Active Learning without Knowing Individual Instance Labels: A Pairwise Label Homogeneity Query Approach

Yifan Fu, Bin Li, Xingquan Zhu, *Senior Member, IEEE*, and Chengqi Zhang, *Senior Member, IEEE*

Abstract—Traditional active learning methods require the labeler to provide a class label for each queried instance. The labelers are normally highly skilled domain experts to ensure the correctness of the provided labels, which in turn results in expensive labeling cost. To reduce labeling cost, an alternative solution is to allow nonexpert labelers to carry out the labeling task without explicitly telling the class label of each queried instance. In this paper, we propose a new active learning paradigm, in which a nonexpert labeler is only asked "whether a pair of instances belong to the same class", namely, a pairwise label homogeneity. Under such circumstances, our active learning goal is twofold: (1) decide which pair of instances should be selected for query, and (2) how to make use of the pairwise homogeneity information to improve the active learner. To achieve the goal, we propose a "Pairwise Query on Max-flow Paths" strategy to query pairwise label homogeneity from a nonexpert labeler, whose query results are further used to dynamically update a Min-cut model (to differentiate instances in different classes). In addition, a "Confidence-based Data Selection" measure is used to evaluate data utility based on the Min-cut model's prediction results. The selected instances, with inferred class labels, are included into the labeled set to form a closed-loop active learning process. Experimental results and comparisons with state-of-the-art methods demonstrate that our new active learning paradigm can result in good performance with nonexpert labelers.

Index Terms—Active learning, weak labeling, pairwise label homogeneity

1 Introduction

In many real-world applications, manually labeling massive data collections is expensive and impractical. Active learning [6], [29] aims to address this issue by selecting a subset of most critical instances for labeling. An active learner aims to achieve a high classification accuracy using as few labeled instances as possible, thereby minimizing the cost for acquiring labeled instances [30]. In most traditional active learning methods, an expert labeler (also called an "oracle") is required to provide ground truths to the queried instances and the model is updated by incorporating the new labeled data. The updated model is applied to the unlabeled data again and another subset of unlabeled data are selected for the expert's labeling. This procedure is iterated multiple times until some criterion is met.

Although the classical active learning paradigm only requires a subset of instances to be labeled, it is not an easy task for that the selected subset can still be of large size and the active learning can last for many iterations. In addition, because the model is learned only based on a subset of the entire data set, the labeling quality of the selected instances is extremely crucial for the model's performance. As a result, the labeling task in traditional active learning methods is still expensive in many cases.

Recently, researchers resort to employing committees of weak (nonexpert) labelers, which are cheaper but can only provide noisy labels for unlabeled instances. Some works based on this idea have been proposed, such as [26], [32], for solving the standard supervised learning problem with multiple weak labelers. However, such noisy labels may not be helpful in active learning scenarios for at least two reasons: (1) Because only a small subset of critical instances is selected for labeling, the labeling quality in active learning is more sensitive to the model's performance than that in standard supervised learning. (2) Because active learning comprises multiple learning iterations, the errors induced in each round will be passed onto the following rounds and will be amplified. Therefore, asking nonexpert labelers to directly provide noisy labels may be risky for active learning.

In this paper, we propose a new active learning paradigm, named pairwise homogeneity based active learning (PHAL), in which a nonexpert labeler is only asked "whether a pair of instances belong to the same class". Unlike labeling individual instances, pairwise label homogeneity has less requirement on labelers' domain knowledge. This intuition can be illustrated using an animal classification example in Fig. 1, in which pictures (a)-(c) are camels and (d)-(e) are sheep. Suppose (a) and (f) are labeled as "camel" and "sheep", respectively, and our goal is to

- *Y. Fu is with the School of Computing and Mathematics, Charles Sturt University, Bathurst, NSW 2795, Australia. E-mail: yfu@csu.edu.au.*
- *B. Li and C. Zhang are with the Center for Quantum Computation and Intelligent Systems (QCIS), Faculty of Engineering and Information Technology, University of Technology, Sydney (UTS), 235 Jones Street, Sydney, NSW 2007, Australia. E-mail: {Bin.Li-1, Chengqi.Zhang}@uts.edu.au.*
- *X. Zhu is with the Department of Computer & Electrical Engineering and Computer Science, Florida Atlantic University, 777 Glades Road, Boca Raton, FL 33431. E-mail: xzhu3@fau.edu.*

Manuscript received 15 May 2012; revised 21 Mar. 2013; accepted 25 Sept. 2013; date of publication 17 Oct. 2013; date of current version 18 Mar. 2014. Recommended for acceptance by V.S.-M. Tseng.
For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below.
Digital Object Identifier no. 10.1109/TKDE.2013.165

1041-4347 © 2013 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.
See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

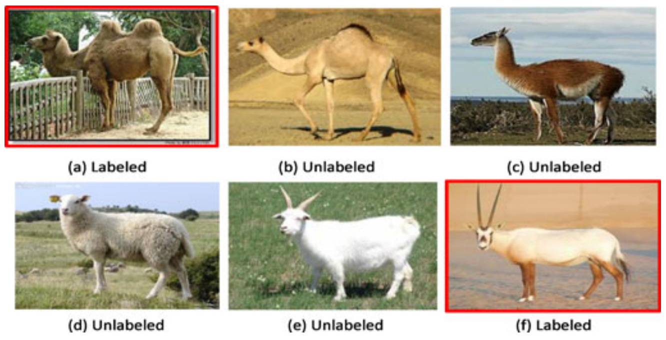


Fig. 1. A motivating example of active learning using pairwise label homogeneity. The underlying learning task is to differentiate two types of animals "camel" (first row) versus "sheep" (second row). Instead of requiring labelers to provide ground truth label for (c), which is difficult to obtain because (c) is visually similar to both (a) and (f), we propose to label homogeneities of pair (a,b) and pair (b,c), which are visually similar and can be easily labeled by a nonexpert, to help build accurate classifiers.

actively label some of the remaining pictures to train a classifier. In traditional active learning, we need a zoologist to label these pictures since a nonexpert may find it difficult to tell the ground truth label of (c), which seems unlike to either (a) or (f). In PHAL, this puzzle can be addressed by querying the label homogeneities of pair (a, b) and pair (b, c), which are visually similar and can be easily labeled by a nonexpert. It does not matter even though the nonexpert provides a wrong label for the pair (a, c)—As long as most label homogeneities in local neighborhoods are correctly labeled, the underlying learner will finally find paths from any unlabeled instance to the labeled ones based on the pairwise homogeneity information. Thus, PHAL can not only reduce labeling cost but also tolerate more noise.

Based on this assumption, the underlying queries in PHAL are to generate pairwise constraints between labels, which will be incorporated into the active learning procedure. Fig. 2 illustrates the difference between the traditional active learning paradigm and the proposed one. In contrast to a specific label assigned to each queried instance in the traditional paradigm, the new paradigm only acquires a pairwise label homogeneity information ("yes/no") for each query, which is much easier and cheaper for the labeler.

While the aforementioned nonexpert labeler based active leaning paradigm provides an opportunity to reduce labeling cost and makes a labeling task easier to fulfil, the "yes/no" pairwise label homogeneity information cannot be directly utilized to benefit active learning due to a lack of specific class labels for individual instances. Therefore, to enable an effective active learning process based on the pairwise label homogeneity information, we need to address two technical challenges: (1) decide which pair of instances should be selected for query, and (2) how to make use of the pairwise homogeneity information to improve the active learner. For pair selection, we propose to query pairwise label homogeneity of unlabeled pairs on the Max-flow paths and update the corresponding Min-cut models with the query results. Using the improved Min-cut models, we select a subset of instances with high confidences on their prediction results, and include these instances, along with their inferred class labels, into the labeled set to improve the active learning process.

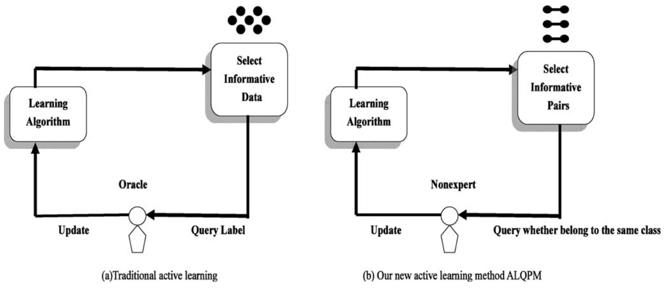


Fig. 2. Traditional active learning paradigm *versus* the proposed PHAL paradigm. Traditional active learning explicitly queries the class label of each instance, whereas our approach queries the class homogeneity between a pair of instances, i.e., whether a pair of instances belong to the same class or not. Because the labeler is not required to provide class label of each queried instance, pairwise label homogeneity queries are much easier/cheaper to answer in reality.

Compared to the existing active learning work, the contribution of PHAL is threefold:

- *Pairwise label homogeneity versus specific labels*: Querying class labels of individual instances is expensive in the traditional active learning paradigm, even if the size of queried subset is not large. To reduce labeling cost, PHAL only queries pairwise label homogeneities of unlabeled instance pairs, which is much easier and can be answered by nonexpert labelers.
- *max-flow based pair selection versus random pair selection*: To decide which pair of instances should be selected for query, we construct an instance graph and regard that instance pairs on the Max-flow paths are more important to help discriminate instances of different classes. We theoretically verify that the Max-flow paths based weight adjustment strategy can reduce the leave-one-out (LOO) error of the corresponding Min-cut based classifier, which confirms that, compared to random pair selection, our instance pair selection strategy is more effective for refining the decision boundary.
- *A utility measure for instance selection*: Based on the prediction results of the Min-cut based classifier ensemble, the proposed utility measure uses a confidence based data selection criterion to choose data with high prediction confidence to extend the labeled data set.

The remainder of the paper is organized as follows. We introduce related work in Section 2. The problem formulation and the proposed method for active learning by querying pairwise label homogeneity are presented in Sections 3 and 4 respectively. Experimental results are reported in Section 5 and we conclude the paper in Section 6.

