Efficient Active Learning by Querying Discriminative and Representative Samples and Fully Exploiting Unlabeled Data

Bin Gu, Member, IEEE, Zhou Zhai, Cheng Deng, Student Member, IEEE, and Heng Huang, Member, IEEE

Abstract—Active learning is an important learning paradigm in machine learning and data mining, which aims to train effective classifiers with as few labeled samples as possible. Querying discriminative (informative) and representative samples are the state-of-the-art approach for active learning. Fully utilizing a large amount of unlabeled data provides a second chance to improve the performance of active learning. Although there have been several active learning methods proposed by combining with semisupervised learning, fast active learning with fully exploiting unlabeled data and querying discriminative and representative samples is still an open question. To overcome this challenging issue, in this article, we propose a new efficient batch mode active learning algorithm. Specifically, we first provide an active learning risk bound by fully considering the unlabeled samples in characterizing the informativeness and representativeness. Based on the risk bound, we derive a new objective function for batch mode active learning. After that, we propose a wrapper algorithm to solve the objective function, which essentially trains a semisupervised classifier and selects discriminative and representative samples alternately. Especially, to avoid retraining the semisupervised classifier from scratch after each query, we design two unique procedures based on the path-following technique, which can remove multiple queried samples from the unlabeled data set and add the queried samples into the labeled data set efficiently. Extensive experimental results on a variety of benchmark data sets not only show that our algorithm has a better generalization performance than the state-of-the-art active learning approaches but also show its significant efficiency.

Index Terms—Active learning, maximum mean discrepancy (MMD), path-following algorithm, semisupervised support vector machine.

I. INTRODUCTION

A lot of real-world machine learning applications, labeled samples are very difficult, time-consuming, and sometimes expensive to obtain since it may need human efforts. However, unlabeled samples could be ubiquitous because they can be obtained abundantly and cheaply. For example, unlabeled medical images [1] are widely available in hospital, but labeled medical images normally require experienced radiologists intervention, so are costly. Semisupervised learning (SSL) [2]–[4] and active learning [5]–[7] are the most prominent approaches for utilizing unlabeled samples to improve the performance of classifiers. Specifically, SSL utilizes both labeled and unlabeled data to train a high-quality classifier. Active learning tries to select important unlabeled samples for manually labeling and trains effective classifiers with as few labeled samples as possible. Thus, active learning is an important learning paradigm in machine learning.

Many active learning methods have been proposed in the past decades that query the unlabeled samples based on two fundamental criteria, i.e., informativeness (also known as discriminativeness) and representativeness. The first criterion is to find the most informative samples for the classifier which will shrink the space of candidate classifiers as rapidly as possible. For example, Seung et al. [8] and Freund et al. [9] queried samples based on the principle of maximal disagreement by a committee of classifiers. Wang and Zhou [12], Hoi et al. [16], Tong and Koller [20], and Chen and Krause [21] selected samples according to the maximum uncertainty via the distances to the classification boundaries. Roy and McCallum [10], Guo and Schuurmans [11], Mac Aodha et al. [14], and Zhu et al. [15] selected samples according to the most uncertainty measured by entropy or expected error. The second one aims to find the most representative samples which are a complementary criterion to informativeness. For example, Hoi et al. [16] measured the representativeness of a sample by its similarity to the remaining unlabeled samples. Huang et al. [17], Mackkassy [18], Li et al. [22], and Ienco et al. [23] exploited the clustering structure of unlabeled data and chose the query samples closest...
to the cluster centers. Wang and Ye [19] used the maximum mean discrepancy (MMD) to characterize the representativeness. In addition, there are some works to consider these two criteria simultaneously for better results. For example, Wang et al. [24] proposed a multiple kernel active learning algorithm by incorporating the representative information into informativeness with multiple kernel learning and multikernel variant of MMD. Du et al. [25] fused the informativeness and representativeness by using a radial basis function and a modified best-versus-second-best strategy. Liu et al. [26] proposed to combine the informativeness and representativeness together based on weighted incremental dictionary learning in a deep network. However, this method paid too much attention to the most uncertain samples, and the structures with few labeled samples may be hard to discover. We summarize representative active learning algorithms in Table I. Note that the representativeness is normally used with the informativeness simultaneously.

Fully utilizing a large amount of unlabeled data provides a second chance to improve the performance of active learning. As mentioned previously, SSL is another way of utilizing unlabeled samples to improve the performance of classifiers. As shown in Table I, there have been several active learning algorithms [11], [12], [14]–[16] utilizing SSL to get a further improvement in querying the useful samples. However, there exist several weaknesses to the existing methods as discussed next.

1) Querying informative and representative samples are the state-of-the-art approach for active learning as mentioned earlier. However, most of the existing active learning algorithms via querying the informative and representative samples did not utilize SSL to improve the quality of queried samples, as shown in Table I. Note that Hoi et al. [16] only considered the representativeness of a sample by its similarity to the remaining unlabeled samples, which is weaker than the representativeness measured by MMD\(^1\) or cluster structure.

2) Most active learning algorithms require to retrain the classifier with each new sample being labeled which would degrade the efficiency of active learning significantly. A batch mode active learning strategy [11], [19] that selects the top \(k > 1\) most uncertain samples each time and the warm start strategy that trains a new model based on the previous model can largely alleviate this problem. However, existing active learning algorithms did not utilize both strategies systematically to improve the efficiency of active learning, especially for the procedure of SSL, as shown in Table I.

To the best of our knowledge, fast (batch mode) active learning with fully exploiting unlabeled data (SSL) to query discriminative and representative samples is still an open question.

To overcome this challenging issue, in this article, we propose a new efficient batch mode active learning algorithm (ALDR+) based on querying discriminative and representative samples by fully utilizing the unlabeled data. Specifically, we first provide an active learning risk bound by fully considering the unlabeled samples in characterizing the informativeness and representativeness. Based on the risk bound, we derive our objective function for batch mode active learning. Then, we propose a wrapper algorithm to solve the objective function, which essentially trains a semisupervised classifier and selects discriminative and representative samples alternately. Especially, to avoid retraining the semisupervised classifier from scratch after each query, we design two unique procedures based on the path-following technique, which can remove multiple queried samples from the unlabeled data set and add the queried samples into the labeled data set efficiently. Extensive experimental results on a variety of benchmark data sets not only show that ALDR+ has a better generalization performance than the state-of-the-art active learning approaches but also show the significant efficiency of ALDR+.

