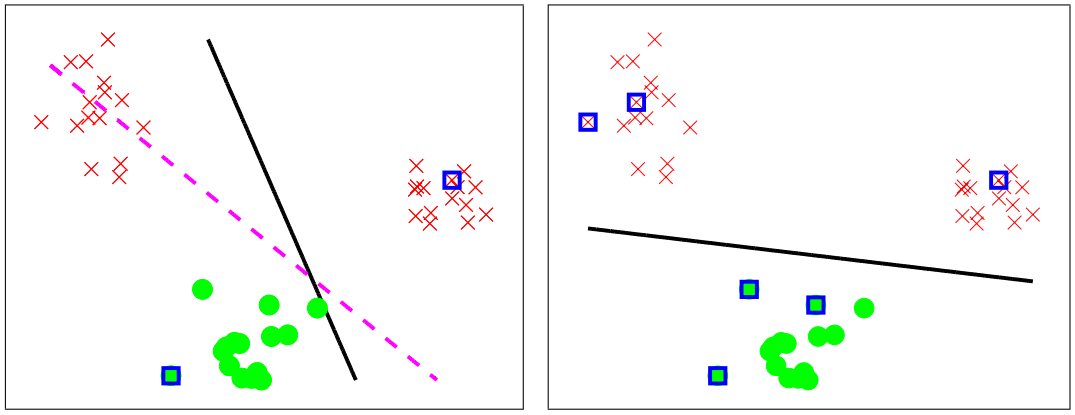
Figure 1: illustration of uncertainty sampling

Another simple algorithm employs transductive SVM (TSVM; Joachims, 1999b) to replace SVM for uncertainty sampling (Wang et al., 2011). Note that TSVM aims to estimate the labels of unlabeled data to maximize its margin, which is similar to the algorithm proposed by Huang et al. (2010). Therefore, using TSVM to replace SVM in uncertainty sampling for querying can also be viewed a concrete instance of representative sampling. We will have more detailed discussions of the difference between the two algorithms in Section 4.

3 Hinted Sampling Framework

Instead of facing the challenges of either clustering or label-estimation, we propose to view the information in $\mathcal{D}_u$ differently. In particular, the unlabeled instances $\widetilde{\mathbf{x}}_j \in \mathcal{D}_u$ are taken as hints (Abu-Mostafa, 1995) that guide the querying algorithm $\mathcal{Q}$. The idea of using hints leads to a simpler active learning algorithm with better empirical performance.

First, we illustrate the potential drawback of uncertainty sampling with a linear SVM classifier (Vapnik, 1998), which is applied to a two-dimensional artificial dataset. Figure 1 shows the artificial dataset, which consists of three clusters, each of which contains instances of a particular class. We denote one class by a red cross and the other by a filled green circle. The labeled instances in $\mathcal{D}_l$ are marked with a blue square while other instances are in $\mathcal{D}_u$. In Figure 1(a), the initial two labeled instances reside in two of the clusters with different labels. The initial decision function $f^{(0)}$ trained on the labeled instances (from the two clusters) is not aware of the third cluster. The decision function $f^{(0)}$ then mis-classifies the instances in the third cluster, and causes the querying algorithm $\mathcal{Q}$ (which is based on $f^{(0)}$) to query only from the instances near the "wrong" boundary rather than exploring the third cluster. After several iterations, as shown in Figure 1(b), the uncertainty sampling algorithm still outputs an unsatisfactory



(a) the hinted query boundary (dashed magenta) that is aware of the top-left cluster

(b) when using the hinted query function in (a) for uncertainty sampling, all three clusters are explored

Figure 2: illustration of hinted sampling

decision function that mis-classifies the entire unqueried (third) cluster.

The unsatisfactory performance of uncertainty sampling originates in its lack of awareness of candidate unlabeled instances that should be queried. When trained on only a few labeled instances, the resulting (linear) decision function is overly confident about the unlabeled instances that are far from the boundary. Intuitively, uncertainty sampling could be improved if the querying algorithm $\mathcal{Q}$ were aware of and less confident about the unqueried regions. Both clustering (Nguyen and Smeulders, 2004) and label-estimation (Huang et al., 2010) are based on this intuition, but they explore the unlabeled regions in a rather sophisticated way.

We propose a simpler alternative as follows. Note that the uncertainty sampling algorithm measures the uncertainty by the distance between instances and the boundary. In order to make $\mathcal{Q}$ less confident about the unlabeled instances, we seek a "query boundary" that not only classifies the labeled instances correctly but also passes through the unqueried regions, denoted by the dashed magenta line in Figure 2(a). Then, in the later iterations, the query algorithm $\mathcal{Q}$, using the query boundary, would be less confident about the unqueried regions, and thus be able to explore them. The instances in the unqueried regions give hints as to where the query boundary should pass. Using these hints about the unqueried regions, the uncertainty sampling algorithm can take both uncertainty and the underlying distribution into account concurrently, and achieve better performance, as shown in Figure 2(b).

Based on this idea, we propose a novel active learning framework, *hinted sampling*. The learning algorithm $\mathcal{L}$ in hinted sampling is similar to that in uncertainty sampling, but the querying algorithm is different. In particular, the querying algorithm $\mathcal{Q}$ is provided with some unlabeled instances, called the hint pool $\mathcal{D}_h \subseteq \mathcal{D}_u$. When properly using the information in the hint pool $\mathcal{D}_h$, both uncertainty and representativeness can be considered concurrently to obtain a query boundary that assists $\mathcal{Q}$ in making query decisions. We sketch the framework of *Active Learning under Hinted Sampling* (ALHS)

in Algorithm 1.

Algorithm 1 Active Learning under Hinted Sampling (ALHS) framework

Input: the number of rounds R; a labeled pool $\mathcal{D}_l$; an unlabeled pool $\mathcal{D}_h$; parameters $\theta_{\mathcal{Q}}$ for querying algorithm and $\theta_{\mathcal{L}}$ for learning algorithm
Output: decision functions $f^{(1)}, ..., f^{(R)}$
For $r \leftarrow 1$ **to** R **do**

Select $\mathcal{D}_h$ from $\mathcal{D}_u$

$h \leftarrow \mathcal{Q}(\theta_{\mathcal{Q}}, \mathcal{D}_h \cup \mathcal{D}_l)$

$(\widetilde{\mathbf{x}}_s, y_s) \leftarrow Query(h, \mathcal{D}_u)$

$\mathcal{D}_u \leftarrow \mathcal{D}_u \setminus \widetilde{\mathbf{x}}_s;\ \mathcal{D}_l \leftarrow \mathcal{D}_l \cup (\widetilde{\mathbf{x}}_s, y_s)$

$f^{(r)} \leftarrow \mathcal{L}(\theta_{\mathcal{L}}, \mathcal{D}_l)$

End

Next, we design a concrete active learning algorithm of hinted sampling based on SVM, which is also used as the core of many state-of-the-art algorithms (Tong and Koller, 2000; Xu et al., 2003; Huang et al., 2010), as both $\mathcal{L}$ and $\mathcal{Q}$. Before illustrating the complete algorithm, we show how SVM can be appropriately extended to use the information in $\mathcal{D}_h$ for $\mathcal{Q}$.

3.1 HintSVM

The extended SVM is called HintSVM, which takes hints into account. The goal of HintSVM is to locate a query boundary which does well on two objectives: (1) classifying labeled instances in $\mathcal{D}_l$, and (2) being close to the unlabeled instances in hint pool $\mathcal{D}_h$. Note that the two objectives are different from the usual semi-supervised SVM (Bennett and Demiriz, 1998) such as transductive SVM (Joachims, 1999b), which pushes the unlabeled instances away from the decision boundary.