2 Related Work

Traditional active learning [29] intends to reduce labeling cost by selecting the most informative instances to label, where informativeness is typically defined as the maximum expected improvement in classification accuracy. After selecting an optimal subset of most informative instances (an instance subset can also contain only one instance), it will query the labels of these instances from the labeler. Based on the types of query strategies, active learning can be divided into the following categories (interested readers can refer to our recent survey [15] for a detailed summarization of existing active learning methods).

Active learning with membership queries represents a group of methods which directly query the class labels of individual instances. Briefly speaking, the active learner acquires instance labels from the labeler to extend the labeled training set and refines the model iteratively. This strategy explicitly queries the class membership of each instance. Existing methods in this category can be roughly divided into three subcategories: *pool-based active learning*, *stream-based active learning*, and *query construction based active learning*. Pool-based active learning [17], [19] assumes all (labeled and unlabeled) instances can be observed as a candidate pool. It first measures sample utilities in the pool to decide which ones can maximally improve the performance of the current model; then the learner queries their class memberships from a domain expert. Stream-based active learning [42], [43], on the other hand, assumes that unlabeled instances are constantly flowing in a stream fashion. An active learning method is required to label the most informative instances to help train an accurate prediction model from the stream data. Query construction based active learning [3], [20] can generate some synthetic instances (without an unlabeled pool) and then query labels for these pseudo-instances to extend the labeled training set.

Active learning with generalized queries represents a set of algorithms which ask some simplified (or generalized) questions to help improve the classification accuracy. Generalized query is different from membership query mainly because the former asks the class labels of a subset of instances instead of a specific instance. In contrast to asking the labeler specific queries, the domain experts are ready to answer "simplified" or "generalized" queries formed by an optimal feature subset. For example, instead of directly asking whether a specific patient (i.e., an instance) has one type of disease (i.e., class label) or not, a learner can ask generalized questions like "for patients with high blood pressure whether they are likely to have the disease or not?". The answers to the generalized questions can help refine the decision boundary for classification. In [10], the authors proposed a feature based query method in active learning, which utilizes a simple feature to label instances. A generalized query is introduced in [11], [28] to label instances in groups. [37] adds confidence scores to instances based on affinities between features and labels. In [14], we employed a nonexpert labeler based active learning to query whether a pair of instances belong to the same class.

A number of studies [9], [22], [25], [27], [33] have shown that active learning greatly helps reduce labeling effort in various domains. However, traditional active learning depends on some strong assumptions about labelers. For example, active learning assumes there exists a unique omniscient labeler. In reality, it is more likely to have multiple labelers with different areas of expertise. Active learning also assumes that the unique labeler is perfect (say "oracle") and always provides correct answers to the queried instances. In reality, though, the labeler may be incorrect and provide noisy answers sometimes. Furthermore, the labeler is not always indefatigable, that is to say, it may refuse to answer if it is uncertain or too busy. Active learning presumes the labeler is either free or inexpensive and charges uniform cost in labeling tasks. To relieve these strong assumptions, a large number of methods have been proposed to handle applications with nonexpert labelers scenarios. From the labeler's perspective, existing solutions mainly follow two directions:

- *Active learning with one labeler*: Many methods [21] in active learning focus on selecting instances for labeling and assume that the labeling task is handled by a single, noise-free labeler. This kind of methods only consider the labeling cost on a large size of data set, which is addressed by selecting a subset of data with maximum utilities. The classical utility measures used in active learning can be categorized into four major types: *Uncertainty sampling* is based on posterior probabilities, including margin sampling [1], entropy measures [30], and least confidence measures [2]. *Query by committee* (QBC) [31] is the second commonly used type, where a committee of classifiers are used to assess unlabeled instances based on voting disagreements or divergences. The third type uses *expected model change* for discriminative probabilistic models, such as sequence labeling using conditional random fields [30]. In the fourth type, the selection criterion is to find instances directly *reducing model bias and/or variance* [16], [27] such that the model trained from the labeled instances is expected to achieve the minimum error rate.
- *Active learning with crowdsouring labelers*: The methods in this category use a group of cheap and noisy labelers to address the labeling cost issue. Due to the noisy labels provided by nonexpert labelers, the tradeoff between labeling noise and labeling cost is a big challenge for active learning with crowdsourcing labelers [32], [38]. In order to improve labeling quality of weak labelers, some works [8], [23], [34], [35], [40] explicitly consider each labeler's annotation cost and confidence to select one or a subset of optimal labelers to a specific queried instance; others may use incrementally relabeling method [39] or transfer learning [13] to obtain an accurate labeler.

Different from all the above methods, the proposed active learning paradigm employs nonexpert labelers to perform labeling in each iteration of the active learning process, but only requests them to provide pairwise label homogeneity information (i.e., whether a pair of instances have the same labels or not). Compared to specifying instance labels, answering "whether two instances belong to the same class" is much simpler for nonexpert labelers and the queried results are more reliable for training prediction models.

3 Problem Formulation

3.1 Problem Setting

Given a data set $\mathcal{D}$, which comprises a labeled subset $\mathcal{D}^L$, an unlabeled subset $\mathcal{D}^U$, and a test set $\mathcal{D}^T$. The ith instance in $\mathcal{D}^L$ is denoted by $(\mathbf{x}_i, y_i)$, where $\mathbf{x}_i$ is the feature vector and

TABLE 1
Notations Used in the Paper

$\mathcal{D}^L$, $\mathcal{D}^U$ and $\mathcal{D}^T$	denote the labeled, unlabeled, and test data sets, respectively
$\mathcal{V}^L$ and $\mathcal{V}^U$	denote the labeled and unlabeled vertex sets, respectively
$\mathcal{V} = \mathcal{V}^L \cup \mathcal{V}^U$	denotes the vertex set of a graph
$\mathcal{E}_n$	denotes the edge set of the nth graph
$\mathcal{G}_n = (\mathcal{V}, \mathcal{E}_n)$	denotes the nth graph
$\mathbb{G} = \{\mathcal{G}_1, \ldots, \mathcal{G}_N\}$	denotes a set of graphs
N	denotes the number of graphs in $\mathbb{G}$
$v_i \in \mathcal{V}$	denotes the ith vertex in the graph
$(\mathbf{x}_i, y_i)$	denotes the pair of feature vector and class label of v_i
$C \in \{+, -\}$	denotes a class label
$\hbar_n, n = 1, 2, \ldots, N$	denotes the nth Min-cut classifier learned on $\mathcal{G}_n$
$\mathbb{H} = \{\hbar_1, \ldots, \hbar_N\}$	denotes a set of Min-cut classifiers
$\hbar_n(v_i) \in \{+, -\}$	denotes the predicted label for v_i using $\hbar_n$
Δ	denotes the budget (total number of queried pairs during active learning)
$\aleph$	denotes the data set selected for labeling in each iteration
$p_C(v_i)$	denotes the probability that v_i belongs to class C
$q(v_i)$	denotes the confidence of the predicted label for v_i

y_i is the class label. Meanwhile, the ith instance in $\mathcal{D}^U$ or $\mathcal{D}^T$ is denoted by $(\mathbf{x}_i, ?)$, with the question mark denoting an unknown label. In order to train a classifier on $\mathcal{D}^L$ with maximum prediction accuracy, a commonly employed strategy for active learning is to query the class labels of the most informative instances in $\mathcal{D}^U$ from an expert labeler (also called oracle) and expand $\mathcal{D}^L$ with the new labeled data.

Instead of directly querying the instance labels in a traditional active learning way, we consider a pairwise label homogeneity query setting in this paper. Assume we employ a nonexpert labeler, who can only answer whether a pair of instances $(\mathbf{x}_i, \mathbf{x}_j)$ belong to the same class or not. We aim to solve the following two technical challenges: (1) Given an active learner, how to select unlabeled pairs for querying label homogeneity. (2) After collecting the answers, how to make use of such information to train a better classifier.

For ease of presentation, the notations used in this paper are summarized in Table 1.

3.2 Method Overview

The incorporation of pairwise label homogeneity information immediately inspires a graph-based transductive learning approach. Our main idea is to make use of homogeneity information to iteratively correct the edge weights of the similarity graph in the graph-based transductive learner and finally boost its prediction accuracy. Specifically, we first select some most important pairs to query from the nonexpert labeler. Then, the label homogeneity information is used to update the current model. After that, we infer the class memberships of unlabeled data based on the updated model and evaluate data utility with a utility measure. Finally, a classifier including the selected most informative instances with inferred labels is trained to predict the test set. This active learning paradigm is illustrated in Fig. 3.

To instantiate the above active learning paradigm by incorporating pairwise label homogeneity information, we can choose a graph-based transduction model as a base learner. We employ Min-cut [4], [5] as the base leaner, which naturally rests on a pairwise similarity graph for vertex bipartition (binary classification) by minimizing the sum of the edge weights between two partitions (one for positive instances and the other for negative instances). Because Max-flow paths play an important role for graph bipartition, we select unlabeled pairs on the Max-flow paths as the most important pairs to query their label homogeneity. We use an ensemble of Min-cut classifiers to infer the class memberships of unlabeled vertices and treat the majority voting results outputted by the Min-cut classifier ensemble as the final prediction result of an unlabeled vertex. The maximum probability output value is considered as its confidence. It is assumed that the vertices with the highest confidence values provide most useful information to help build an accurate model.

Before proceeding, we give an overview of the proposed PHAL procedure with the following four major steps:

1. *Graph ensemble construction*. To build Min-cut based classifiers, we first construct an ensemble of k-NN graphs in terms of k in a range with a fixed step.
2. *Weight adjustment in min-cut sets*. After applying the Min-cut algorithm to the obtained graphs, a nonexpert labeler is asked to provide label homogeneity information to the queried instance pairs. Based on the query results, we adjust the weights of the queried pairs.
3. *Confidence based data selection*. The class memberships of unlabeled vertices are inferred according to the ensemble of Min-cut classifiers. By sorting the unlabeled vertices according to their label confidences, the top vertices are selected as the optimal subset $\aleph$.