<table>
<thead>
<tr>
<th>Name</th>
<th>Reference</th>
<th>Criteria for querying samples</th>
<th>Exploiting unlabeled data</th>
<th>Acceleration</th>
</tr>
</thead>
<tbody>
<tr>
<td>QBC</td>
<td>[8], [9]</td>
<td>MUS based on disagreement</td>
<td>None</td>
<td>No</td>
</tr>
<tr>
<td>ERE</td>
<td>[10]</td>
<td>MUS based on expected error</td>
<td>None</td>
<td>No</td>
</tr>
<tr>
<td>DISC</td>
<td>[11]</td>
<td>MUS based on entropy</td>
<td>None</td>
<td>Yes (SSL)</td>
</tr>
<tr>
<td>MULTI-V</td>
<td>[12], [13]</td>
<td>MUS based on margin</td>
<td>None</td>
<td>Partially (SSL)</td>
</tr>
<tr>
<td>HSE</td>
<td>[14]</td>
<td>MUS based on expected error</td>
<td>None</td>
<td>Yes (SSL)</td>
</tr>
<tr>
<td>ZHU</td>
<td>[15]</td>
<td>MUS based on expected error</td>
<td>None</td>
<td>Yes (SSL)</td>
</tr>
<tr>
<td>S3VM BMAL</td>
<td>[16]</td>
<td>MUS based on margin</td>
<td>SBS</td>
<td>Yes (SSL)</td>
</tr>
<tr>
<td>QUIRE</td>
<td>[17]</td>
<td>MUS based on margin</td>
<td>Cluster structure</td>
<td>No</td>
</tr>
<tr>
<td>EMR</td>
<td>[18]</td>
<td>MUS based on margin</td>
<td>Cluster structure</td>
<td>No</td>
</tr>
<tr>
<td>RMDR</td>
<td>[19]</td>
<td>MUS based on margin</td>
<td>MMD</td>
<td>No</td>
</tr>
<tr>
<td>ALDR+</td>
<td>Our</td>
<td>MUS based on margin</td>
<td>MMD</td>
<td>Yes (SSL)</td>
</tr>
</tbody>
</table>

\(^1\)MMD [19] is a distance measure between two distributions \(p(x)\) and \(q(x)\), which is defined as \(\text{MMD}(p, q) = \sup_{f \in \mathcal{F}} E_{x_1 \sim p(x_1)} f(x_1) - E_{x_2 \sim q(x_2)} f(x_2)\) and \(\mathcal{F}\) is a function class.

A. Contributions

The main contributions of this article are summarized as follows.

1) We propose an enhanced active learning risk bound, which could be better than the one in [19] due to fully
considering the unlabeled data. Based on the enhanced active learning risk bound, we propose our ALDR+ algorithm. To the best of our knowledge, ALDR+ is the first batch mode active learning algorithm to fully exploit unlabeled data in characterizing the informativeness and representativeness.

2) To achieve fast active learning especially for the non-convex SSL (a subproblem of our ALDR+ algorithm), we propose two unique procedures based on the path-following technique, which can remove multiple queried samples from the unlabeled data set and add the queried samples into the labeled data set for obtaining the updated solution of SSL efficiently.

B. Organization

The rest of this article is organized as follows. We provide the enhanced active learning risk bound in Section II. We propose our ALDR+ algorithm in Section III. We present the experimental results in Section IV. Finally, we conclude this article in Section V.

II. ENHANCED ACTIVE LEARNING RISK BOUND

In this section, we first give a brief review of original active learning risk bound in [19] and then propose our enhanced active learning risk bound by fully considering the unlabeled data.

A. Brief Review of Active Learning Risk Bound

Given an underlying data distribution \( p(x, y) \), where \( x \in \mathbb{R}^d \) is the input of a sample and \( y \in \{-1, 1\} \) is the output for binary classification problems, a function space \( \mathcal{F} \), and a loss function \( L(f(x), y) \) on a sample \( (x, y) \), the goal of a learning algorithm is to find a classifier \( f \in \mathcal{F} \) with the property that its expected risk \( \mathcal{R}(f) \) is as small as possible, where \( \mathcal{R}(f) \) is defined as follows:

\[
\mathcal{R}(f) = \mathbb{E}_{(x,y) \sim p(x,y)} L(f(x), y).
\]

However, due to the fact that the distribution \( p(x, y) \) of samples is unknown, it is challenging to minimize the expected risk. In pool-based active learning scenarios [12], we are given a training set \( Q = \{(x_1, y_1), (x_2, y_2), \ldots, (x_q, y_q)\} \) queried by algorithms and an unlabeled set as \( U = \{x_{q+1}, x_{q+2}, \ldots, x_{q+u}\} \) and define \( \hat{S} = Q \cup U, n = q + u \). Thus, we usually consider the empirical risk on the queried sample set defined as follows:

\[
\hat{R}_Q(f) = \frac{1}{q} \sum_{x_i \in Q} L(f(x_i), y_i).
\]

Given a kernel function \( K(x_i, x_j) \), this maps a sample into a reproducing kernel Hilbert space (RKHS) associated with a nonlinear feature mapping function \( \phi(x) \), and the MMD between \( \mathcal{F} \) and \( Q \) is defined as follows, which empirically estimates MMD between \( S \) and \( Q \) in the RKHS. Define the Rademacher complexity of the function class \( \mathcal{F} \) as

\[
\text{Rad}_q(\mathcal{F}) = \mathbb{E}_\sigma \left[ \sup_{f \in \mathcal{F}} \left( \frac{2}{n} \sum_{i=1}^{q} \sigma_i L(f(x_i), y_i) \right) \right].
\]

where \( \sigma_1, \ldots, \sigma_n \) are independent random variables uniformly chosen from \([-1, 1]\), known as Rademacher variables. Wang and Ye [19] provided an active learning risk bound to the expected risk based on the empirical risk on the queried sample set.

Theorem 1 ([19]): Given a kernel function \( K(x_i, x_j) \), this maps a sample into an RKHS associated with a nonlinear feature mapping function \( \phi(x) \). If we have a labeled and unlabeled set \( S \) with the size of \( n \) and a queried sample set \( Q \) with the size of \( q \ll n \), the following holds the probability at least \( 1 - \delta \):

\[
\mathcal{R}(f) \leq \hat{R}_Q(f) + \text{MMD}_\mathcal{F}(S, Q) + C(\mathcal{F}, q, \delta)
\]

where \( C(\mathcal{F}, q, \delta) = ((\text{Rad}_q(\mathcal{F}))/2) + ((\ln(1/\delta))/q)^{1/2} \) is the function class complexity term.

Remark 1: As mentioned in Theorem 1, unlabeled samples are only used in MMD\( _\mathcal{F}(S, Q) \). We hope that unlabeled samples can be further exploited (such as SSL) to improve the active learning risk bound.