The first objective matches an ordinary support vector classification (SVC) problem. To deal with the second objective, we consider ϵ-support vector regression (ϵ-SVR) and set regression targets to 0 for all instances in $\mathcal{D}_h$, which means that instances in $\mathcal{D}_h$ should be close to the query boundary. By combining the objective functions of SVC and ϵ-SVR together, HintSVM solves the following convex optimization problem,

which simultaneously achieves the two objectives.

$$
\begin{aligned}
\min_{\mathbf{w},b,\xi,\widetilde{\xi},\widetilde{\xi}^*} \quad & \frac{1}{2}\mathbf{w}^T\mathbf{w} + C_l\sum_{i=1}^{|\mathcal{D}_l|}\xi_i + C_h\sum_{j=1}^{|\mathcal{D}_h|}\left(\widetilde{\xi}_j + \widetilde{\xi}_j^*\right) \\
\text{subject to} \quad & \begin{array}{rcll} y_i(\mathbf{w}^T\mathbf{x}_i + b) & \geq & 1-\xi_i & \text{for } (\mathbf{x}_i, y_i) \in \mathcal{D}_l, \\ \mathbf{w}^T\widetilde{\mathbf{x}}_j + b & \leq & \epsilon + \widetilde{\xi}_j & \text{for } \mathbf{x}_j \in \mathcal{D}_h, \\ -(\mathbf{w}^T\widetilde{\mathbf{x}}_j + b) & \leq & \epsilon + \widetilde{\xi}_j^* & \text{for } \mathbf{x}_j \in \mathcal{D}_h, \\ \xi_i & \geq & 0 & \text{for } (\mathbf{x}_i, y_i) \in \mathcal{D}_l, \\ \widetilde{\xi}_j, \widetilde{\xi}_j^* & \geq & 0 & \text{for } \mathbf{x}_j \in \mathcal{D}_h. \end{array}
\end{aligned} \tag{1}
$$

Here ϵ is the margin of tolerance for being close to the boundary, and C_l, C_h are the weights of the classification errors (on $\mathcal{D}_l$) and hint errors (on $\mathcal{D}_h$), respectively. Similar to the usual SVC and ϵ-SVR, the convex optimization problem can be transformed to the dual form to allow using the kernel trick. Define $\hat{\mathbf{x}}_i = \mathbf{x}_i$, $\hat{\mathbf{x}}_{|\mathcal{D}_l|+j} = \hat{\mathbf{x}}_{|\mathcal{D}_l|+|\mathcal{D}_h|+j} = \widetilde{\mathbf{x}}_j$, $\hat{y}_i = y_i$, $\hat{y}_{|\mathcal{D}_l|+j} = 1$, and $\hat{y}_{|\mathcal{D}_l|+|\mathcal{D}_h|+j} = -1$ for $1 \leq i \leq |\mathcal{D}_l|$ and $1 \leq j \leq |\mathcal{D}_h|$. The dual problem of (1) can be written as follows:

$$
\begin{aligned}
\min_{\alpha} \quad & \tfrac{1}{2}\alpha^T Q\alpha + \mathbf{p}^T\alpha \\
\text{subject to} \quad & \hat{\mathbf{y}}^T\alpha = 0, \\
& 0 \leq \alpha_i \leq C_l \quad \text{for } i = 1, 2, \cdots, |\mathcal{D}_l|, \\
& 0 \leq \alpha_j \leq C_h \quad \text{for } j = |\mathcal{D}_l|+1, \cdots, |\mathcal{D}_l| + 2|\mathcal{D}_h|,
\end{aligned}
$$

where $p_i = -1$, $p_j = \epsilon$, and $Q_{ab} = \hat{y}_a\hat{y}_b\hat{\mathbf{x}}_a^T\hat{\mathbf{x}}_b$. The derived dual form can be easily solved by any state-of-the-art quadratic programming solver, such as the one implemented in LIBSVM (Chang and Lin, 2011).

3.2 Hint Selection Strategy

A naïve strategy for selecting a proper hint pool $\mathcal{D}_h \subseteq \mathcal{D}_u$ is to directly let $\mathcal{D}_h = \mathcal{D}_u$, which retains all the information about the unlabeled data. However, given that the size of $\mathcal{D}_u$ is usually much larger than the size of $\mathcal{D}_l$, this strategy may cause the hints to overwhelm HintSVM, which leads to performance and computational concerns. In our earlier version of this work (Li et al., 2012), a specifically designed selection strategy that drops hints with radial functions have been studied. We have conducted some broader studies that suggest the sufficiency of using a simpler uniform sampling strategy. Next, we will use the strategy to demonstrate the essence, validity and usefulness of hint information.

3.3 Hint Influence Control

Settles (2009) shows that uncertainty sampling can outperform uniform sampling when enough examples have been queried. Thus, after querying more examples, there can be advantages by changing the focus of the active learning approach to "refine" the boundary by uncertainty sampling. (Donmez et al., 2007) tries to dynamically balance the representative sampling and uncertainty sampling. Our earlier work (Li et al., 2012)

exploits two strategies, *hint dropping* and *hint termination*, to achieve this goal. Similar ideas have also been widely used in the bandit problem (Langford and Zhang, 2007) to balance exploration and exploitation. Here we take a simple alternative by multiplying the parameter C_h by a ratio δ each iteration, where $0 < \delta < 1$, to gradually change our focus to uncertainty sampling. That is, in iteration r, the cost parameter $C_h^{(r)}$ of hint instances is $C_h^{(1)} \times \delta^{r-1}$. After enough many iterations, $C_h^{(r)}$ will be closed to 0, which essentially transforms hinted sampling to typical uncertainty sampling.

3.4 Hinted Sampling with HintSVM

Next, we incorporate the proposed ALHS with the derived HintSVM formulation to make a novel active learning algorithm, ALHS-SVM.

The querying algorithm $\mathcal{Q}$ of ALHS-SVM selects unlabeled instances from the unlabeled pool $\mathcal{D}_u$ as the hint pool $\mathcal{D}_h$ and trains HintSVM from $\mathcal{D}_l$ and $\mathcal{D}_h$ to obtain the query boundary for uncertainty sampling. The use of both $\mathcal{D}_l$ and $\mathcal{D}_h$ combines uncertainty and representativeness. The learning algorithm $\mathcal{L}$ of ALHS-SVM, on the other hand, trains a stand-alone SVM from $\mathcal{D}_l$ to get a decision function $f^{(r)}$, just like $\mathcal{L}$ in uncertainty sampling (Tong and Koller, 2000). The full ALHS-SVM algorithm is listed in Algorithm 2.

Algorithm 2 The ALHS-SVM algorithm

Input: the number of rounds R; a labeled pool $\mathcal{D}_l$; an unlabeled pool $\mathcal{D}_u$; parameters for HintSVM and SVM; ratio δ
Output: decision functions $f^{(1)}, ..., f^{(R)}$
For $r \leftarrow 1$ **to** R **do**

Uniformly select $\mathcal{D}_h$ from $\mathcal{D}_u$

$h \leftarrow Train$ HintSVM$(C_h, C_l, \epsilon, \mathcal{D}_h, \mathcal{D}_l)$

$(\widetilde{\mathbf{x}}_s, y_s) \leftarrow Query(h, \mathcal{D}_u)$

$\mathcal{D}_u \leftarrow \mathcal{D}_u \setminus \widetilde{\mathbf{x}}_s;\ \mathcal{D}_l \leftarrow \mathcal{D}_l \cup (\widetilde{\mathbf{x}}_s, y_s)$

$f^{(r)} \leftarrow Train$ SVM$(C, \mathcal{D}_l)$

$C_h \leftarrow C_h \times \delta$

End

Uncertainty sampling with SVM is a special case of ALHS-SVM when always setting $C_h = 0$. In other words, ALHS can be viewed as a generalization of uncertainty sampling that considers representativeness through the hints. The simple use of hints avoids the challenges in clustering or label-estimation steps.