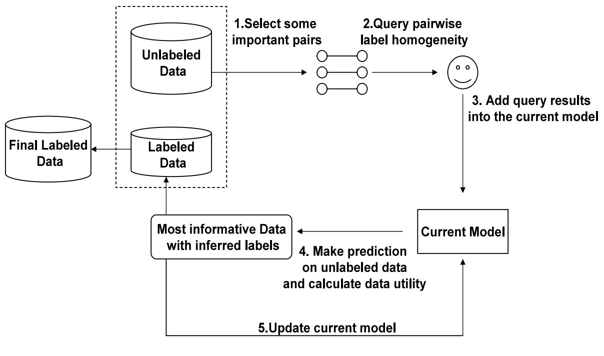

Unlabeled Data
1.Select some important pairs
2.Query pairwise label homogeneity
3. Add query results into the current model
Final Labeled Data
Labeled Data
Most informative Data with inferred labels
4. Make prediction on unlabeled data and calculate data utility
Current Model
5.Update current model

Fig. 3. The proposed pairwise homogeneity based active learning framework.

4. *Weight update in selected subset*. We use the final prediction result of an unlabeled vertex in $\aleph$ as its class label. Then we further check the edges which have the vertices in $\aleph$. If an edge links two labeled vertices, we update the edge weight according to its label homogeneity. By doing so, we obtain the class labels of the vertices in $\aleph$, and then use the vertices with inferred class labels to update the graph weights. As a result, the labeler's answers to the label homogeneity queries can be incorporated into the Min-cut classifiers in the next iteration to improve the active learning process.

The last three steps will iterate multiple times until reaching the budget Δ (i.e., the total number of queried pairs during active learning iterations). In each iteration, the active learner selects a small optimal subset $\aleph$ and treats the predicted labels as their genuine labels. This information is then used to update both the graph ensemble $\mathbb{G}$ and the classifier ensemble $\mathbb{H}$, and also helps select the optimal subset in the next iteration. The entire algorithm is described in Algorithm 1.

Algorithm 1 Active Learning by Querying Pairwise Label Homogeneity

```
Input: (1) $\mathcal{D}^L$: a labeled data set; (2) $\mathcal{D}^U$: an unlabeled data set; (3)
    $n$: the number of graphs to construct the graph ensemble; (4) $\Delta$:
    the budget of pairs queried during active learning.
Output: $\mathcal{V}^L$: a labeled vertex set.
 1: Construct a graph ensemble $\mathbb{G} = \{\mathcal{G}_1, \dots, \mathcal{G}_N\}$ with different $k$
    values based on $\mathcal{D}^L \cup \mathcal{D}^U$;
 2: while $the\ number\ of\ queried\ pairs < \Delta$ do
 3:     Learn the Min-cut classifier ensemble $\mathbb{H} = \{\hbar_1, \dots, \hbar_N\}$ on
        $\mathbb{G}$;
 4:     Query class homogeneities of unlabeled pairs $(v^i, v^j)$ on the
        Max-flow paths of each graph in $\mathbb{G}$;
 5:     Adjust the edge weights of the queried pairs according to their
        class homogeneities;
 6:     Predict the class memberships of $\mathcal{V}^U$ using $\mathbb{H}$;
 7:     Calculate the label prediction confidences for $\mathcal{V}^U$;
 8:     Sort $\mathcal{V}^U$ according to their prediction confidences and select
        the top ones as the optimal subset $\aleph$;
 9:     $\mathcal{V}^U \longleftarrow \mathcal{V}^U \setminus \aleph$;
10:     $\mathcal{V}^L \longleftarrow \mathcal{V}^L \cup \aleph$;
11:     Update the edge weight if an edge has at least one vertex in
        $\aleph$;
12: end while
```

4 The Proposed Method

In this section, we introduce the technical details of the proposed pairwise label homogeneity query based active learning method. Section 4.1 and Section 4.2 will address the first challenge mentioned in Section 3.1, and Sections 4.3 and 4.4 will address the second challenge, followed by the computational complexity analysis.

4.1 Graph Ensemble Construction

Given a distance metric, many methods exist to construct graphs. In the following, we introduce the design criteria for constructing an graph ensemble used in the Min-cut algorithm. First of all, it is expected that a graph at least has some small balanced cuts for the Min-cut based approach. While these cuts may be inconsistent with the labeled vertices, we do not anticipate that the Min-cut algorithm fails in the beginning. This suggests that the potential graph construction method only produces edges between very similar nodes. Second, the graph is expected to have the property that a small number of connected components cover nearly all the instances. This indicates that the graph can represent the real data distribution and provide sufficient correlation information between the instances in the data set.

Based on the above criteria, we adopt the k-nearest neighbor (k-NN) algorithm [7] to construct graphs, where an edge exists between two vertices (instances) if one vertex is a member of the other one's top k nearest neighbors, and vice versa. This setting caters the first criterion with the assumption that vertices near in the topology structure are similar to each other. Furthermore, it is helpful to select the best model parameter k to reach its optimal performance. However, it is difficult to obtain the optimal k for adapting different data sets, so as to reflect real data distributions as the second criterion says. To this end, we construct an ensemble of graphs with different k values ranging from 3 to 24 with a fixed step of 3. Because of the generalization capability of the ensemble model, it guarantees that our method can at least outperform the average performance of the individual models built separately with different k values.

Given the labeled data set $\mathcal{D}^L$ and the unlabeled data set $\mathcal{D}^U$, we collect all instances in $\mathcal{D}^L \cup \mathcal{D}^U$ to form the vertex set $\mathcal{V} = \mathcal{V}^L \cup \mathcal{V}^U$ in the graph ensemble. That is to say, $v_i \in \mathcal{V}$ is assigned a feature vector $\mathbf{x}_i$ and a class label y_i, if labeled, or "?" if unlabeled. For N different values of k, we construct N edge sets $\mathcal{E}_1, \dots, \mathcal{E}_N \subseteq \mathcal{V} \times \mathcal{V}$, respectively. As a result, we can obtain a graph ensemble $\mathbb{G} = \{\mathcal{G}_1 = (\mathcal{V}, \mathcal{E}_1), \dots, \mathcal{G}_N = (\mathcal{V}, \mathcal{E}_N)\}$. We use $\mathcal{V}^L_+$ to indicate the vertex set with positive labels and $\mathcal{V}^L_-$ the vertex set with negative labels. An edge weight $w(v_i, v_j)$ in a graph is set using following steps:

- *Add classification vertices*. Because we use the Min-cut algorithm, it is required to set a source vertex and a sink vertex. We add two binary classification vertices v_+ and v_- to the vertex set, which are treated as the source and the sink, respectively. As a result, the vertex set for constructing the graph ensemble becomes $\mathcal{V} = \mathcal{V} \cup \{v_+, v_-\}$. All the other vertices in $\mathcal{V}$, except $\{v_+, v_-\}$, are called data vertices.
- *Set edge weights with classification vertices*. The classification vertex v_+ and v_- are only connected to the labeled vertices in $\mathcal{V}^L_+$ and $\mathcal{V}^L_-$, respectively. The edge weight between the classification vertex and a labeled vertex is set to a large value ∞. Specifically, $w(v_+, v_i) = \infty$ for all $v_i \in \mathcal{V}^L_+$ and $w(v_-, v_i) = \infty$ for all $v_i \in \mathcal{V}^L_-$.
- *Set edge weights without classification vertices*. As analyzed before, we adopt the k-NN algorithm to generate the edge for each pair of data vertices. The edge weight between two data vertices represents the similarity between them. Specifically, the weighting function used in the paper is determined as follows:

$$w(v_i, v_j) = \exp(-\delta(v_i, v_j)), \tag{1}$$

where $\delta(v_i, v_j)$ denotes the distance between $\mathbf{x}_i$ and $\mathbf{x}_j$. We adopt Hamming distance for categorical features and euclidean distance for numerical features.

4.2 Weight Adjustment in Min-Cut Sets

After graph construction, we use the Min-cut algorithm to bipartition the graphs for binary classification [4]. The Min-cut based classifiers are based on the Max-flow Min-cut theorem [24], which states that, given a flow network, the maximum flow passing from the source to the sink equals the minimum cut of edge capacities (weights) in the network.

Theorem 1 (Max-flow Min-cut Theorem [24]). *Let f be a flow passing from the source to the sink in a network $\mathcal{G}$ and $(\mathcal{A}, \mathcal{B})$ be a cut, where $\mathcal{G} = \mathcal{A} \cup \mathcal{B}$. Then, for any cut, we have $f(\mathcal{G}) \leq w(\mathcal{A}, \mathcal{B})$, where $w(\mathcal{A}, \mathcal{B})$ is the capacity of the cut. When $f(\mathcal{G}) = w(\mathcal{A}, \mathcal{B})$, f is a maximum flow and $(\mathcal{A}, \mathcal{B})$ is a minimum cut of the network.*

Proof. Suppose the source is in $\mathcal{A}$ and the sink is in $\mathcal{B}$, then $f(\mathcal{G})$ from the source to the sink is equal to the flow passing from $\mathcal{A}$ to $\mathcal{B}$. Let $f(v_i, v_j)$ denote the flow on a directed edge $v_i \rightarrow v_j$, we have

$$f(\mathcal{G}) = \sum_{v_i \in \mathcal{A}, v_j \in \mathcal{B}} f(v_i, v_j) - \sum_{v_i \in \mathcal{B}, v_j \in \mathcal{A}} f(v_i, v_j) \tag{2}$$

$$\leq \sum_{v_i \in \mathcal{A}, v_j \in \mathcal{B}} f(v_i, v_j) \tag{3}$$

$$\leq \sum_{v_i \in \mathcal{A}, v_j \in \mathcal{B}} w(v_i, v_j) = w(\mathcal{A}, \mathcal{B}). \tag{4}$$

The first inequality holds obviously and the second holds because the flow on an edge cannot exceed its capability. When there is no backflow from $\mathcal{B}$ to $\mathcal{A}$ and no additional capacity can be explored in the cut, we can obtain $f(\mathcal{G}) = w(\mathcal{A}, \mathcal{B})$. □

The maximum flow through a series of paths relies on the smallest flow of the edge on each path, which is also the bottleneck of each path. This implies, if these bottleneck edges are removed from the network, it results that no flow can pass from the source to the sink. Thus, Max-flow and Min-cut is an equivalent problem and we can determine a minimum cut using the maximum flow algorithm.