B. New Active Learning Risk Bound

In this section, we try to consider the unlabeled sample set \( U \) to improve the active learning risk bound in (4).

First, we consider the empirical risk \( \hat{R}_Q(f) \) in (4). Obviously, it is infeasible to consider unlabeled sample set \( U \) into the empirical risk \( \hat{R}_Q(f) \) because the true loss function on the unlabeled samples in \( U \) is undefined. Here, we assume that there exists a ground-truth label \( \bar{y} \) to each sample \( x_i \) in \( U \). Thus, we have the ground-truth sample set \( \bar{U} = \{(x_1, \bar{y}_1), (x_2, \bar{y}_2), \ldots, (x_q, \bar{y}_q)\} \) with respect to \( U \). Note that although the labels in ground-truth sample set \( \bar{U} \) are virtual, we hope that the ground-truth labels can be well estimated by SSL embedded in our ALDR+ algorithm. Based on the ground-truth sample set \( \bar{U} \), we define \( \hat{S} = Q \cup \bar{U} \). Thus, we have an enhanced empirical risk \( \hat{R}_{\bar{U}}(f) \), which evolves more samples than \( \hat{R}_Q(f) \). Second and more importantly, based on \( \hat{S} \), the function class complexity term can be defined as \( C(\mathcal{F}, n, \delta) = ((\text{Rad}_q(\mathcal{F}))/2) + ((\ln(1/\delta))/n)^{1/2} \), which could be significantly smaller than \( C(\mathcal{F}, q, \delta) \).

Thus, based on the empirical risk \( \hat{R}_{\bar{U}}(f) \) and the function class complexity \( C(\mathcal{F}, n, \delta) \) defined on the new virtual set \( \hat{S} \), we have a new active learning risk bound as shown in Theorem 2. We provide the proof of Theorem 2 in the Appendix.

Theorem 2: Given a kernel function \( K(x_i, x_j) \), this maps a sample into an RKHS associated with a nonlinear feature
mapping function \( \phi(x) \). If we have a queried sample set \( Q \) with the size of \( q \), an unlabeled set \( U \) with the size of \( n-q \) and its ground-truth sample set \( \bar{U} \), and let \( S = Q \cup \bar{U} \), the following holds at least \( 1-\delta \):

\[
\mathcal{R}(f) \leq \hat{\mathcal{R}}_S(f) + \text{MMD}_\phi(S, Q) + C(F, n, \delta)
\]

(5)

where \( C(F, n, \delta) = ((\text{Rad}_n(F))/2) + ((\ln(1/\delta))/n)^{1/2} \) is the function class complexity term.

**Remark 2:** Compared to Theorem 1, they query the samples that are expected to reduce the empirical risk of labeled samples and ignore the empirical risk on unlabeled samples. The ground-truth label to the unlabeled sample can derive an enhanced active risk \( \hat{\mathcal{R}}_S(f) \). Thus, we use SSL to approximate the empirical risk \( \hat{\mathcal{R}}_S(f) \). We could obtain a tighter active learning risk bound than the one in (4) because we make full use of all unlabeled samples into the empirical risk and the function class complexity term. Especially, if \( \hat{\mathcal{R}}_S(f) \leq \hat{\mathcal{R}}_Q(f) \), according to Lemma 1, we definitely have a tighter active learning risk bound in (5) than the one in (4).

### III. New Efficient Batch Mode Active Learning

In this section, we first present the principle of our batch mode active learning algorithm (i.e., ALDR+) based on Theorem 2 and then give the details of ALDR+ algorithm. Then, we propose a warm start strategy to dynamically update the model of ALDR+. Finally, we give the computational complexity of our ALDR+ algorithm.

#### A. Principle of Our ALDR+ Algorithm

We have a labeled sample set \( Q = \{(x_1, y_1), (x_2, y_2), \ldots, (x_q, y_q)\} \) initially. In our batch mode active learning algorithm, we iteratively select the best subset \( \hat{Q} \subset U \) with \( k \) samples to label and put them into the labeled set \( Q \). Note that the ground-truth sample set \( \bar{U} \) is a virtual set, which is only used for theoretical usage. In practice, we use a new loss function \( L' \) (e.g., the symmetric hinge loss used in SSL) to approximate the empirical risk on the ground-truth sample set \( \bar{U} \).

Thus, we propose a practical batch mode active learning algorithm combined with SSL by minimizing the active learning risk bound in (5). Specifically, we present the batch mode active learning as an optimization problem with respect to the classifier \( f \) and the query set \( \hat{Q} \) as follows:

\[
\{ \hat{Q}^*, f^* \} = \arg \min_{\hat{Q} \subseteq U, |\hat{Q}| = k} \sum_{x \in Q \cup \hat{Q}} L(f(x), y) + \sum_{x \in U \setminus \hat{Q}} L'(f(x)) + n\text{MMD}_\phi(S, Q \cup \hat{Q}) + \lambda \| f \|^2_F
\]

(6)

where \( \| f \|^2_F \) is used to constrain the complexity of the classifier class, which is equivalent to constraining \( C(F, n, \delta) \) [28]. The model (6) queries the most effective samples by minimizing the active learning risk bound after each query. After a certain number of queries, model (6) can find a classifier \( f \in F \) with the property that its expected risk \( \mathcal{R}(f) \) is as small as possible. The model (6) can be divided into two parts: representativeness and informativeness. Specifically, we use MMD to describe representativeness and \( S^3 \text{VM} \) to describe informativeness.

1) **Representativeness:** We substitute the MMD term in (6) with its quadratic form for convenience of calculation. Thus, the MMD between the sets \( Q \cup \hat{Q} \) and \( S \) can be computed using the expression in [27], [29], and [30] as follows:

\[
\text{MMD}_S^2(Q \cup \hat{Q}, Q) = \left\| \frac{1}{n} \sum_{x_i \in S} \phi(x_i) - \frac{1}{q+k} \sum_{x_i \in Q \cup \hat{Q}} \phi(x_i) \right\|_F^2.
\]

Similar to [31], this formulation can be represented as

\[
\frac{1}{2} \alpha^T K_{UU} \alpha + \frac{u-k}{n} 1_Q K_{QU} - \frac{q+k}{n} 1_U K_{UU} + \text{constant}
\]

where \( 1_Q \) is a vector of length \( q \) with all entries 1, \( 1_U \) of length \( u \), \( \alpha \) is the indicator vector with \( u \) elements and each element \( a_i \in [0, 1] \), and \( \alpha^T 1_U = k \). \( K \) is the kernel matrix with its elements as \( K_{ij} = K(x_i, x_j) = \langle \phi(x_i), \phi(x_j) \rangle \), and \( K_{AB} \) denotes its submatrix between the samples from set \( A \) and \( B \). The objective can be further simplified as follows:

\[
\alpha^T K_{1} \alpha + k a
\]

where \( K_{1} = (1/2) K_{UU} - k_{3} - k_{2} \), and \( \forall x_i \in U, k_{3}(i) = (q+k)/n \sum_{x_j \in U} K(i, j) \) and \( k_{3}(i) = (u-k)/n \sum_{x_j \in \bar{U}} K(i, j) \).