Table 1: Comparison on accuracy (mean $\pm$ se) after querying 5% of unlabeled pool

	Algorithms (%), the highest accuracy for each dataset is in boldface					
data	UNCERTAIN	REPRESENT	QUIRE	DUAL	TSVM-SVM	ALHS-SVM
$australian$	82.188 ± 1.571	$\mathbf{83.739 \pm 0.548}$	82.319 ± 1.126	81.304 ± 0.647	78.116 ± 1.490	83.362 ± 0.439
$diabetes$	63.229 ± 2.767	66.758 ± 0.505	66.771 ± 0.960	65.143 ± 0.381	67.148 ± 0.570	$\mathbf{68.438 \pm 0.709}$
$german$	69.060 ± 0.497	67.240 ± 1.099	68.750 ± 0.605	$\mathbf{69.620 \pm 1.323}$	68.160 ± 0.382	69.330 ± 0.373
$letterMvsN$	89.632 ± 1.103	83.463 ± 1.348	81.372 ± 1.693	83.437 ± 1.211	80.051 ± 1.165	$\mathbf{91.112 \pm 0.444}$
$letterVvsY$	79.245 ± 1.176	63.523 ± 2.335	68.516 ± 2.132	76.213 ± 1.549	71.258 ± 0.973	$\mathbf{79.300 \pm 0.695}$
$segment$	95.437 ± 0.367	94.390 ± 0.482	96.074 ± 0.224	86.078 ± 2.834	79.160 ± 0.716	$\mathbf{96.095 \pm 0.204}$
$splice$	74.430 ± 0.606	69.117 ± 1.452	70.340 ± 0.942	56.969 ± 0.576	72.847 ± 0.332	$\mathbf{76.334 \pm 0.315}$
$wdbc$	93.842 ± 3.137	95.616 ± 0.711	96.613 ± 0.230	96.056 ± 0.250	95.777 ± 0.264	$\mathbf{97.020 \pm 0.123}$

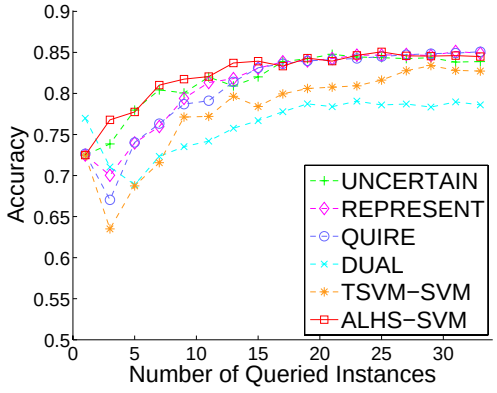
3.5 Experimental Studies of ALHS-SVM

Next, we compared the proposed ALHS-SVM algorithm with the following active learning algorithms: (1) UNCERTAIN (Tong and Koller, 2000): uncertainty sampling with SVM, (2) REPRESENT (Xu et al., 2003): representative sampling with SVM and clustering, (3) DUAL (Donmez et al., 2007): mixture of uncertainty and representative sampling, (4) QUIRE (Huang et al., 2010): representative sampling with label estimation based on the min-max view, We also list the results of another related active learning algorithm, TSVM-SVM (Wang et al., 2011), which conducts representative sampling with transductive SVM, and will compare it with ALHS-SVM in detail in Section 4.

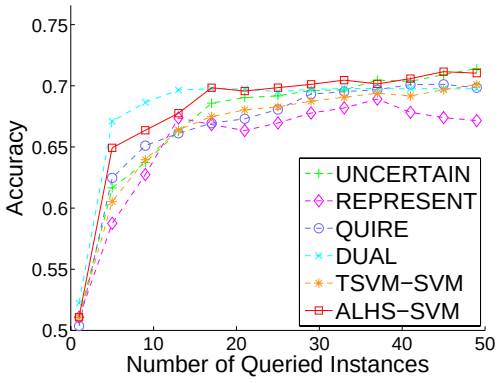
We conducted experiments on eight UCI benchmarks (Frank and Asuncion, 2010), which are $australian$, $diabetes$, $german$, $splice$, $wdbc$, $letetrMvsN$, $letterVvsY$ (Donmez et al., 2007; Huang et al., 2010) and $segment\text{-}binary$ (Ratsch et al., 2001; Donmez et al., 2007) as chosen by other related works. For each dataset, we randomly divided it into two parts with equal size. One part was treated as the unlabeled pool $\mathcal{D}_u$ for active learning algorithms. The other part was reserved as the test set. Before querying, we randomly select one positive instance and one negative instance to form the labeled pool $\mathcal{D}_l$. For each dataset, we ran the algorithms 20 times with different random splits.

Due to the difficulty of locating the best parameters for each active learning algorithms in practice, we chose to compare all algorithms on fixed parameters. In the experiments, We adapt the implementation in SVM-light (Joachims, 1999a) for TSVM and LIBSVM (Chang and Lin, 2011) for other SVM-based algorithms with the RBF kernel and the default parameters, except for $C = 5$. Correspondingly, the parameter λ in the works of Donmez et al. (2007) and Huang et al. (2010) was set to $\lambda = \frac{1}{C}$. These parameters ensure that all four algorithms behave in a stable manner. For ALHS-SVM, we fixed $\delta = 0.5$ and uniformly sample 10% data from $\mathcal{D}_u$ as $\mathcal{D}_h$ without any further tuning for each dataset. For other algorithms, we take the parameters in the original papers.

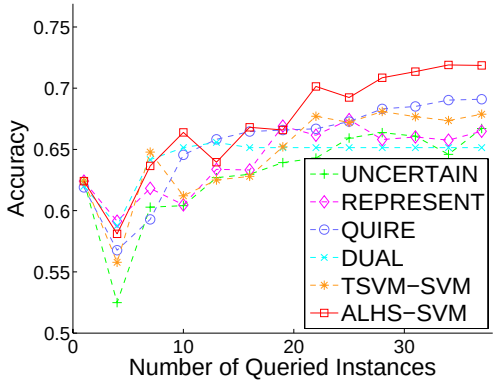
Figure 3 presents the accuracy of different active learning algorithms along with the number of rounds R, which equals the number of queried instances. Tables 1 and 2 list the mean and standard error of accuracy when $R = |\mathcal{D}_u| \times 5\%$ and $R = |\mathcal{D}_u| \times 10\%$, respectively. The highest mean accuracy is shown in boldface for each dataset. We also conducted the t-test at 95% significance level (Melville and Mooney, 2004; Guo