When the labeled data are insufficient and the unlabeled data are abundant, as in the active learning problem setting, Min-cut based classification may have many minimum cuts (with equivalent maximum flows). This may lead to extremely imbalanced cuts, which is harmful for binary classification. Since Min-cut based classification [4], [5] belongs to a family of semi-supervised learning methods based on the manifold assumption [44], which assumes that the instances are more likely to belong to the same class if they are close in the feature space. This commonly used assumption motivates our new active learning paradigm to obtain additional label homogeneity information by querying unlabeled pairs on Max-flow paths in the Edmonds-Karp algorithm [12].

To acquire pairwise label homogeneity information, we query "whether v_i and v_j belong to the same class" to the nonexpert labeler. Based on the query result, we adjust the weight of edge between v_i and v_j as follows:

$$w(v_i, v_j) = \begin{cases} w(v_i, v_j) \times (1 + \partial) & \text{if } y_i = y_j, \\ w(v_i, v_j) \times (1 - \partial) & \text{if } y_i \neq y_j, \end{cases} \tag{5}$$

where ∂ is an adjustment factor, which determines the weight updating scale ($0 < \partial < 1$). According to Eq. (5), the edge weight increases by ∂, if the pair has the same class label, and decreases by ∂, otherwise.

We theoretically verify that the above weight adjustment can reduce the leave-one-out error of the underlying Min-cut based classifier. [4, Lemma 3.4], shows the prediction result $y_i = \text{sign}(\sum_{j \in kNN(i)} y_j w(v_i, v_j))$, where $kNN(i)$ denotes the k nearest neighboring vertices. Based on this result, a margin-like quantity is defined in [18]

$$\xi_i = y_i \frac{\sum_{j \in kNN(i)} y_j w(v_i, v_j)}{\sum_{j \in kNN(i)} w(v_i, v_j)}, \tag{6}$$

which can be viewed as the margin between an instance and the decision boundary. We use this margin-like quantity to upper bound the leave-one-out error

$$\epsilon_{LOO}(\mathcal{D}) \leq \sum_{i=1}^{|\mathcal{D}|} (1 - \xi_i), \tag{7}$$

which suggests that the error rate can be reduced by making the upper bound tighter. We can have the following result.

Theorem 2. *The upper bound Eq. (7) will become tighter if the edge weights are adjusted according to the pairwise label homogeneity query result using Eq. (5).*

Proof. To prove that the upper bound Eq. (7) can become tighter, we can equivalently prove that Eq. (8) will increase after adjusting the edge weights using Eq. (5):

$$\sum_{i=1}^{|\mathcal{D}|} \xi_i = \sum_{i=1}^{|\mathcal{D}|} \sum_{j \in kNN(i)} y_i y_j \frac{w(v_i, v_j)}{\sum_{j \in kNN(i)} w(v_i, v_j)}. \tag{8}$$

If the queried pair have the same label (i.e., $y_i y_j = 1$), $w(v_i, v_j)$ will become larger and the margin Eq. (8) will increase accordingly; if the queried pair have different labels (i.e., $y_i y_j = -1$), $w(v_i, v_j)$ will become smaller and the margin Eq. (8) will also increase accordingly. Therefore, the feedback of pairwise label homogeneity information will monotonously reduce the upper bound of the leave-one-out error over active learning iterations (we do not change $kNN(i)$ for v_i after edge weight adjustment). □

The above weight adjustment operation can also be interpreted from the view of the Max-flow Min-cut theorem: It will agglomerate the data in same classes (increasing the weights of same labeled instances) while separate the data in different classes (reducing the weights of differently labeled instances). As a result, the total weight of the edges across the two sections will be smaller and the decision boundary determined by these edges will be refined. Moreover, based on the Max-flow Min-cut theorem, the cut is formed by the edges with full capacity flows, which are on the Max-flow paths. This also suggests that querying pairs on the Max-flow paths is more effective than querying pairs randomly for reducing the upper bound Eq. (7). Over active learning iterations, the decision boundary of the classifiers will become clearer and the imbalanced cut issue will be relieved.

4.3 Confidence Based Instance Selection

By applying the Min-cut algorithm to the graph ensemble $\mathbb{G}$, an ensemble of Min-cut based classifiers $\mathbb{H} = \{\hbar_1, \ldots, \hbar_N\}$ are naturally derived on $\mathbb{G}$. Thus, the class labels of all unlabeled vertices can be inferred by the cuts. Specifically, vertices in the source and sink partitions are labeled positive and negative, respectively.

After obtaining the predicted labels of all unlabeled vertices in $\mathbb{G}$, we employ the predictions of $\mathbb{H}$ on each vertex to calculate its class label distribution

$$p_+(v_i) = \frac{1}{N}\sum_{n=1}^{N} \mathbb{I}(\hbar_n(v_i) = +), \tag{9}$$

$$p_-(v_i) = 1 - p_+(v_i), \tag{10}$$

where $\mathbb{I}(\hbar_n(v_i) = +)$ is an indicator function that outputs 1, if $\hbar_n(v_i) = +$, and 0, otherwise.

We choose the label with a higher probability as the final prediction $h(v_i)$ (e.g., $h(v_i) = 1$ if $p_+(v_i) > p_-(v_i)$), and the value is considered as the prediction confidence for the unlabeled vertex. The prediction confidence for vertex v_i is

$$q(v_i) = \max\{p_+(v_i), p_-(v_i)\}. \tag{11}$$

We sort the unlabeled vertices based on their confidence values in a descending order. The top $|\aleph|$ vertices are selected to form the optimal labeling subset, with the final prediction $h(v_i)$ as their labels. We add $\aleph$ into the labeled training set by $\mathcal{V}^L = \mathcal{V}^L \cup \aleph$ to update the active learner.

4.4 Weight Updating in Selected Subset

After incorporating the labeling information of the optimal subset $\aleph$ to the active learner, the graphs need to be updated based on this additional information. The new labels only affect the edges which have vertices in $\aleph$, where both vertex labels of those edges become available. Using their label homogeneity information, we update the corresponding edge weights using the similar operation as Eq. (5) introduced in Section 4.2

$$w(v_i, v_j) = \begin{cases} w(v_i, v_j) \times (1 + \varphi) & \text{if } y_i = y_j, \\ w(v_i, v_j) \times (1 - \varphi) & \text{if } y_i \neq y_j, \end{cases} \tag{12}$$

where φ is an adjustment factor, which determines the weight updating scale ($0 < \varphi < 1$). Because we use the predicted labels here, pairwise label homogeneity information may be incorrect. Therefore, in practice, we select a smaller value for φ than that for ∂ in Eq. (5), where the query answers are almost accurate.

4.5 Time Complexity Analysis

Assume the graph ensemble is updated T times (the maximum number of active learning iterations), the time complexity of our method can be decomposed into two parts: $B(\mathcal{V})$ and $U(\mathcal{V})$, where $B(\mathcal{V})$ denotes the time complexity for model training and $U(\mathcal{V})$ for active pair selection for label homogeneity queries.

The term $B(\mathcal{V})$ is further composed by the complexity of a graph ensemble construction $BG(\mathcal{V})$ and the complexity of the Min-cut based classifier ensemble training $BM(\mathcal{V})$. As aforementioned, we use the k-NN algorithm to construct N graphs with different number of neighbors. This procedure has a complexity of $O(|\mathcal{V}|^2 + N|\mathcal{V}|)$, where $O(|\mathcal{V}|^2)$ is for computing the pairwise similarity matrix and $O(N|\mathcal{V}|)$ for finding N different sets of neighbors. After generating the graph ensemble, we employ the Min-cut algorithm to train N Min-cut based classifiers for T times. We adopt the Edmonds-Karp algorithm [12], which has a complexity of $O(|\mathcal{V}||\mathcal{E}|^2)$, for solving the Min-cut problem. Therefore, we retrain N Min-cut based classifiers for T times, which totally has a complexity of $T\sum_{n=1}^{N} O(|\mathcal{V}||\mathcal{E}_n|^2)$. The total time complexity of $B(\mathcal{V})$ is

$$B(\mathcal{V}) = BG(\mathcal{V}) + BM(\mathcal{V}) \tag{13}$$

$$= O(|\mathcal{V}|^2) + O(N|\mathcal{V}|) + T\sum_{n=1}^{N} O(|\mathcal{V}||\mathcal{E}_n|^2). \tag{14}$$

In practice, since $|\mathcal{E}_n|$ is larger than $|\mathcal{V}|$ while N and T are usually small. Let $|\mathcal{E}|$ be the average of $|\mathcal{E}_n|$, we can further simplify Eq. (13) to be

$$B(\mathcal{V}) = O(TN|\mathcal{V}||\mathcal{E}|^2). \tag{15}$$

The term $U(\mathcal{V})$ is further composed by the complexity of pair queries $UP(\mathcal{V})$ and the complexity of optimal labeled data selection $UC(\mathcal{V})$. We assume that the average number of queried pairs in each iteration is M. Then the total complexity of T iterations is $UP(\mathcal{V}) = O(TM)$. For $UC(\mathcal{V})$, the membership distribution estimation needs $O(N|\mathcal{V}^U|)$ and the unlabeled data sorting based on confidences needs $O(|\mathcal{V}^U|^2)$, both of which are iterated for T times. As a result, the time complexity of $U(\mathcal{V})$ is

$$U(\mathcal{V}) = UP(\mathcal{V}) + UC(\mathcal{V}) \tag{16}$$

$$= O(TM) + O(TN|\mathcal{V}^U|) + O(T|\mathcal{V}^U|^2) \tag{17}$$

$$= O(TM) + O(T|\mathcal{V}^U|^2). \tag{18}$$

With the above analysis, the overall time complexity of our method is given as

$$B(\mathcal{V}) + U(\mathcal{V}) = O(TN|\mathcal{V}||\mathcal{E}|^2). \tag{19}$$

Eq. (19) shows that major computational cost of our method rests on the Min-cut based classifier ensemble training.