2) **Informativeness:** In each query, to minimize the upper bound of true risk, we look for the most uncertain samples. The worst case situation is that the pseudo labels are the opposite of the true labels. In this case, the pseudo labels are given by \( y = -\text{sign}(f(x)) \). We use the hinge loss \( h_1(yf(x)) = \max(0, 1 - yf(x)) \) for the loss function \( L(f(x), y) \). Thus, the risk term related to \( \hat{Q} \) becomes

\[
\arg \min_{\hat{Q} \subseteq U, |\hat{Q}| = k} \sum_{x \in \hat{Q}} \max(0, 1 + |f(x)|) = \sum_{x \in \hat{Q}} 1 + |f(x)|. \quad (7)
\]

For any classifier, (7) identifies the samples closest to the decision boundary, which is given by

\[
\min_{\hat{Q} \subseteq U, |\hat{Q}| = k} \sum_{x \in \hat{Q}} |f(x)|.
\]

In this article, we use \( S^3 \text{VM} \) [32] to approximate the empirical risk \( \hat{\mathcal{R}}_S(f) \) on \( \hat{S} \). Based on the above discussions to the representativeness and informativeness, we have the following optimization problem:

\[
\min_{\alpha^T 1_U = k, w, b} \frac{1}{2}(w, w) + C \sum_{i=1}^{q} h_1(y_i f(x_i)) + C^* \sum_{i=q+1}^{n} h_1(|f(x_i)|) + \sum_{i=q+1}^{n} a_i |f(x_i)| + \beta (\alpha^T K_{1} \alpha + k a) \quad (8)
\]

where \( h_1(|\cdot|) \) is the symmetric hinge loss, and \( C \) and \( C^* \) are predefined parameters. Part 1 is actually the \( S^3 \text{VM} \) as a proxy of the summation of the empirical risk \( \hat{\mathcal{R}}_S(f) \) and the function class complexity, and part 2 is the criteria (i.e.,
Algorithm 1 Batch Mode Active Learning Algorithm for Querying Discriminative and Representative Samples via SSL (ALDR+)

Input: $Q = \{(x_i, y_i)\}_{i=1}^{q}$, $U = \{x_i\}_{i=q+1}^{n}$, parameters $C, C^*$, $\beta$, batch size $k$, tolerance $\epsilon$ for convergence condition.
Initialize: Set initial variables and parameters.

1: repeat
2: \textbf{Step 1:} Solve $S^3$VM by iteratively solving the objective function (12) w.r.t. $\theta$ based on the framework of CCCP.
3: \textbf{Step 2:} Solve the objective function (14) w.r.t $\alpha$ using a QP solver, set the largest $k$ elements in $\alpha$ to 1 and others to 0.
4: until Convergence condition is satisfied

Output: The query indicator vector $\alpha$.

informativeness (discriminativeness) and representativeness) for querying the unlabeled samples. $\beta$ in (8) balances the discriminative and representative information during the query process. In the remainder of this section, we will propose a practical batch mode active learning algorithm combined with SSL to solve the resulting optimization problem.

B. ALDR+ Algorithm

We provide a wrapper algorithm (also alternating direction method of multipliers) to solve the optimization problem (8), which mainly includes two steps.

Step 1: Solve $S^3$VM based on the framework of the concave–convex procedure (CCCP) with a fixed $\alpha$.
Step 2: Solve $\alpha$ based on a quadratic problem (QP) solver with fixed $w$ and $b$. We summarize our ALDR+ algorithm in Algorithm 1.

1) \textbf{Step 1 (Solve $S^3$VM):} The decision function of $S^3$VM is $f(x) = \langle w, \phi(x) \rangle + b$, and it aims to learn a maximum margin over labeled and unlabeled samples as follows:

$$
\min_{w,b} \frac{1}{2} \langle w, w \rangle + C \sum_{i=1}^{l} h_1(y_i f(x_i)) + C^* \sum_{i=q+1}^{n} h_1(|f(x_i)|)
$$

(9)

We double the unlabeled data set $U$ and create an artificial labeled data set $\widehat{U} = \{(x_{q+1}, 1), \ldots, (x_{q+u}, 1), (x_{q+u+1}, -1), \ldots, (x_{q+2ua}, -1)\}$. Thus, the original formulation (9) can be transformed to a difference between two convex functions, i.e., $o(w, b)$ and $v(w, b)$, as follows:

$$\min_{w,b} \frac{1}{2} \langle w, w \rangle + C \sum_{i=1}^{l} h_1(y_i f(x_i)) + C^* \sum_{i=q+1}^{q+2u} h_1(y_i f(x_i))
$$

$$- C^* \sum_{i=q+1}^{q+2u} h_0(y_i f(x_i)) + o(w, b)
$$

(10)

It must be pointed out that the problem (10) is definitely equivalent to the original problem (9) as proved in [32]. During the process of solving (10), we use $\mu_i$ which is defined in (11) to simplify the calculation procedure of the CCCP.

$$
\mu_i = \frac{1}{\partial f(x_i)} \nabla f(x_i) = \begin{cases} C^*, & \text{if } y_i f(x_i) < 0, \ i \geq q + 1 \\ 0, & \text{otherwise.} \end{cases}
$$

(11)

Then, we can obtain the primal convex inner loop (denoted as CIL) problem for (10) based on the CCCP which is skipped here. We directly show the corresponding dual CIL problem as follows [32]:

$$
\min_{\theta} \frac{1}{2} \theta^T H \theta - y^T \theta
$$

s.t. $\sum_{i=1}^{q+2u} \theta_i = 0; \ \ C_\ell \leq \theta_i \leq C_\ell, \ i = 1, \ldots, q + 2 \times u$

(12)

where $H$ is a positive semidefinite matrix with $H_{ij} = K(x_i, x_j) = \langle \phi(x_i), \phi(x_j) \rangle$ for all $1 \leq i, j \leq q + 2 \times u$, and $K(x_i, x_i)$ is the kernel function. $C_\ell$ and $C_\ell$ are defined as follows:

$$C_\ell = \begin{cases} -\mu_i, & \text{if } y_i = +1 \\ \mu_i - C, & \text{if } y_i = -1, i = 1, \ldots, q \\ \mu_i - C^*, & \text{if } y_i = -1, i \geq q + 1 \end{cases}
$$

$$C_\ell = \begin{cases} C - \mu_i, & \text{if } y_i = +1, i = 1, \ldots, q \\ C^* - \mu_i, & \text{if } y_i = +1, i \geq q + 1 \\ \mu_i, & \text{if } y_i = -1, \end{cases}
$$

The problem (12) is a standard constrained QP similar to the formulation of SVM, which can be solved by the sequential minimal optimization (SMO) algorithm [33]–[35]. The CCCP algorithm iteratively solves a sequence of the convex problem (12) until $\mu_i$ converges, and then, we can find a local minimal turning point.