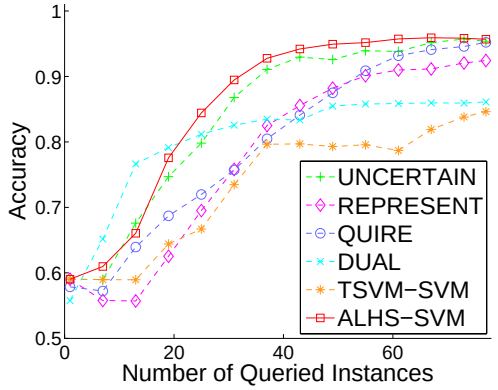
(a) *australian*



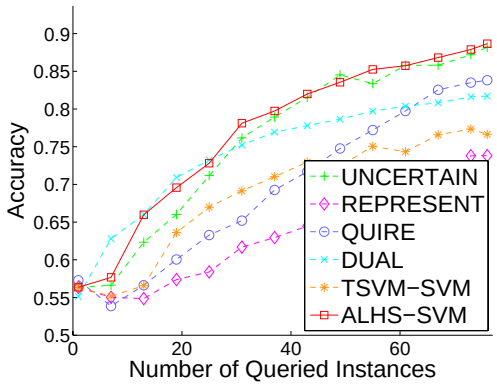
(b) *german*



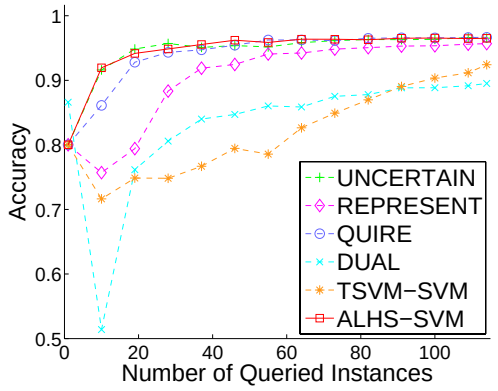
(c) *diabetes*



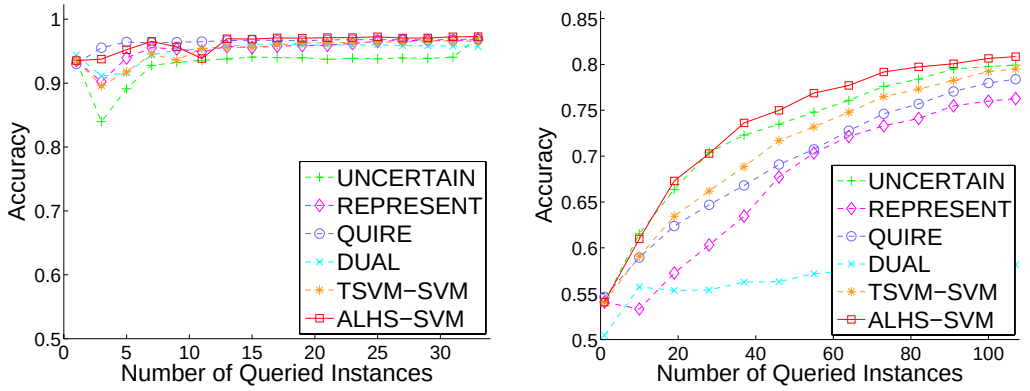
(d) *leterMvsN*



(e) *letterVvsY*



(f) *segment*



(g) *wdbc* (h) *splice*

Figure 3: Comparison on different datasets

Table 2: Comparison on accuracy (mean $\pm$ se) after querying 10% of unlabeled pool

	Algorithms (%), the highest accuracy for each dataset is in boldface					
data	UNCERTAIN	REPRESENT	QUIRE	DUAL	TSVM-SVM	ALHS
australian	83.884 ± 0.460	$\mathbf{84.884 \pm 0.367}$	84.870 ± 0.455	81.174 ± 0.798	82.725 ± 0.628	84.4391 ± 0.259
diabetes	66.706 ± 2.632	66.484 ± 1.223	67.500 ± 1.337	65.143 ± 0.381	67.878 ± 0.609	$\mathbf{71.758 \pm 0.618}$
german	$\mathbf{71.410 \pm 0.488}$	67.150 ± 0.773	70.250 ± 0.560	69.760 ± 0.299	70.110 ± 0.342	71.030 ± 0.257
letterMvsN	95.369 ± 0.315	92.433 ± 0.777	95.114 ± 0.486	86.893 ± 0.870	84.625 ± 0.879	$\mathbf{95.578 \pm 0.214}$
letterVvsY	88.213 ± 0.635	73.806 ± 1.551	84.723 ± 0.891	80.123 ± 1.359	76.645 ± 0.677	$\mathbf{88.561 \pm 0.431}$
segment	96.528 ± 0.143	95.684 ± 0.155	$\mathbf{96.658 \pm 0.110}$	89.519 ± 1.760	92.420 ± 0.217	96.445 ± 0.100
splice	79.931 ± 0.274	76.274 ± 0.895	78.560 ± 0.648	58.947 ± 0.853	79.503 ± 0.197	$\mathbf{80.750 \pm 0.143}$
wdbc	97.155 ± 0.141	96.818 ± 0.191	96.862 ± 0.206	95.748 ± 0.247	96.554 ± 0.219	$\mathbf{97.170 \pm 0.127}$

and Greiner, 2007; Donmez et al., 2007). The t-test results are given in Table 3, which summarizes the number of datasets in which ALHS-SVM performs significantly better (or worse) than the other algorithms.

Comparison between ALHS-SVM and Uncertainty Sampling For some datasets, such as *wdbc* and *diabetes* in Figure 3(g) and 3(c), the result for UNCERTAIN is unsatisfactory. This unsatisfactory performance is possibly caused by the lack of awareness of unlabeled instances, which echoes our illustration in Figure 1. Note that we have considered some more aggressive querying criteria (Tong and Koller, 2000) than UNCERTAIN on the side, and have observed that those criteria are designed for hard-margin SVM and hence can be worse than UNCERTAIN with soft-margin SVM in our experiments. Thus, we excluded them from the tables. In these two cases, ALHS-SVM surely improves on UNCERTAIN with much lower standard error by using the hint information to avoid the worse local optimal. The results demonstrate the validity of the proposed ALHS framework. For other datasets which UNCERTAIN performs well on them, ALHS-SVM is still competitive and can sometimes reach even better performance. For instance, in *splice*, ALHS-SVM results in significantly higher accuracy after 30 queries. The observation further justifies that the hint information can be useful in boosting the performance of UNCERTAIN.

Table 3: ALHS-SVM versus the other algorithm based on t-test at 95% significance level

	Algorithms (win/tie/loss)				
Percentage of queries	UNCERTAIN	REPRESENT	QUIRE	DUAL	TSVM-SVM
5%	1/7/0	6/2/0	5/3/0	4/4/0	6/2/0
10%	3/5/0	7/1/0	6/2/0	5/3/0	8/0/0

Comparison between ALHS-SVM and Representative Sampling We will leave the detailed comparison to TSVM-SVM in the next section. For DUAL, CLUSTER and QUIRE, we see that ALHS-SVM is only worse than DUAL on *german* and *letter M vs N* when ALHS-SVM has not queried enough many instances. For all other datasets and situations, ALHS-SVM generally results in better performance than all the other three algorithms. For instance, in Figure 3(h), since $splice$ is a larger and higher dimensional dataset, representative sampling algorithms that perform clustering (REPRESENT, DUAL) or label estimation (QUIRE) fail to reach a decent performance. We attribute the results to the fact that it is usually non-trivial to perform distribution estimation, clustering or label estimations in a high dimensional space. On the other hand, ALHS uses the hint information without aggressive assumptions, and can thus result in better and stabler performance.

In summary, in Figure 3 shows that ALHS-SVM can achieve comparable results to those of the best representative sampling and uncertainty sampling algorithms. As shown in Tables 1 and 2, after querying 5% of the unlabeled instances (Table 1), ALHS-SVM achieves the highest mean accuracy in 6 out of 8 datasets; after querying 10% of unlabeled instances (Table 2), ALHS-SVM achieves the highest mean accuracy in 5 out of 8 datasets. Table 3 further confirms that ALHS-SVM usually outperforms each of the other algorithms at the 95% significance level.