5 Experiments

We conduct the following three sets of experiments to validate the robustness and effectiveness of the proposed method. The experiments include three major parts, gradually advanced from the validation of the parameter settings, checking label errors and noise introduced by the

TABLE 2
Description of the Benchmark Data Sets

ID	Dataset	Instances	Features	Classes
1	Breastc	286	10	2
2	Liver	345	12	2
3	Wdbc	569	31	2
4	Monks1	432	7	2
5	Pima	769	9	2
6	Horse	368	23	2
7	Lucas	2000	11	2
8	German	1000	21	2
9	Vote	435	17	2
10	Kr-vs-kp	3196	37	2

proposed method, to the comparisons of the proposed method with baseline approaches. More specifically, 1) in Section 5.3, we investigate the parameter settings for k-NN graphs and initial training sets; 2) in Sections 5.4, we study the sensitivity of the proposed method by investigating noise in both initial labeled sets and expanded labeled sets; and 3) in Section 5.5, we compare the classification performance of our method to a number of baseline methods, including the traditional instance-labeled method and various pair-labeled strategies.

5.1 Data Description and Experiment Settings

We conduct experiments on ten benchmark data sets listed in Table 2. All data sets, except "lucas", are real-world binary classification data sets, which can be downloaded from UCI Machine Learning Repository.[1] "lucas" is a synthetic data set[2] simulating a medical application of lung cancer diagnosis, prevention, and treatment. It is generated using causal Bayesian networks with binary variables, with the target variable denoting whether a patient has lung cancer or not.

For fair comparisons, all experimental results are reported based on 10 times 10-fold cross validation. All methods are compared on the same training and test sets (the initial randomly labeled samples are also the same for all methods). We use the number of queried instance pairs as the cost factor and all methods are compared based on the same labeling budget, i.e., querying the same number of instance pairs.

All the compared methods are implemented using Java and WEKA [36] data mining toolbox. Once the labeling process is done, we use J48 (which is a WEKA implementation of the C4.5 decision tree algorithm) to train a classifier from the final labeled data set of each method. The performance of different methods is then compared based on the accuracies of the trained J48 classifiers validated on the same test set. Given that all the compared methods use the same training/test sets and the same labeling cost (number of queried instance pairs), we can conclude that our method has a better active learning performance than its peers if it outperforms the baseline methods in terms of the classification accuracy.

5.2 Baseline Methods

To the best of our knowledge, there is no existing method considering the "pairwise label homogeneity query" active learning paradigm. To comparatively study the performance of the proposed method (denoted as PHAL in the experiments), we design the following baseline methods using different pairwise label homogeneity query strategies. It is worth noting that, after obtaining the pairwise label homogeneity query results using different strategies, the rest steps of these baseline methods are as same as those in PHAL.

- *Querying pairwise label homogeneity active learning (QHAL)* [14] is the original version of the proposed method. The difference between QHAL and PHAL lies in the graph construction step, where QHAL only constructs a single k-NN graph with a fixed k value while PHAL constructs an ensemble of k-NN graphs with a set of different k values. QHAL queries unlabeled pairs on the Max-flow paths of the constructed graph and updates the model with the acquired information.
- *Random edge weight update active learning (REAL)* is a variant of PHAL within the same framework. The difference between REAL and PHAL is that REAL randomly selects edges in the graph rather than selecting edges on the Max-flow paths. For each randomly selected edge, REAL queries the label homogeneity between the two vertices linked by the selected edge.
- *Uncertain sample based active learning (USAL)* uses entropy [29] as an uncertainty measure. Each unlabeled instance $\mathbf{x}_i$'s entropy is calculated using the class distributions predicted by a classifier, defined as $H(\mathbf{x}_i) = -\sum_{y_i \in \{+,-\}} P(y_i|\mathbf{x}_i) \log P(y_i|\mathbf{x}_i)$, where $P(y_i|\mathbf{x}_i)$ is the probability of $\mathbf{x}_i$ belonging to class y_i. First, all unlabeled instances are ranked according to their uncertainties. Then we select top ranked instances to form a set of pairs for label homogeneity query.
- *Uncertain pair based active learning (UPAL)* is another variant of PHAL within the same framework. After generating an ensemble of Min-cut based classifiers, it first calculates the uncertainties (entropies) of unlabeled vertices according to the prediction results of the classifier ensemble. The uncertainty of an edge is the summation of the uncertainties of the two vertices of the edge. We rank edges in each graph according to their uncertainties and select top ranked pairs for label homogeneity query.
- *Pairwise homogeneity active learning (α) (PHAL(α))* varies the percentage of pairs queried on the Max-flow paths using a parameter α ($0 \leq \alpha \leq 1$). This method is a combination of PHAL and UPAL. PHAL(1.0) is exactly PHAL and PHAL(0) is exactly UPAL. For example, PHAL(0.5) means that a half pairs are queried on the Max-flow paths and the other half are queried based on pair uncertainty values.[3] The purpose of using PHAL(α) as a baseline is to study whether querying instance pairs on the Max-flow is indeed a good choice. Given an active learning task, if we observe an increasing performance gain from PHAL(α) as the value of α

1. http://archive.ics.uci.edu/ml/.
2. http://www.causality.inf.ethz.ch/challenge.php?page=datasets.

3. In our experiments, UPAL always outperforms REAL, so we use UPAL to select the remaining pairs for the combined method.

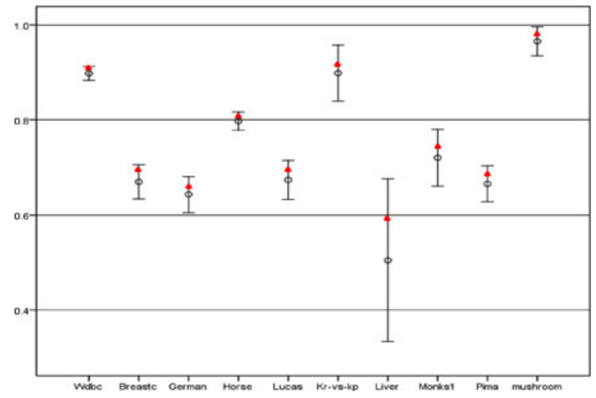


Fig. 4. The accuracy comparison between QHAL and PHAL. Each vertical line segment denotes the accuracy range of QHAL with k varying from 3 to 24 with a step of 3, and the circle on the line denotes the average accuracy in the range; a red triangle denotes the accuracy of PHAL, which comprises an ensemble of k-NN graphs with k varying from 3 to 15 with a step of 3.

increases, it will validate that querying instance pairs on the Max-flow paths is, at least, a better choice than the random query.

In addition to the above baseline methods based on pairwise label homogeneity query strategies for performance comparison, we also consider a baseline method based on individual instance query for comparatively study of the sensitivity to label errors (i.e., noise).

- *Instance label based active learning (ILAL)* uses a labeler to provide ground truth class label for each queried instance. It uses entropy as the instance utility measure for instance selection.

For fair comparisons, all baseline methods except ILAL are designed to work in a "batch mode" by selecting the *same* number of pairs at a time. In each iteration, PHAL queries pairwise label homogeneity information of the unqueried pairs on the Max-flow paths. Because the Max-flow paths often change as the active learning process iterates, the number of these pairs is not fixed in all iterations. In the experiments, we guarantee that all baseline methods query the same number of pairs as PHAL does in each iteration. For USAL, given n instances, they can form $\frac{n(n-1)}{2}$ pairs at most. Accordingly, assume we want to label ϱ pairs, we need to find γ to satisfy

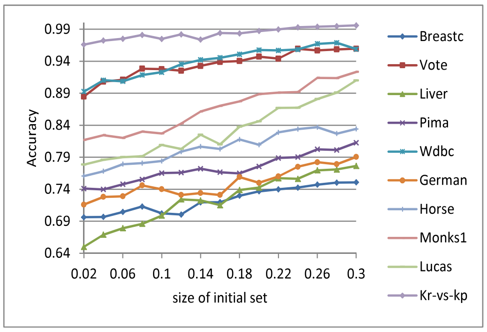
$$\varrho = \frac{\gamma(\gamma-1)}{2}. \tag{20}$$

After finding γ, we just select top γ uncertain instances to form a set of pairs for label homogeneity query.

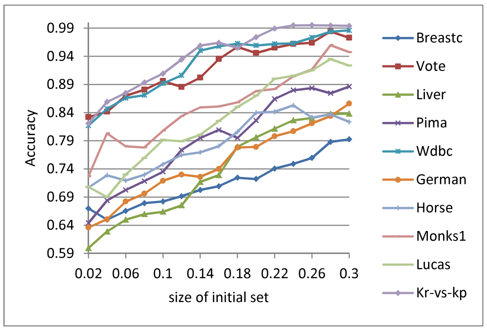
5.3 Parameter Settings

5.3.1 Number of the k-Nearest Neighbors

QHAL uses a predefined k value to construct a single k-NN graph and updates this graph by adjusting its edge weights on the Max-flow paths based on the pairwise label homogeneity query results. This approach cannot guarantee that the selected k will result in the best performance, unless we exhaustively search the optimal k value, which is computationally expensive. Moreover, as explained in Section 4.1, it is difficult to find a general criterion to search the optimal k which often varies and depends on the data sets. To address this problem, the proposed PHAL method adopts an ensemble of k-NN graphs with different k values to improve the generalization capability.



(a) PHAL



(b) ILAL

Fig. 5. Evaluation of accuracy with respect to the size of initial training set. Accuracy curves of (a) PHAL and (b) ILAL on 10 data sets with different sizes of initial training set ranging from 2 to 30 percent with a step of 2 percent.