2) \textbf{Step 2 (Solve the $\alpha$):} Once the CCCP algorithm converges as mentioned earlier, the decision function $f(x)$ can be obtained based on the final $\theta$. Thus, the objective function of Part 2 in (8) becomes

$$\min_{a^T 1_U = k} \sum_{i=1}^{u} a_i |f(x_i)| + \beta(a^T K_1 a + k a).
$$

(13)

The binary constraint on $a_i$ makes the integer QP defined in (13) NP-hard. A common strategy is to relax the constraints to make it a continuous optimization problem, resulting in the following formulation:

$$\min_{0 \leq a \leq 1, a^T 1_U = k} \sum_{i=1}^{u} a_i |f(x_i)| + \beta(a^T K_1 a + k a).
$$

(14)

The minimization problem (14) is a standard constrained QP and similar to the formulation of SVM [33]. Thus, the problem (14) can be solved efficiently by many existing solvers (e.g., the quadprog function in MATLAB and LIBSVM [33] with a little modification).
Algorithm 2 Batch Incremental Learning for $S^3$VM

Input: $\theta$, $b'$, $C$, $C$, $M$, $E$, $O$, $Q$ and $U$.
Output: $\theta$, $b'$, $C$, $C$, $M$, $E$, $O$, and $U$.

1: Remove sample $(x_i, y_i)$ from $M$, $E$, $O$ and $U$.
2: Add $(x_i, y_i)$ into $A$.
3: While $A \neq\emptyset$ do
4: Compute $\beta_b$, $\beta_M$ and $\gamma$.
5: Compute the maximal quantity $\eta_{\max}$.
6: Update $\theta_A$, $\theta$, $b'$, $g$, $C$, $C$, $M$, $E$, $O$ and $Q$.
7: end while

Algorithm 3 Batch Incremental Learning for $S^3$VM

Input: $\theta$, $b'$, $C$, $C$, $M$, $E$, $O$ and $Q$.
Output: $\theta$, $b'$, $C$, $C$, $M$, $E$, $O$ and $Q$.

1: Initialize $\theta_{\text{old}} = \emptyset$.
2: Add $(x_i, y_i)$ into $A$.
3: While $A \neq\emptyset$ do
4: Compute $\beta_b$, $\beta_M$ and $\gamma$.
5: Compute the maximal quantity $\eta_{\max}$.
6: Update $\theta_A$, $\theta$, $b'$, $g$, $C$, $C$, $M$, $E$, $O$ and $Q$.
7: end while

C. Update the Model by Warm Start Strategy

To avoid retraining the model of $S^3$VM from scratch after each query, we design two unique procedures based on the path-following technique [36]–[38], which can remove the chosen samples from the unlabeled set $U$ and add the queried samples into the labeled set $Q$ efficiently. We summarized the two procedures in Algorithms 2 and 3, respectively, and illustrated them in Fig. 1.

Differences Compared With [36]: The differences are summarized as follows.

1) Gu et al. [37] proposed an incremental learning algorithm to $S^3$VM, which can only handle one added sample at a time. However, our batch decremental and incremental learning algorithms can handle multiple removed or added samples at a time.
2) Our batch decremental and incremental learning algorithms can update the model directly, avoiding retraining the semisupervised classifier from scratch after each query.

In the following, we first give the Karush–Kuhn–Tucker (KKT) conditions [36] to the dual CIL problem of $S^3$VM and then provide the details to the two algorithms.

1) KKT Conditions for Dual CIL Problem of $S^3$VM: According to the convex optimization theory [39], by introducing Lagrangian multiplier $b'$ corresponding to the constraint in (12), the dual CIL problem can be formulated as follows:

$$
\min_{\bar{Q} \leq \bar{b} \leq \bar{C}} \max_{\bar{b} \in \bar{R}} W = \frac{1}{2} \theta^T H \theta - y^T \theta + b'(\sum_{i=1}^{q+2u} \theta_i). \quad (15)
$$

The first-order partial derivative of $W$ leads to the following KKT conditions:

$$
\frac{\partial W}{\partial b'} = \sum_{i=1}^{q+2u} \theta_i = 0 \quad (16)
$$

$$
g_i \triangleq \frac{\partial W}{\partial \theta_i} = \sum_{j=1}^{q+2u} \theta_j H_{ij} + b' - y_i \begin{cases}
> 0, & \theta_i = \bar{C}_i \\
= 0, & \text{then } \bar{C}_i \leq \theta_i \leq \bar{C}_i \\
< 0, & \text{then } \theta_i = \bar{C}_i.
\end{cases} \quad (17)
$$

Thus, according to the value of $g_i$ in (17), the extended training sample set $Q \cup \bar{U}$ can be partitioned as $\pi = (M, E, O, \bar{C})$, where $M = \{i | \in Q \cup U : g_i = 0, \bar{C}_{\leq} \leq \theta_i \leq \bar{C}_i\}$, $E = \{i | \in Q \cup \bar{U} : g_i < 0, \theta_i = \bar{C}_i\}$, and $O = \{i | \in Q \cup \bar{U} : g_i > 0, \theta_i = \bar{C}_i\}$.

2) Batch Incremental Learning for $S^3$VM: Because we double the unlabeled data set $U$ and create an artificial labeled data set $\bar{U}$ in the problem (10), we need to remove $\bar{Q} = \{x_{\bar{Q}}, +1\}, (x_{\bar{Q}}, -1)\}$ from $\bar{U}$. If removing $\bar{Q}$ from $\bar{U}$, it may lead to the changes to the weights $\theta$ and parameters $b'$. To handle this situation, we define a KKT-violating set $A$ in Definition 1.

Definition 1: The KKT-violating set $A$ is defined as a subset of $\bar{U} \cup Q$ such that all the samples violating the KKT conditions are included in $A$.