4 Transductive SVM versus HintSVM

As discussed, transductive SVM (TSVM; Joachims, 1999b) has been considered for active learning (Wang et al., 2011) as a representative sampling approach. TSVM arises from semi-supervised learning and has been demonstrated to be useful for many applications, such text mining (Joachims, 1999b). TSVM aims to maximize its margin between the labeled data and the unlabeled data by assigning suitable labels to the un-

labeled data. The formulation is as follows.

$$
\begin{aligned}
\min_{\mathbf{w},b,\xi,\bar{\xi},\bar{y}} \quad & \frac{1}{2}\mathbf{w}^T\mathbf{w} + C_l\sum_{i=1}^{|\mathcal{D}_l|}\xi_i + C_u\sum_{j=1}^{|\mathcal{D}_u|}\bar{\xi}_j \\
\text{subject to} \quad & y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1-\xi_i \quad \text{for } (\mathbf{x}_i, y_i)\in\mathcal{D}_l, \\
& \bar{y}_j(\mathbf{w}^T\mathbf{x}_j + b) \geq 1-\bar{\xi}_j \quad \text{for } \mathbf{x}_i\in\mathcal{D}_u, \\
& \xi_i \geq 0 \quad \text{for } (\mathbf{x}_i, y_i)\in\mathcal{D}_l, \\
& \bar{\xi}_j \geq 0 \quad \text{for } \mathbf{x}_j\in\mathcal{D}_u, \\
& \bar{y}_j \in \{+1,-1\} \quad \text{for } \mathbf{x}_j\in\mathcal{D}_u.
\end{aligned} \tag{2}
$$

Existing approach (Wang et al., 2011) uses TSVM for querying, which can be viewed as a form of representative sampling that locates the querying boundary by estimating the labels of unlabeled instances.

Comparing formulation (2) of TSVM and formulation (1) of HintSVM as described in Section 3.1, we see that the formulations share some similarities, but focus on very different objective functions. In this section, we study the validity and effectiveness of these two formulations for active learning as querying and/or learning algorithms.

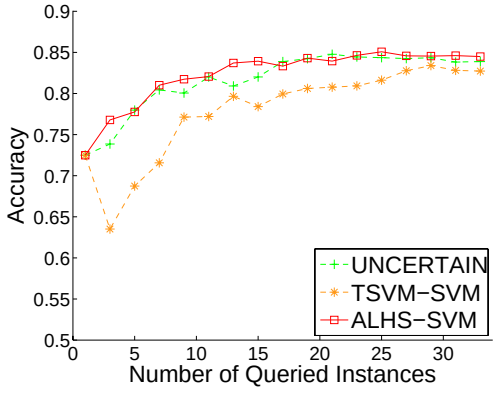
4.1 Comparison between HintSVM and TSVM

Comparison When Using SVM for Learning. We first study the case when using HintSVM and TSVM as the querying algorithm while taking the stand-alone SVM as the learning algorithm. The two algorithms are denoted as ALHS-SVM and TSVM-SVM, respectively. For other experimental settings, we follow the same setup as described in Section 3.5.

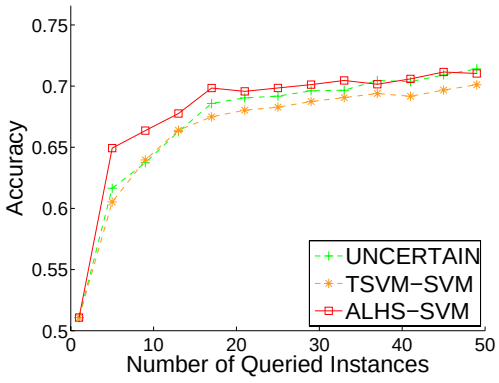
Figure 4 presents the accuracy of TSVM-SVM and ALHS-SVM as well as the baseline UNCERTAIN algorithm with SVM. The mean and standard error of accuracy at different rounds of querying are readily listed in Tables 1 and 2. Clearly, TSVM-SVM performs generally worse than ALHS-SVM across all datasets. The results again justify the usefulness of the proposed ALHS framework. We discuss the performance difference as follows.

In formulation (2), TSVM-SVM aims to estimate the possible labels on the unlabeled data, which is similar to QUIRE Huang et al. (2010). Nevertheless, in QUIRE, the estimation is used for exploring a better query, but TSVM-SVM takes the estimation with a goal of a better classifier. Thus, TSVM-SVM pushes the unlabeled data away from the boundary for better classification ability. Note that ALHS-SVM, on the other hand, aims at a boundary close to parts of unlabeled data (hints) to explore like QUIRE. In the earlier iterations of active learning, exploration rather than pushing the unlabeled data away (as if they are certain) can be important. Thus, TSVM-SVM can be inferior to ALHS-SVM.

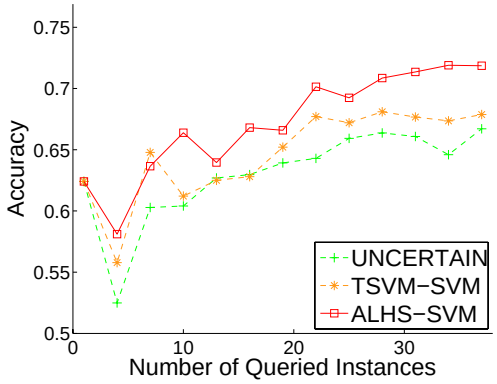
On the other hand, in the latter iterations of active learning, ALHS-SVM is similar to the baseline UNCERTAIN approach, which is known to perform decently when the learning algorithm is SVM. Nevertheless, because the boundaries obtained from TSVM and SVM can be quite different, the instances queried by TSVM may not be "uncertain" for learning with SVM. This explanation matches one interesting observation that TSVM-SVM is also worse than the baseline UNCERTAIN algorithm in many



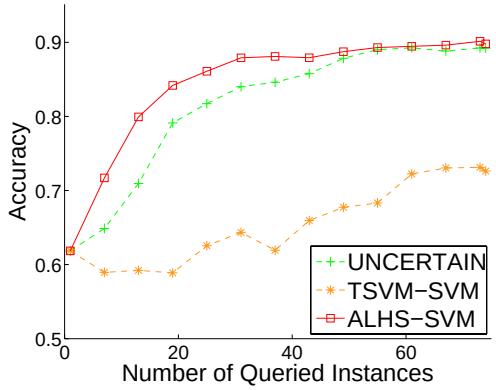
(a) *australian*



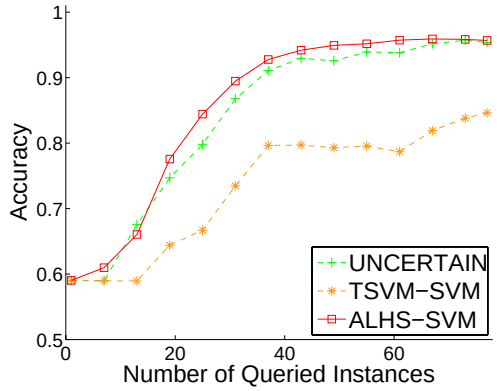
(b) *german*



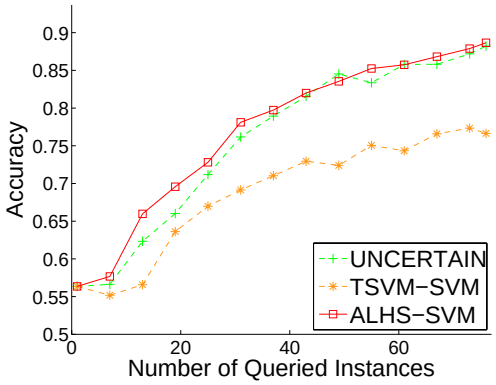
(c) *diabetes*



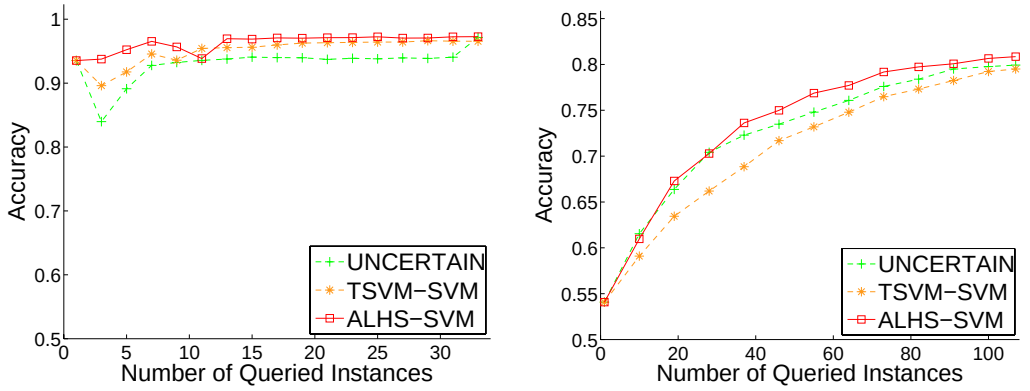
(d) *leterIvsJ*



(e) *leterMvsN*



(f) *letterVvsY*



(g) *wdbc* (h) *splice*

Figure 4: Comparison between different querying algorithms by using SVM for learning on different datasets

datasets such *australian* and *letter I vs J*. Thus, ALHS-SVM also holds an advantage over TSVM-SVM in the latter iterations.