In Fig. 4, we report the results of QHAL and PHAL with respect to different k values on 10 benchmark data sets. The results show that the performance of PHAL is always better than the average performance of QHAL with different k values in a large range (ranging from 3 to 24 with a step of 3). Although PHAL constructs the graph ensemble within a small range (ranging from 3 to 15 with a step of 3), its performance is superior to the average performance of QHAL. We can thus conclude that an ensemble of graphs with different k values can indeed help improve performance due to the generalization capability of the ensemble model.

5.3.2 Size of the Initial Labeled Set

The learning circle of active learning starts from a randomly labeled set. To study the impact of the size of the initial training set, we report the accuracies of PHAL and ILAL on 10 data sets in Fig. 5, respectively, with different sizes of the initial training sets ranging from 2 to 30 percent with a step of 2 percent. We set the same budget (size of the final labeled data set) for the purpose of a fair comparison. The x-axis indicates different sizes of the initial training sets while the y-axis reports the accuracies of the classifiers trained from the final labeled instances. We can easily find the trend that the accuracy curves of both methods increase as the size of the initial training set grows, which suggests that the more information an initial model learns, the more informative unlabeled data the underlying learner can find for labeling, so as to build a final accurate model. Another fact is that the accuracy curves of ILAL increase more steeply than those of PHAL, which indicates that PHAL is less sensitive to the size of the initial labeled set. This is because the performance of PHAL depends on the overall structures of affinity graphs such that the number of labeled nodes on a graph may not dramatically influence the performance. In comparison, the performance of ILAL largely depends on the initial given label information.

The results in Fig. 5 also show that PHAL uses fewer labeled samples than ILAL does to achieve the same accuracy on most data sets. Take "vote" as an example, PHAL and ILAL need label 2 and 10 percent of instances to reach the accuracy of 89 percent, respectively. This result demonstrates that active learning methods considering data topological structures (like PHAL) are able to find good decision boundaries more quickly. Active learning methods simply considering individual instance uncertainties (like ILAL) may result in redundancy and outlier problems for sample selection [15]. As a result, the latter requires more instances to be labelled in order to achieve the same level of accuracy.

5.4 Sensitivity to Noisy Labels

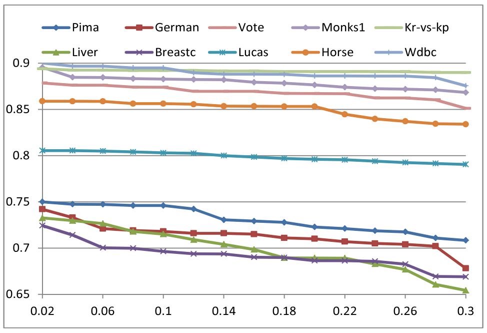
5.4.1 Input Noise

While the proposed pairwise label homogeneity query approach does not require expert labelers to provide ground truth label for each instance, in practice, unskilled labelers may provide noisy pairwise label homogeneity information in some uncertain cases. In this experiment, we comparatively study the sensitivities of PHAL (pairwise homogeneity query based) and ILAL (instance label based) to label errors (i.e., noise).

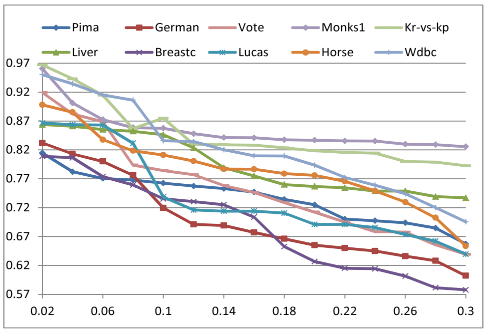
To measure the robustness of a model against noise, we can investigate the decreasing rate of the model's accuracy curve with respect to the percentages of noisy labels provided by a labeler (i.e., input noise). The more slowly the accuracy curve drops, the more robust the model is; otherwise, the model is sensitive to noise. Accordingly, we compare the decreasing rates of the accuracy curves of PHAL and ILAL to validate that PHAL is more robust than ILAL. Note that we do not compare the absolute accuracies of the two methods because the information acquired from pairwise homogeneity labeling and instance labeling are incomparable.

To simulate noise, we randomly generate labels for a certain percentage of queries as noisy labels. In particular, we randomly generate binary labels for the queried instance pairs in PHAL, and also randomly generate class labels for the queried individual instances in ILAL. Because PHAL queries the pairs on Max-flow paths on the graphs, the number of queried pairs may exceed the number of instances in the data set. As a result, we cannot guarantee that ILAL queries the same number of instances as PHAL does at each time. For fair comparisons, we guarantee that the same number of labeled instances are included into the labeled data set in each active learning iteration for both methods.

In Fig. 6, we report the accuracy curves of PHAL and ILAL with respect to different percentages of label noise, ranging from 2 to 30 percent with a step of 2 percent, on the 10 benchmark data sets. For PHAL, its accuracy curves on "kr-vs-kp", "monks1", "vote", and "lucas" have slight drops within the whole range, while the curves on the other six data sets decline slowly as the noise increases. For ILAL, its accuracy curves on all 10 data sets show much quicker decrement than the corresponding curves of PHAL, which asserts that the same amount of label errors have much more severe impact on ILAL than PHAL. This phenomenon might be caused by the following reason: In ILAL, the queried instance results are directly used for supervised model training. So noisy labels will directly impact on the model, and this type of errors have been confirmed to be most harmful for supervised learning [41]. PHAL, on the other hand, only uses pairwise label homogeneity information to update the affinity graphs. Because the classification results of PHAL depend on the overall structures of affinity graphs, changes of some edges may not result in significant errors. Therefore, we can conclude that, compared to instance label query based methods, the proposed pairwise label homogeneity query based method is less sensitive to noise.



(a) PHAL

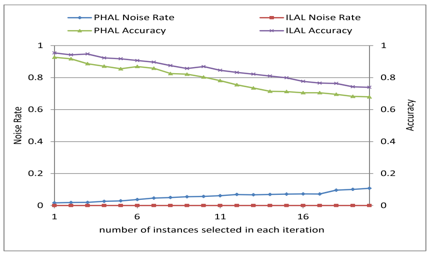


(b) ILAL

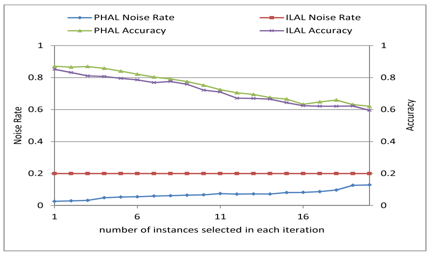
Fig. 6. Evaluation of sensitivity to input noise. Accuracy curves of (a) PHAL and (b) ILAL on 10 data sets with different percentages of noisy labels ranging from 2 to 30 percent with a step of 2 percent.

5.4.2 Noise in Expanded Labeled Sets

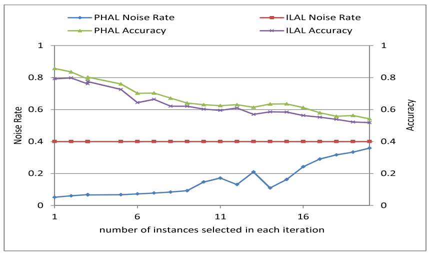
In traditional active learning, like ILAL (instance label based active learning), the queried instances, including their class labels, are used to form an expanded labeled set to update the current model. In contrast, the proposed method PHAL (pairwise homogeneity query based) employs a non-expert to provide pairwise homogeneity information to the queried



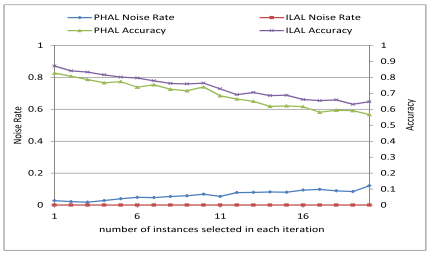
(a) Monks1 (input noise=0.0%)



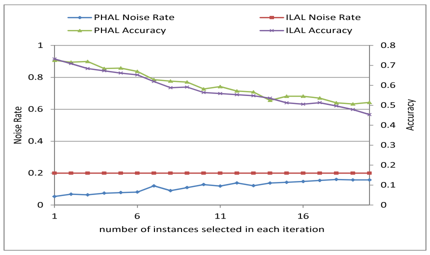
(b) Monks1(input noise=20%)



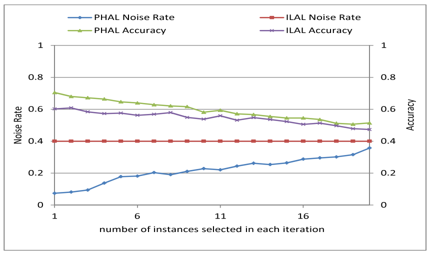
(c) Monks1(input noise=40%)



(d) Pima (input noise=0.0%)



(e) Pima (input noise=20%)



(f) Pima (input noise=40%)

Fig. 7. Detailed comparisons of accuracy (the right y-axis) and noise rate of the final expanded labeled set (the left y-axis) with respect to number of instances selected in each iteration (the x-axis) and input noise (in three columns). Results on all benchmark data sets are summarized in Fig. 8.

instance pairs and chooses a subset of instances with inferred class labels to expand the labeled training set. This implies that some inferred labels in the expanded labeled set of PHAL might be incorrect, even though the pairwise labels provided by the labelers are 100 percent accurate. In this experiment, we study noise labels and their impact in expanded data of PHAL and ILAL, by adding the same number of instances in the expanded data set for both methods. To quantify the results for comparisons, we define the "noise rate of the expanded set" as the percentage of instances with incorrect class labels in the whole expanded set.