Initially, we add the chosen data set $Q$ into the KKT-violating set $A$ and gradually decrease the Lagrangian multiplier $\theta_{\bar{Q}}$ until $\theta_{\bar{Q}} = 0$. When $\theta_i = 0$ for $x_i \in \bar{Q}$,
we can remove \( x_i \) form \( \tilde{U} \). Specifically, to achieve this goal, the following two issues need to be addressed.

1) **Compute the Direction of \( \Delta \theta \)**: Set \( \Delta \theta_A \) as the changes of the weights of set \( A \), and set the direction to \( \Delta \theta_A \) as \( d_A = \tilde{C}_A - \alpha A_A \), where
\[
\tilde{C}_i = \begin{cases} 
C_i, & \text{if } y_i = +1 \\
\tilde{C}_i, & \text{if } y_i = -1.
\end{cases}
\]
Thus, we have \( \Delta \theta_A = \eta \cdot d_A \), where \( \eta \) is a parameter with \( 0 \leq \eta \leq 1 \) to control the adjustment qualities of \( \theta_A \), and the direction of \( \Delta \theta_A \) with respect to \( \eta \) can be obtained by solving the following linear system:
\[
\begin{bmatrix} 0 & 1_M \beta_M \\ 1_M H_{M,A} \end{bmatrix} \begin{bmatrix} \beta_M \\ \beta_M \end{bmatrix} = - \begin{bmatrix} 1_M \\ \theta_{MA} \end{bmatrix} d_A \quad (18)
\]
where \( \beta_M \) refer to the changes of the \( b' \) and \( M \), respectively. Furthermore, with the conclusion as stated earlier, the linear relationship between \( \Delta g_i(\forall i \in Q \cup \tilde{U} \cup A) \) and \( \eta \) can be obtained as follows:
\[
g_i \equiv \sum_{j \in A} \gamma_{ij} j_d + \sum_{j \in \tilde{M}} \gamma_{ij} j_d + \beta_M. \quad (19)
\]

2) **Compute the Maximum Adjustment Quantity \( \eta_{\text{max}} \)**: The maximum adjustment quantity \( \eta_{\text{max}} \) of \( \eta \) can be obtained by solving the following system of linear inequalities (20)–(28). The system of linear inequalities gives an interval of \( \eta \). The right endpoint of this interval is the value of \( \eta_{\text{max}} \)
\[
\begin{align*}
C_i & \leq \theta_i + \beta_i \eta \leq \bar{C}_i \quad \forall i \in \tilde{M} \quad (20) \\
g_i + \gamma_i \eta > 0 & \quad \forall i \in O \quad (21) \\
g_i + \gamma_i \eta < 0 & \quad \forall i \in \tilde{E} \\
y_i g_i + 1 + y_i \gamma_i \eta \geq 0 & \quad \forall i \in \tilde{M} & (23) \\
y_i g_i + 1 + y_i \gamma_i \eta < 0 & \quad \forall i \in \tilde{M} & (24) \\
\theta_i + \beta_i \eta \leq \bar{C}_i & \quad \forall i \in A & (25) \\
\theta_i + \beta_i \eta \geq C_i & \quad \forall i \in A & (26) \\
g_i + \gamma_i \eta > 0 & \quad \forall i \in \tilde{M} & (27) \\
g_i + \gamma_i \eta \leq 0 & \quad \forall i \in \tilde{E} & (28)
\end{align*}
\]

3) **Batch Incremental Learning for \( S^2VM \)**: If the queried data set \( \tilde{Q} \) is added into the labeled data set \( Q \), the change of \( \mu_i \) could lead to the result that the corresponding samples violate the KKT-conditions. Similar to the batch decremental learning process, during the batch incremental process, the migrations of samples among sets could also lead to the changes in the weights \( \beta \) and parameters \( b' \). The update of the parameters and the migration of samples among the sets \( \tilde{M}, \tilde{E}, O, A \) in the batch incremental learning process is the same as the batch decremental learning process, as discussed in Section III-C2. Note that the direction of the changes of the parameters \( \Delta \theta_A \) in batch incremental learning is contrary to the batch decremental learning process. Thus, if \( i \in A, \tilde{C}_i \) can be obtained as follows:
\[
\tilde{C}_i = \begin{cases} 
C_i, & \text{if } y_i = +1 \\
\tilde{C}_i, & \text{if } y_i = -1.
\end{cases}
\]

**D. Computational Complexity**

In this section, we analyze the computational complexity of our ALDR+ algorithm.

First, we give the computational complexity of batch incremental and decremental learning algorithms. The computational complexity of batch decremental and incremental learning algorithms mainly consist of three parts: 1) compute the \( \beta \) and \( \gamma \); 2) compute the \( \eta_{\text{max}} \); and 3) the computational complexity of incremental and decremental learning algorithm. To compute \( \beta \), we mainly solve the system of equations (18) of size \( |M| + 1 \) using the QR decomposition, which involves \( O(|M|^2) \) computations. According to (11), the computation of \( \gamma \) requires \( O(n|M|) \) computations. However, in each iteration of Algorithms 2 and 3, the change in set \( M \) is slight (either adding a sample to \( M \) or removing a sample from \( M \)), which leads that the change in \( H \) is just either adding or deleting a row and a column from \( H \). Thus, the QR decomposition can be updated without recomputing the QR decomposition from scratch and the computational complexity of updating the \( \beta \) is \( O(|M|^2) \). Furthermore, the computational complexity for solving (20)–(28) to compute \( \eta_{\text{max}} \) can be reduced to \( O(|M|^2) \). The updates of \( \theta_i, \beta_i, C_i, \tilde{C}_i, M, \tilde{E}, O, A \) require \( O(n) \) computations. Thus, similar to (37), the computational complexity of batch decremental and incremental learning algorithms is \( O(n^2|M| + n|M|^2) \). Then, to solve the objective function (14) with respect to \( a \) for Step 2 in Algorithm 1, the QP solver requires \( O(n^2) \) computations. Finally, the computational complexity of our ALDR+ algorithm is \( O(n^2|M| + n|M|^2 + n^2) \).

**IV. EXPERIMENTS**

In this section, we first present the experimental setup and then provide the experimental results and discussions.

**A. Experimental Setup**

1) **Design of Experiments**: In the experiments, to verify the accurateness and effectiveness of our ALDR+, we compare our method with random selection and several state-of-the-art active learning approaches. Specifically, we count the accuracy and running time of each query. By counting the accuracy, we want to verify that the discriminative and representative samples can be chosen accurately by our ALDR+. By counting the running time, we want to show that ALDR+ can reduce huge computational cost and are efficient compared with the existing active learning approaches. We list the state-of-the-art active learning approaches compared in the experiments as follows.