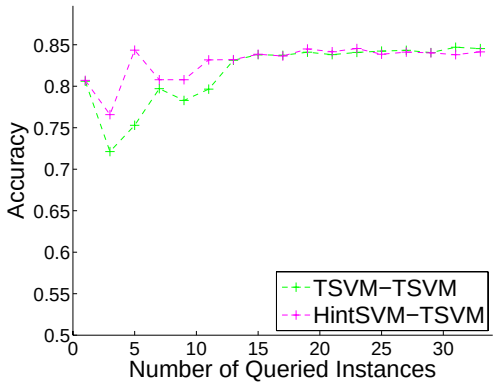
Comparison When Using TSVM for Learning. The discussion above shows that the discrepancy between TSVM and SVM may be part of the reason that TSVM-SVM is inferior for active learning. What if we take TSVM for learning instead? Next, we couple TSVM for learning with two querying approaches: HintSVM or TSVM. The two algorithms are named HintSVM-TSVM and TSVM-TSVM, and the results are shown in Figure 5.

According to Figure 5, HintSVM does not always achieve competitive performance over TSVM with using TSVM as the learning algorithm. The results verify our earlier claim that the discrepancy between TSVM and SVM leads to inferior performance for TSVM-SVM. Compared with TSVM, HintSVM benefits the querying stage for exploration and results in significantly better performance in some datasets, such as *letter M vs N*, *letter I vs J* and *splice*. The exploration makes the learning boundary converge to better local optimal one within a few queries.

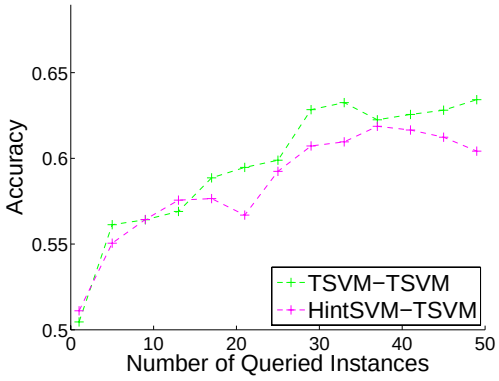
Nevertheless, we observe that HintSVM-TSVM may not result in satisfactory performance in other datasets, such as *german* and *wdbc*. We address this problem to the same reason that TSVM-SVM is inferior: HintSVM, which is based on a stand-along SVM, is very different from TSVM. Thus, uncertain instances queried by HintSVM may not be uncertain to the learning TSVM. Thus, HintSVM-TSVM is yet another combination where the discrepancy between the querying and the learning parts results in unsatisfactory performance.

4.2 Hint Transductive SVM

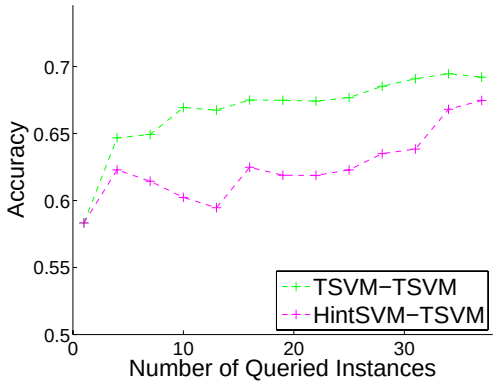
The results in Section 4.1, show that considering the learning algorithm is an important issue when designing the querying algorithm to avoid discrepancy. Similar idea has also been studied in other active learning works (Donmez et al., 2007). Thus, ALHS-TSVM



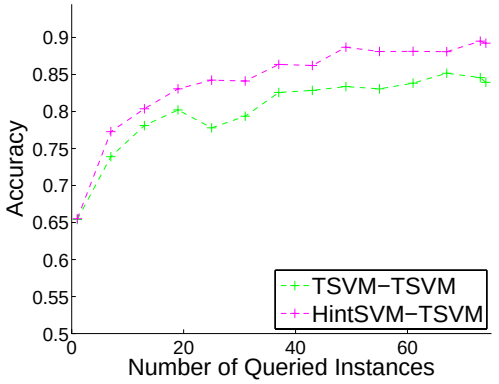
(a) *australian*



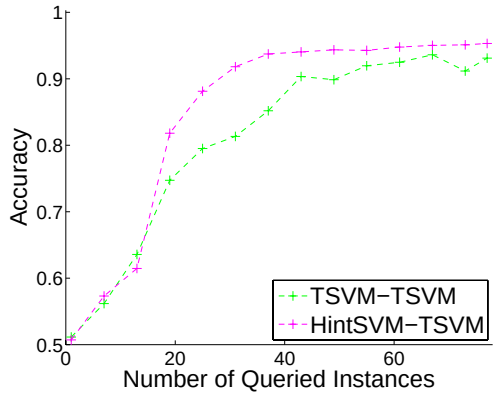
(b) *german*



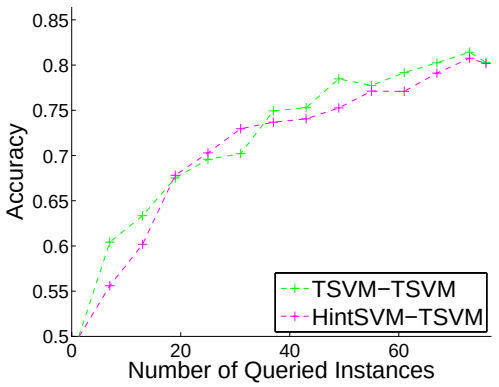
(c) *diabetes*



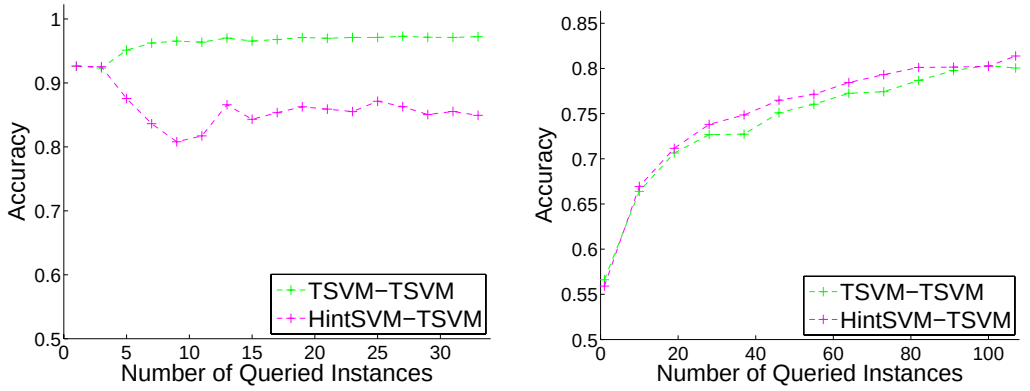
(d) $leterIvsJ$



(e) $leterMvsN$



(f) $letterVvsY$



(g) *wdbc* (h) *splice*

Figure 5: Comparison between TSVM and HintSVM for querying by using TSVM for learning on different datasets

results in inferior performance.