In Fig. 7, we report the noise rates of the final expanded labeled sets (the left y-axis) and the accuracy curves of the classifiers trained from the corresponding final expanded labeled sets (the right y-axis) with respect to the input noise and the number of instances selected in each iteration. The number of instances added in the labeled set, in each iteration, varies from 1 to 20 with a step of 1, as indexed by the x-axis. In the case of input noise being 0 percent, the performance of ILAL is better than PHAL, because no noise is induced in the expanded labeled set of ILAL. On the other hand, the noise rates of PHAL remain at a low level, so performance is comparable to ILAL in this case. When increasing the input noise rates to 20 and 40 percent, PHAL outperforms ILAL with lower noise rates in the expanded labeled sets and higher classification accuracies. This is because PHAL only uses pairwise label homogeneity information to update the affinity graphs and its classification performance depends on the overall structures of affinity graphs. Introducing a fraction of erroneous edges may not result in significant noise in the expanded labeled sets. Another observation is that the accuracy curves of both methods drop as the number of selected instances in each iteration increases. This is because, given a fixed budget, the number of active learning iterations (i.e., model updating times) will decrease with a larger size of subset selected in each iteration, which in turn result in less accurate models. Therefore, the noise in the expanded labeled sets of PHAL increases as the number of instances selected in each iteration increases accordingly.

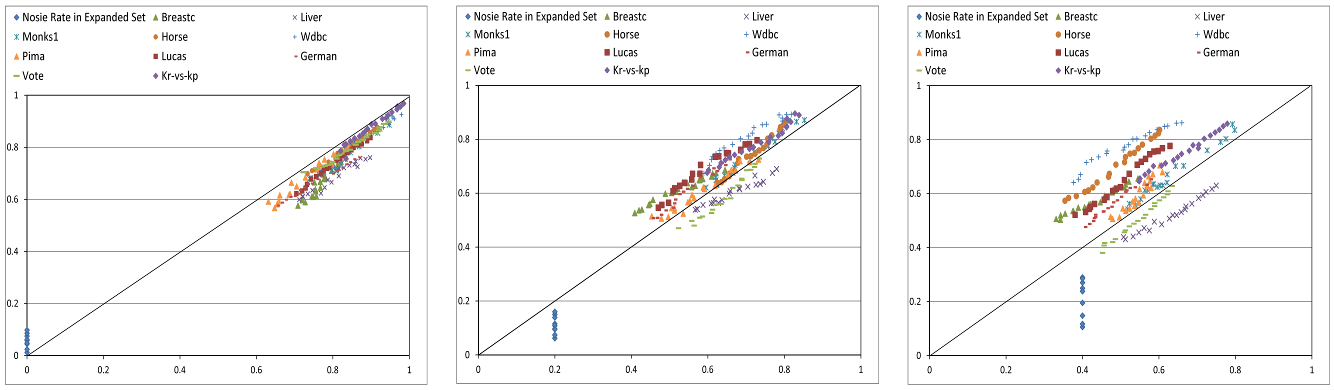
In addition to the detailed results in Fig. 7, we also compare the performance of PHAL and ILAL with respect to noise rate of expanded set and accuracy on all benchmark data. For each data set and each method, there are 20 accuracies corresponding to different batch sizes of selected subset in each iteration (ranging from 1 to 20 with a step of 1), and the noise rate of expanded set is the average with respect to the same range of batch sizes of selected subset. If we use the accuracies (or noise rate) of PHAL and ILAL, under the same setting, as y-axis and x-axis, respectively, it will produce 200 accuracy points and 10 noise rate points for PHAL and ILAL on all benchmark data sets. In Fig. 8, we report the 200 head-to-head accuracy points and 10 noise rate points, where a point above $y = x$ line indicates that the accuracy (or the noise rate) of PHAL is higher than that of ILAL, and vice versa.

The results in Fig. 7 show that when input noise is 0 percent, ILAL outperforms PHAL among all 200 observations in terms of accuracy and noise rate in expanded set. However, when input noise increases to 20 and 40 percent respectively, PHAL is much more accurately than ILAL on eight data sets. Moreover, PHAL always has a lower noise rate in expanded set than ILAL. These further assert that active learning algorithms taking instance pair-wise correlations into consideration, like PHAL does, are more noise tolerant and robust than traditional instance based active learning.

5.5 Comparison of Classification Performance

5.5.1 Comparison of Different Pair-Selection Strategies

Fig. 9 reports the performance of PHAL and the compared baseline methods on 10 benchmark data sets. All the methods are built in the same framework with different pair selection strategies, including Max-flow paths for PHAL, random selection for REAL, instance uncertainty



(a) (input noise=0.0%) (b) (input noise=20%) (c) (input noise=40%)

Fig. 8. Head-to-head accuracy and noise rate comparisons between PHAL and ILAL on all benchmark data sets. For each figure, the y-axis denotes the accuracy (or noise rate) of PHAL and the x-axis denotes the accuracy (or noise) of ILAL in the same setting. A point above $y = x$ line indicates that the accuracy (or the noise rate) of PHAL is higher than that of ILAL, and vice versa.

for USAL, edge uncertainty for UPAL, and a combination selection in PHAL(α). For PHAL(α), we investigate different values of α in $\{0.25, 0.5, 0.75\}$, which correspond to the percentages of pairs queried on the Max-flow paths. The x-axis indicates the numbers of queried pairs. All the compared methods in each active learning iteration have the same number of queried pairs, that is, the tth tick on the x-axis is the average number of accumulated instance pairs queried in the previous t iterations in PHAL. In each iteration, we include the same amount of labeled instances into the training set for each method. We compare the performance of different methods over 10 iterations with an increasing number of queried pairs.

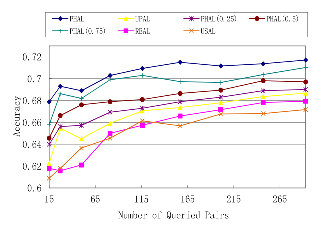
As the number of queries increases, the performance of all methods improves. This observation suggests that pairwise label homogeneity information does help improve model effectiveness no matter what kind of pair selection strategy is employed. It is very clear that the proposed PHAL method performs best on most data sets except “vote” and “kr-vs-kp”. These results indicate that pairs on the Max-flow paths are more effective for improving model performance than those pairs selected using other strategies. Moreover, PHAL(0.25) is slightly superior to UPAL; as α increases, the performance of PHAL(α) continually approaches to PHAL. We can thus assert that pairs on the Max-flow paths are more critical than the most uncertainty pairs for improving model performance. This is because Max-flow paths play an important role for generating the decision boundary. The pair weight adjustments on the Max-flow paths have more concentration on fitting the genuine decision boundary than those pairs selected based on uncertainty. Thus, selecting pairs on the Max-flow paths help accelerate finding the optimal decision boundary for classification. In contrast, the pairs selected based on high uncertainty may ignore the correlations of instances and introduce redundances and outliers.

Another interesting observation is that all the graph-based pair selection methods, which query pairs on the k-NN graphs, are superior to USAL, which queries any pairs of uncertain instances. Even UPAL that uses the same uncertainty metric outperforms USAL. These results imply that pairwise correlations play an important role on training an accurate model. In the graph-based pair selection methods, a data set is represented as k-NN graphs, in which the edges represent pairwise correlations of the data. In this case, the selected pairs have strong relationships from each other, the nonexpert labeler is more likely to provide accurate pairwise homogeneity information for the queried pairs. However, the pairs generated in USAL only consider uncertainties of individual instances, without incorporating correlations. In this case, it is possible for the nonexpert labeler to provide wrong answers for these disconnected pairs. Moreover, it is also possible to introduce outliers with high uncertainties into the model. These factors lead to the noticeable performance gap between the graph-based pair selection methods and USAL.

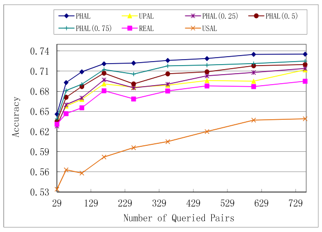
Overall, the results show that UPAL outperforms REAL in most cases, which suggest that pairs selected based on uncertainty are more informative than randomly selected pairs. Randomly selected pairs may introduce redundant information into the model. In contrast, uncertainty pairs can supplement the missing information for the underlying model to improve model performance.

5.5.2 Detailed Comparison of All Methods

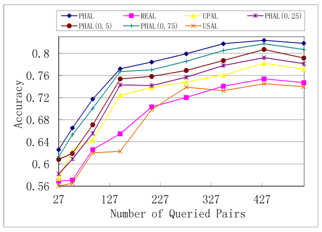
In each active learning iteration, we include a batch of labeled instances with high prediction confidences into the training set and retrain the model. This process repeats 10 times in total. For each method, we use the training set extended in each iteration to construct a J48 classifier for prediction, and record its accuracy on the same test set for a fair comparison. Tables 3a, 3b and 3c report the detailed performance of all the compared methods on 10 benchmark data sets in the third iteration, the sixth iteration, and the ninth iteration, respectively. Among all the methods, the proposed PHAL method achieves the best performance. UPAL is the second best method, but only marginally outperforms REAL. These results again validate that instance pairs selected on the Max-flow paths play an important role on training an accurate model. As we have discussed in Sections 5.2, UPAL integrates uncertainty measure in the pair selection strategy, which does help Min-cut based classifiers



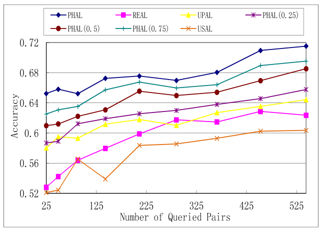
(a) Breastc



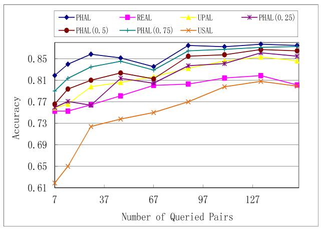
(b) German



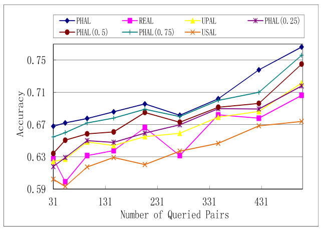
(c) Horse



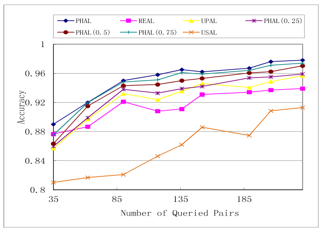
(d) Liver



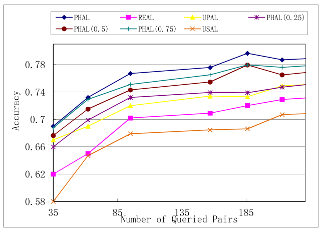
(e) Monks1



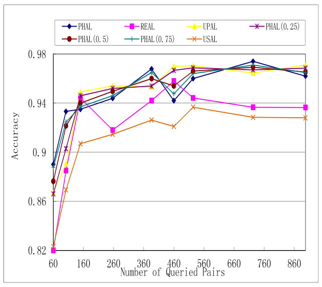
(f) Pima



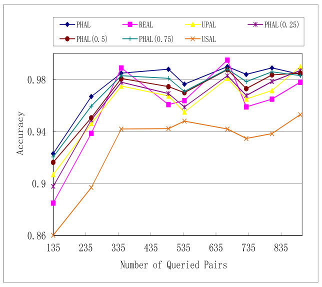
(g) Wdbc



(h) Lucas



(i) Vote



(j) Kr-vs-kp

Fig. 9. Performance comparison of PHAL and the baseline methods with different pair selection strategies on 10 data sets.

to find better cuts than random pair selection in REAL to some extent.