1) **RANDOM**: Randomly select the query samples.
2) **ERE**: Choose the sample that maximizes the reduction in the total predicted label entropy [10].
3) **ERM**: Use graph-based metrics with empirical risk minimization [18].
4) **ZHU**: Combine active learning and SSL using Gaussian fields and harmonic functions to minimize the estimated expected classification error [15].
This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS

Fig. 2. Illustrative example for querying discriminative and representative samples by our approach. (a) Binary classification. (b) Initial labeled samples. (c) After querying four samples. (d) After querying eight samples.

Fig. 3. Comparison of different active learning methods on benchmark data sets.

5) **HSE**: Hierarchical subquery evaluation to release the potential of expected error reduction [14].

6) **QUIRE**: Measure both the informativeness and representativeness of an instance by its prediction uncertainty [17].

7) **DRBM**: Query discriminative and representative samples with empirical risk minimization [19].

8) **S3VM BMAL**: S3VM batch mode active learning via a min–max framework [16].

9) **ALDR+**: Our proposed batch mode active learning algorithm by fully exploiting unlabeled data.

2) **Implementation**: We implemented RONDOM, DRBM, SVM BMAL, and our ALDR+ algorithms in MATLAB. We used the MATLAB code from http://visual.cs.ucl.ac.uk/pubs/graphActiveLearning/ as the implementations of ERE, ERM, and HSE. We used the MATLAB code from http://lamda.nju.edu.cn/code_QUIRE.ashx as the implementation of QUIRE. We used the MATLAB code from

**TABLE II**

**BENCHMARK DATA SETS USED IN THE EXPERIMENTS. “P” AND “N” DENOTE THE POSITIVE AND NEGATIVE SAMPLES, RESPECTIVELY**

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Dimensionality</th>
<th>Samples</th>
<th>Source</th>
<th>P vs. N</th>
</tr>
</thead>
<tbody>
<tr>
<td>codrna</td>
<td>8</td>
<td>59,535</td>
<td>LIBSVM</td>
<td>0.7735:1</td>
</tr>
<tr>
<td>ijcnn1</td>
<td>22</td>
<td>49,990</td>
<td>LIBSVM</td>
<td>0.1072:1</td>
</tr>
<tr>
<td>usps</td>
<td>256</td>
<td>7,291</td>
<td>LIBSVM</td>
<td>0.7074:1</td>
</tr>
<tr>
<td>mushrooms</td>
<td>112</td>
<td>8,124</td>
<td>LIBSVM</td>
<td>1.7351:1</td>
</tr>
<tr>
<td>a9a</td>
<td>123</td>
<td>32,561</td>
<td>LIBSVM</td>
<td>0.3195:1</td>
</tr>
<tr>
<td>svmguide1</td>
<td>4</td>
<td>7,089</td>
<td>LIBSVM</td>
<td>1.2949:1</td>
</tr>
<tr>
<td>isolet</td>
<td>617</td>
<td>7,797</td>
<td>UCI</td>
<td>1.2236:1</td>
</tr>
<tr>
<td>phishing</td>
<td>68</td>
<td>11,055</td>
<td>LIBSVM</td>
<td>1.2585:1</td>
</tr>
<tr>
<td>letter</td>
<td>16</td>
<td>20,000</td>
<td>LIBSVM</td>
<td>1.0255:1</td>
</tr>
<tr>
<td>w3a</td>
<td>300</td>
<td>4,912</td>
<td>LIBSVM</td>
<td>0.0018:1</td>
</tr>
</tbody>
</table>
Fig. 4. Comparison of different active learning methods on real-world data sets.

TABLE III
WIN/TIE/LOSS COUNTS (%) FOR OUR METHOD VERSUS EACH COMPETING METHOD DURING THE WHOLE ACTIVE LEARNING PROCESS, BASED ON PAIRED T-TEST AT THE 95% CONFIDENCE LEVEL, WHERE THE MOST ADVANTAGEOUS RESULTS OF OUR ALGORITHM ARE HIGHLIGHTED IN BOLD FONT AND THE CLOSEST RESULTS OF OUR ALGORITHM ARE HIGHLIGHTED IN UNDERLINE

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Vs ERE</th>
<th>Vs ZHU</th>
<th>Vs HSE</th>
<th>Vs QUIER</th>
<th>Vs DRBM</th>
<th>Vs SVM BMAL</th>
</tr>
</thead>
<tbody>
<tr>
<td>codrna</td>
<td>81/15/4</td>
<td>92/4/4</td>
<td>92/4/4</td>
<td>96/4/0</td>
<td>0/100/0</td>
<td>0/100/0</td>
</tr>
<tr>
<td>iccv1</td>
<td>12/88/0</td>
<td>14/86/0</td>
<td>0/100/0</td>
<td>0/100/0</td>
<td>0/100/0</td>
<td>0/100/0</td>
</tr>
<tr>
<td>usps</td>
<td>32/68/0</td>
<td>75/35/0</td>
<td>24/76/0</td>
<td>40/60/0</td>
<td>92/8/0</td>
<td>43/37/0</td>
</tr>
<tr>
<td>mushrooms</td>
<td>33/67/0</td>
<td>37/63/0</td>
<td>33/67/0</td>
<td>42/58/0</td>
<td>36/64/0</td>
<td>22/88/0</td>
</tr>
<tr>
<td>a9a</td>
<td>6/94/0</td>
<td>38/62/0</td>
<td>3/96/3</td>
<td>100/0/0</td>
<td>0/100/0</td>
<td>39/61/0</td>
</tr>
<tr>
<td>svnguide1</td>
<td>0/96/4</td>
<td>0/96/4</td>
<td>4/96/0</td>
<td>0/100/0</td>
<td>14/86/0</td>
<td>49/6/0</td>
</tr>
<tr>
<td>islet</td>
<td>20/77/0</td>
<td>17/83/0</td>
<td>14/86/0</td>
<td>14/86/0</td>
<td>11/89/0</td>
<td>0/100/0</td>
</tr>
<tr>
<td>phishing</td>
<td>58/42/0</td>
<td>56/44/0</td>
<td>47/53/0</td>
<td>53/47/0</td>
<td>24/76/0</td>
<td>36/64/0</td>
</tr>
<tr>
<td>letter</td>
<td>44/55/1</td>
<td>84/16/0</td>
<td>32/68/0</td>
<td>28/72/0</td>
<td>22/78/0</td>
<td>22/78/0</td>
</tr>
<tr>
<td>w3a</td>
<td>39/61/0</td>
<td>18/82/0</td>
<td>8/92/0</td>
<td>8/92/0</td>
<td>11/89/0</td>
<td>0/96/4</td>
</tr>
</tbody>
</table>