One interesting question is then whether ALHS can be used when employing TSVM as the learning algorithm. Next, we demonstrate one such possibility. We combine formulation (2) of TSVM and formulation (1) to an extension of TSVM with hint information, which is called Hint Transductive SVM (HintTSVM). HintTSVM can then be used for hinted sampling with TSVM, and expected to be a better match of ALHS when employing TSVM as the learning algorithm. The formulation of HintTSVM is as follows:

$$
\begin{aligned}
\min_{\mathbf{w},b,\xi,\bar{\xi},\bar{y}} \quad & \frac{1}{2}\mathbf{w}^T\mathbf{w} + C_l\sum_{i=1}^{|\mathcal{D}_l|}\xi_i + C_u\sum_{j=1}^{|\mathcal{D}'_u|}\bar{\xi}_j + C_h\sum_{j=1}^{|\mathcal{D}_h|}\left(\widetilde{\xi}_j + \widetilde{\xi}^*_j\right) \\
\text{subject to} \quad & \begin{aligned}
y_i(\mathbf{w}^T\mathbf{x}_i + b) &\geq 1 - \xi_i && \text{for } (\mathbf{x}_i, y_i) \in \mathcal{D}_l, \\
\bar{y}_j(\mathbf{w}^T\mathbf{x}_j + b) &\geq 1 - \bar{\xi}_j && \text{for } \mathbf{x}_i \in \mathcal{D}'_u, \\
\mathbf{w}^T\widetilde{\mathbf{x}}_j + b &\leq \epsilon + \widetilde{\xi}_j && \text{for } \mathbf{x}_j \in \mathcal{D}_h, \\
-(\mathbf{w}^T\widetilde{\mathbf{x}}_j + b) &\leq \epsilon + \widetilde{\xi}^*_j && \text{for } \mathbf{x}_j \in \mathcal{D}_h, \\
\xi_i &\geq 0 && \text{for } (\mathbf{x}_i, y_i) \in \mathcal{D}_l, \\
\bar{\xi}_j &\geq 0 && \text{for } \mathbf{x}_j \in \mathcal{D}'_u, \\
\widetilde{\xi}_j, \widetilde{\xi}^*_j &\geq 0 && \text{for } \mathbf{x}_j \in \mathcal{D}_h, \\
\bar{y}_j &\in \{+1, -1\} && \text{for } \mathbf{x}_j \in \mathcal{D}'_u.
\end{aligned}
\end{aligned} \tag{3}
$$

where $\mathcal{D}_u = \mathcal{D}'_u \cup \mathcal{D}_h$.

Different from HintSVM, it is difficult to solve HintTSVM efficiently since the TSVM part of the formulation is NP-hard to solve. Therefore, we consider a simple approximation by splitting the training of HintTSVM into two stages. In the first stage, we only consider training the TSVM part on $\mathcal{D}_l$ and $\mathcal{D}'_u$ by any existing algorithm (Joachims, 1999b). In the second stage, we use the inferred labels and cost parameters C_h from the first stage to train a HintSVM as described in Section 3.1. We call the variant of ALHS with HintTSVM as ALHS-TSVM, and list the details in Algorithm 3.

Algorithm 3 The ALHS-TSVM algorithm

Input: the number of rounds R; a labeled pool $\mathcal{D}_l$; an unlabeled pool $\mathcal{D}_h$; parameters for HintTSVM and TSVM; ratio δ
Output: decision functions $f^{(1)}, ..., f^{(R)}$
For $r \leftarrow 1$ **to** R **do**

Uniformly select $\mathcal{D}_h$ from $\mathcal{D}_u$

$\mathcal{D}'_u \leftarrow \mathcal{D}_u \backslash \mathcal{D}_h$

$h \leftarrow Train$ HintTSVM$(C_h, C_u, C_l, \epsilon, \mathcal{D}_h, \mathcal{D}'_u, \mathcal{D}_l)$

$(\widetilde{\mathbf{x}}_s, y_s) \leftarrow Query(h, \mathcal{D}_u)$

$\mathcal{D}_u \leftarrow \mathcal{D}_u \setminus \widetilde{\mathbf{x}}_s;\ \mathcal{D}_l \leftarrow \mathcal{D}_l \cup (\widetilde{\mathbf{x}}_s, y_s)$

$f^{(r)} \leftarrow Train$ TSVM$(C_l, C_u, \mathcal{D}_l, \mathcal{D}_u)$

$C_h \leftarrow C_h \times \delta$

End

Table 4: ALHS-TSVM versus the other algorithm based on t-test at 95% significance level

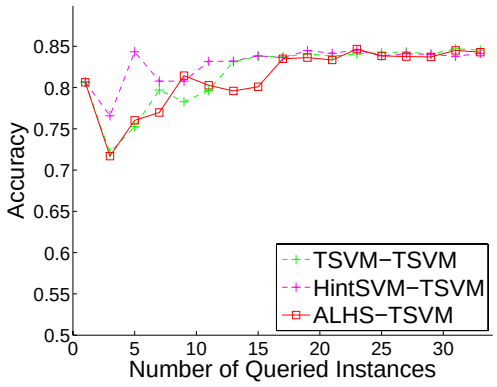
	Algorithms (win/tie/loss)	
Percentage of queries	TSVM-TSVM	HintSVM-TSVM
5%	2/6/0	4/1/3
10%	3/5/0	5/1/3

4.3 Experimental Studies of ALHS-TSVM

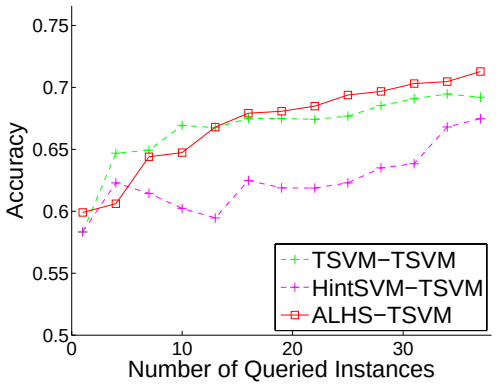
We follow the same experiment setup as previous experiments and report the results in Figure 6 and Table 4.

From the results, we see that the only two datasets that ALHS-TSVM do not perform the strongest are *letter I vs J* and *letter M vs N*. On those datasets, HintSVM-TSVM reaches the best performance. We address this to the difficulty of properly training HintTSVM in ALHS-TSVM, and the simple HintSVM results in more stable performance.

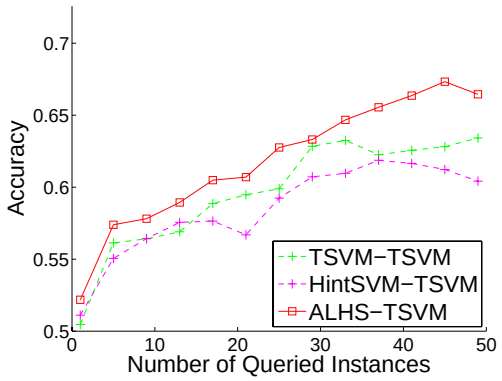
On the other hand, for the datasets that HintSVM-TSVM performs worse than the TSVM-TSVM, such as *german* and *wdbc*, ALHS-TSVM results in better or competitive performance than both of them. The results demonstrate the validity of employing HintTSVM in ALHS-TSVM to explore unknown region of the data to TSVM and resolve the potential drawback of HintSVM-SVM as discussed in Section 4.1.