http://www.cs.cmu.edu/~zhuxj/pub/semisupervisedcode/active_learning/ as the implementation of ZHU. For each data set, we randomly selected 60% of the data as the training set and the remaining 40% as the test set. We used the training set for active learning and compared the prediction accuracy on the test set. We randomly sampled ten labeled samples from the 60% training data to train an initial classifier. In each iteration, five unlabeled samples were first selected to solicit their class labels and the classification model was then retrained. We queried the oracle 50 times per data set. We used LIBSVM provided in [33] as the implementation of SVM to report the accuracy curve after each query. We use fivefold cross validation to optimize the hyperparameters of the Gaussian kernel and performed on the 60% training data, and consequently, 12% of the original data set fell into each fold. We use a grid search to search for relevant parameters in cross validation. In ALDR+, the parameters $C$ and $C^*$ were fixed at 10 and 5, respectively. We ran our algorithm with tradeoff parameter $\beta$ values from a candidate set {1, 10, 100, 1000} by fivefold cross validation to choose the appropriate parameter.

3) Data Sets: We do the experiments on the benchmark and real-world data sets as follows.

a) Benchmark data sets: Ten benchmark data sets summarized in Table II were used in the experiments that are from
LIBSVM \textsuperscript{2} and UCI\textsuperscript{3} Originally, usps, isolet, and letter data sets are the data sets for multiclass classification. We created a binary version of usps by classifying digits 0–4 versus 5–9. We created binary versions for the isolet and letter data sets by classifying letters A–M versus N–Z. We added the ratio of positive-to-negative samples (P versus N) as the last column in Table II.

\textbf{b) Real-world data sets:} The two real-world data sets (ImageNet\textsuperscript{4} and news20\textsuperscript{5}) were used in the experiments. ImageNet is an image database organized according to the WordNet hierarchy. We randomly selected two classes and built a subset with 2600 images. We extracted their 4096-D CNN features via AlexNet and performed PCA to reduce the dimension to 512. The news20 data set is a collection of 19,928 messages, collected from 20 different newsgroups. We randomly selected two subject matters for active learning.

\section*{B. Results and Discussion}

Fig. 2 shows an illustrative example for querying discriminative and representative samples by our approach. Our ALDR\textsuperscript{+} can select discriminative and representative samples alternately by optimizing an enhanced active learning risk bound. The results show that our approach can get a good result only after querying four samples. With the increase in the number of samples queried, the decision boundaries will become more and more accurate.

Fig. 3 shows the average results of ten independent runs of different active learning methods on ten benchmark data sets.

\textsuperscript{2}https://www.csie.ntu.edu.tw/cjlin/libsvmtools/datasets/
\textsuperscript{3}http://archive.ics.uci.edu/ml/datasets.html
\textsuperscript{4}http://www.image-net.org/
\textsuperscript{5}http://qwone.com/~jason/20Newsgroups/

Fig. 4 shows the learning accuracies over queries compared with different active learning methods on two real-world data sets. Table III summarizes the win/tie/loss counts for our method versus each competing method during the whole active learning process, based on the paired t-test at the 95\% confidence level. The results clearly demonstrate that our ALDR\textsuperscript{+} is always among the best ones on all data sets and has the ability in dealing with a practical problem. This is because we fully exploit unlabeled data to make our approach to perform better in active learning.

Fig. 5 shows that the tunable parameter $\beta$ has a different effect on different data sets. We can observe that smaller $\beta$ works better on the svmguide1 and codrna data sets, whereas larger $\beta$ works better on the mushrooms, usps, and isolet data sets. In other words, informativeness can better mine the data information for the svmguide1 and conra data sets, whereas the representativeness is more important to the mushrooms, usps, and isolet data sets. The reason may be that svmguide1 is distributed more densely. Thus, representativeness can help boost active learning. We conclude from these experiments that we need to pay more attention to the data distribution when the distribution is messy.

Table IV shows the average time (in seconds) per query of each competitive active learning algorithm. Table V shows the final accuracies of each competitive active learning algorithm. The results show that our method can achieve the best accuracy at a relatively low time cost. Our method has the highest learning efficiency by comparing the total average query time of ten benchmark data sets. This is due to the fact that we design two unique procedures based on the path-following technique to avoid retraining the model from scratch. Specifically, we can update the classifier of ALDR\textsuperscript{+} efficiently after each query by...
removing the chosen samples from $U$ and adding the queried samples into $Q$.

V. CONCLUSION

Querying informative and representative samples is the state-of-the-art approach for active learning. However, most of the existing active learning algorithms via querying the informative and representative samples did not utilize SSL to improve the quality of queried samples. In this article, we propose a novel batch mode active learning combined with semisupervised SVM based on an enhanced active learning risk bound. To the best of our knowledge, our new ALDR+ algorithm is the first batch mode active learning algorithm to fully exploit unlabeled data in characterizing the informativeness (discriminativeness) and representativeness. Especially, to avoid retraining the semisupervised classifier from scratch after each query, we design two unique procedures based on the path-following technique, which can remove multiple queried samples from the unlabeled data set and add the queried samples into the labeled data set efficiently. The superior performance of our method is verified by our extensive evaluation using benchmark data sets and real-world data sets compared with the state-of-the-art active learning methods.

APPENDIX

Proof of Theorem 2

Proof 1: Following [28], we know that the relationship between the expected risk and empirical risk is:

$$\mathcal{R}(f) \leq \hat{\mathcal{R}}_{\hat{S}}(f) + \frac{\text{Rad}_{\mathcal{S}}(F)}{2} + \sqrt{\frac{\ln(1/\delta)}{n}}.$$ 

To extend the empirical risk minimization principle to active learning, we use $\text{MMD}_{\phi}(S, Q) = \| (1/n) \sum_{i \in S} \phi(x_i) - (1/q) \sum_{j \in Q} \phi(x_j) \|_F$ to empirically estimate MMD between $S$ and $Q$ in the RKHS. Thus, we have the new inequality as follows:

$$\mathcal{R}(f) \leq \hat{\mathcal{R}}_{\hat{S}}(f) + \text{MMD}_{\phi}(S, Q) + C(F, n, \delta)$$

where $C(F, n, \delta) = ((\text{Rad}_{\mathcal{S}}(F)) / 2) + ((\ln(1/\delta)) / n)^{1/2}$ is the function class complexity term.

REFERENCES