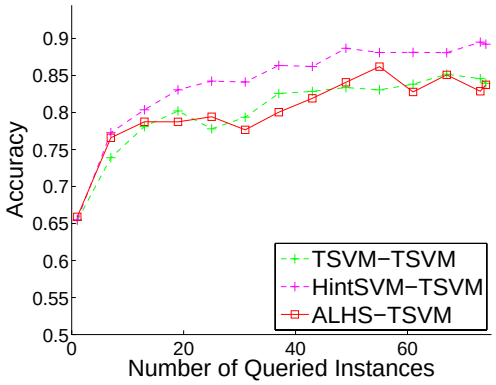
(a) *australian*



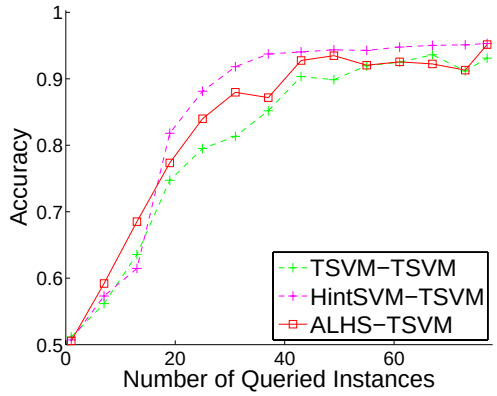
(b) *german*



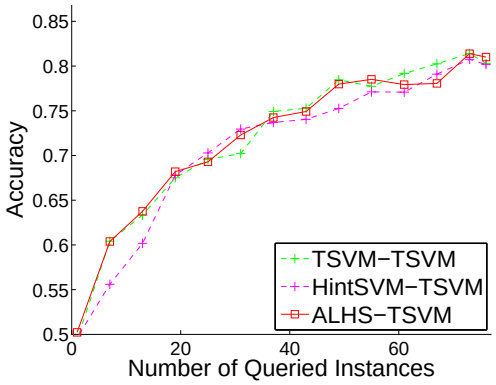
(c) *diabetes*



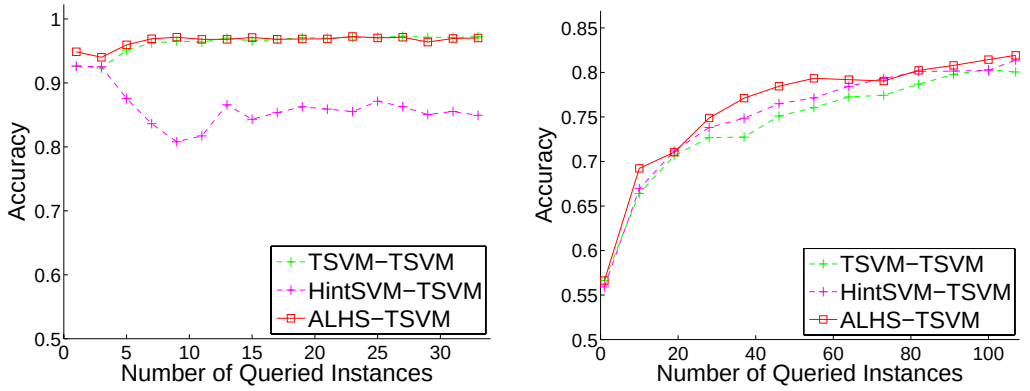
(d) *leterIvsJ*



(e) *leterMvsN*



(f) *letterVvsY*



(g) *wdbc* (h) *splice*

Figure 6: Comparison between TSVM and HintSVM for querying by using TSVM for learning on different datasets

5 Conclusion

We propose a new framework of active learning, *hinted sampling*, which exploits the unlabeled instances as hints. Hinted sampling can take both uncertainty and representativeness into account concurrently in a more natural and simpler way. We design a novel active learning algorithm ALHS within the framework, and couple the algorithm with a promising hint selection strategy. Because ALHS models the representativeness by hints, it avoids the potential problems of other more sophisticated approaches that are employed by other representative sampling algorithms. Hence, ALHS results in a significantly better and more stable performance than other state-of-the-art algorithms, and can be used to immediately improve SVM-based uncertainty sampling and TSVM-based representative sampling.

Due to the simplicity and effectiveness of *hinted sampling*, it is worth studying more about this framework. An intensive research direction is to couple hinted sampling with other classification algorithms, and investigate deeper on the hint selection strategies. While we use SVM in ALHS, this framework could be generalized to other classification algorithms. In the future, we plan to investigate more general hint selection strategies and extend hinted sampling from binary classification to other classification problem.

References

Abu-Mostafa, Y. S. (1995). Hints. *Neural Computation*, 4:639–671.

Bennett, K. P. and Demiriz, A. (1998). Semi-supervised support vector machines. In *Advances in Neural Information Processing Systems 11*, pages 368–374.

Chang, C.-C. and Lin, C.-J. (2011). LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, pages 27:1–27:27.

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

Dasgupta, S. and Hsu, D. (2008). Hierarchical sampling for active learning. In *Proceedings of the 25th International Conference on Machine learning*, pages 208–215.

Donmez, P., Carbonell, J. G., and Bennett, P. N. (2007). Dual strategy active learning. In *Proceedings of the 18th European Conference on Machine Learning*, pages 116–127.

Frank, A. and Asuncion, A. (2010). UCI machine learning repository.

Guo, Y. and Greiner, R. (2007). Optimistic active learning using mutual information. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence*, pages 823–829.

Hoi, S. C. H., Jin, R., Zhu, J., and Lyu, M. R. (2008). Semi-supervised SVM batch mode active learning for image retrieval. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 1–7.

Huang, S.-J., Jin, R., and Zhou, Z.-H. (2010). Active learning by querying informative and representative examples. In *Advances in Neural Information Processing Systems 23*, pages 892–900.

Joachims, T. (1999a). Advances in kernel methods. chapter Making large-scale support vector machine learning practical.

Joachims, T. (1999b). Transductive inference for text classification using support vector machines. In *Proceedings of the Sixteenth International Conference on Machine Learning*.

Langford, J. and Zhang, T. (2007). The epoch-greedy algorithm for contextual multi-armed bandits. In *Advances in Neural Information Processing Systems 20*.

Lewis, D. D. and Gale, W. A. (1994). A sequential algorithm for training text classifiers. In *Proceedings of the 17th ACM International Conference on Research and Development in Information Retrieval*, pages 3–12.

Li, C.-L., Ferng, C.-S., and Lin, H.-T. (2012). Active learning with hinted support vector machine. In *Proceedings of the forth Asian Conference on Machine Learning*.

Melville, P. and Mooney, R. J. (2004). Diverse ensembles for active learning. In *Proceedings of the 21st International Conference on Machine Learning*, pages 584–591.

Nguyen, H. T. and Smeulders, A. (2004). Active learning using pre-clustering. In *Proceedings of the 21st International Conference on Machine Learning*, pages 623–630.

Ratsch, G., Onoda, T., and Müller, K. R. (2001). Soft margins for AdaBoost. *Machine Learning*, 2:27:1–27:27.

Settles, B. (2009). Active learning literature survey. Technical report, University of Wisconsin–Madison.

Tong, S. and Koller, D. (2000). Support vector machine active learning with applications to text classification. In *Proceedings of the 17th International Conference on Machine Learning*, pages 999–1006.

Vapnik, V. (1998). *Statistical learning theory*. Wiley.

Wang, Z., Yan, S., and Zhang, C. (2011). Active learning with adaptive regularization. *Pattern Recogn.*

Xu, Z., Yu, K., Tresp, V., Xu, X., and Wang, J. (2003). Representative sampling for text classification using support vector machines. In *Proceedings of the 25th European Conference on Information Retrieval Research*, pages 393–407.