Obviously, USAL is inferior to all the graph-based pair selection methods, which take pairwise instance correlations into account. This is because USAL employs a pair selection strategy that only considers the uncertainties of the instances of a pair without considering their correlation. Although we design a pair selection scheme for USAL as introduced in Sections 5.2, the selected pairs seem of less help for improving model performance than the other methods, in which the selected pairs reflect the real data correlations in a graph topology. These results demonstrate that

TABLE 3
Detailed Performance Comparison

(a) After the 3rd iteration

Dataset	♯ of labeled pairs	PHAL	REAL	UPAL	USAL
Breastc	50	**0.689**	0.621	0.642	0.636
Liver	88	**0.652**	0.570	0.593	0.568
Wdbc	90	**0.950**	0.921	0.932	0.821
Monks1	29	**0.858**	0.764	0.797	0.723
Pima	96	**0.677**	0.636	0.645	0.617
Horse	94	**0.717**	0.625	0.643	0.620
Lucas	95	**0.767**	0.702	0.720	0.679
German	102	**0.709**	0.653	0.663	0.558
Vote	150	0.935	0.944	**0.959**	0.907
Kr-vs-kp	344	0.982	**0.989**	0.975	0.942
Average	114	**0.793**	0.742	0.757	0.707

(b) After the 6th iteration

Dataset	♯ of labeled pairs	PHAL	REAL	UPAL	USAL
Breastc	157	**0.715**	0.665	0.673	0.656
Liver	285	**0.669**	0.617	0.610	0.585
Wdbc	152	**0.968**	0.931	0.946	0.886
Monks1	88	**0.874**	0.802	0.831	0.769
Pima	274	**0.681**	0.631	0.659	0.636
Horse	278	**0.799**	0.720	0.748	0.738
Lucas	213	**0.787**	0.729	0.749	0.707
German	354	**0.726**	0.680	0.689	0.605
Vote	460	0.942	0.958	**0.970**	0.921
Kr-vs-kp	670	0.99	**0.995**	0.981	0.942
Average	293	**0.815**	0.772	0.785	0.744

(c) After the 9th iteration

Dataset	♯ of labeled pairs	PHAL	REAL	UPAL	USAL
Breastc	293	**0.717**	0.679	0.686	0.671
Liver	544	**0.715**	0.623	0.644	0.603
Wdbc	231	**0.978**	0.939	0.956	0.913
Monks1	154	**0.874**	0.801	0.845	0.798
Pima	508	**0.766**	0.706	0.721	0.673
Horse	509	**0.818**	0.747	0.771	0.739
Lucas	587	**0.809**	0.761	0.785	0.732
German	757	**0.735**	0.695	0.712	0.639
Vote	897	0.962	0.936	**0.971**	0.928
Kr-vs-kp	894	0.984	0.978	**0.99**	0.953
Average	537	**0.835**	0.786	0.808	0.764

pairwise correlations do play an important role for pairwise label homogeneity based methods to select informative pairs for labeling.

6 Conclusion

In this paper, we formulated a new active learning paradigm, in which an active learner is to query the label homogeneity of a pair of instances instead of querying the class label of an individual instance. We argued that obtaining pairwise label homogeneity information ("yes/no") is much easier and less costly than querying ground truth labels for individual instances, which normally requires strong expertise. The key technical challenges include (1) how to find important instance pairs for query; and (2) how to make use of the pairwise homogeneity information to improve the active learner. To solve the problem, we proposed to incorporate the query results into a Min-cut based active learner by adjusting the edge weights of unlabeled pairs on the Max-flow paths of an ensemble of k-NN graphs. After that, a subset of vertices with high prediction confidences are selected to be included in the labeled data set for model training. Extensive comparisons, on a number of benchmark data sets, demonstrate that the proposed method clearly outperforms the baselines. Furthermore, the proposed active learning paradigm is more robust to noisy labels than traditional active learning that queries class labels for individual instances.

REFERENCES

[1] H. Abe and H. Mamitsuka, "Query Learning Strategies Using Boosting and Bagging," *Proc. Int'l Conf. Machine Learning (ICML '98)*, pp. 1-9, 1998.

[2] A. Culotta and A. McCallum, "Reducing Labeling Effort for Structured Prediction Tasks," *Proc. 20th Nat'l Conf. Artificial Intelligence (AAAI)*, pp. 746-751, 2005.

[3] D. Angluin, "Queries and Concept Learning," *Machine Learning*, vol. 2, pp. 319-342, 1988.

[4] A. Blum and S. Chawla, "Learning from Labeled and Unlabeled Data Using Graph Mincuts," *Proc. 18th Int'l Conf. Machine Learning (ICML)*, pp. 19-26, 2001.

[5] A. Blum, J. Lafferty, M.R. Rwebangira, and R. Reddy, "Semi-Supervise Learning Using Randomized Mincuts," *Proc. 21st Int'l Conf. Machine Learning (ICML)*, 2004.

[6] D. Cohn, Z. Ghahramani, and M. Jordan, "Active Learning with Statistical Models," *J. Artificial Intelligence Research*, vol. 4, pp. 129-145, 1996.

[7] T. Cover and P. Hart, "Nearest Neighbor Pattern Classification," *IEEE Trans. Information Theory*, vol. IT-13, no. 1, pp. 21-27, Jan. 1967.

[8] P. Donmez and J. Carbonell, "Proactive Learning: Cost-Sensitive Active Learning with Multiple Imperfect Oracles," *Proc. 17th ACM Conf. Information and Knowledge Management (CIKM '08)*, pp. 619-628, 2008.

[9] P. Donmez and J.G. Carbonell, "Optimizing Estimated Loss Reduction for Active Sampling in Rank Learning," *Proc. Int'l Conf. Machine Learning (ICML '08)*, 2008.

[10] G. Druck, G. Mann, and A. McCallum, "Learning from Labeled Features Using Generalized Expectation Criteria," *Proc. 31st Ann. Int'l ACM SIGIR Conf. Research and Development in Information Retrieval (SIGIR)*, pp. 595-602, 2008.

[11] J. Du and C.C. Ling, "Asking Generalized Queries to Domain Experts to Improve Learning," *IEEE Trans. Knowledge and Data Eng.*, vol. 22, no. 6, pp. 812-825, June 2010.

[12] J. Edmonds and R. Karp, "Theoretical Improvements in Algorithmic Efficiency for Network Flow Problems," *J. the ACM*, vol. 19, pp. 248-264, 1972.

[13] M. Fang, J. Yin, and X. Zhu, "Knowledge Transfer for Multi-Labeler Active Learning," *Proc. European Conf. Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML/PKDD)*, Sept. 2013.

[14] Y. Fu, B. Li, X. Zhu, and C. Zhang, "Do They Belong to the Same Class: Active Learning by Querying Pairwise Label Homogeneity," *Proc. 20th ACM Int'l Conf. Information and Knowledge Management (CIKM)*, pp. 2161-2164, 2011.

[15] Y. Fu, X. Zhu, and B. Li, "A Survey on Instance Selection for Active Learning," *Knowledge and Information Systems*, vol. 35, pp. 249-283, 2013.

[16] Y. Guo and R Greiner, "Optimistic Active Learning Using Mutual Information," *Proc. 20th Int'l Joint Conf. Artificial Intelligence (IJCAI '07)*, pp. 823-829, 2007.

[17] S. Hoi, R. Jin, J. Zhu, and M. Lyu, "Batch Mode Active Learning and its Application to Medical Image Classification," *Proc. Int'l Conf. Machine Learning (ICML '06)*, 2006.

[18] T. Joachims, "Transductive Learning via Spectral Graph Partitioning," *Proc. Int'l Conf. Machine Learning (ICML)*, pp. 290-297, 2003.

[19] D.D. Lewis and J. Catlett, "Heterogeneous Uncertainty Sampling for Supervised Learning," *Proc. 11th Int'l Conf. Machine Learning (ICML '94)*, pp. 148-156, 1994.

[20] C.X. Ling and J. Du, "Active Learning with Direct Query Construction," *Proc. 14th ACM SIGKDD Int'l Conf. Knowledge Discovery and Data Mining (SIGKDD)*, pp. 480-487, 2008.

[21] T. Luo, K. Kramer, D.B. Goldgof, S. Samson, A. Remsen, T. Hopkins, and D. Cohn, "Active Learning to Recognize Multiple Types of Plankton," *Machine Learning Research*, vol. 6, pp. 589-613, 2005.

[22] H. Nguyen and A. Smeulders, "Active Learning with Pre-Clustering," *Proc. Int'l Conf. Machine Learning (ICML '04)*, pp. 623-630, 2004.

[23] E. Ni and C. Ling, "Active Learning with C-Certainty," *Proc. 16th Pacific-Asia Conf. Advances in Knowledge Discovery and Data Mining*, pp. 231-242, 2012.

[24] C. Papadimitriou and K. Steiglitz, *Combinatorial Optimization: Algorithms and Complexity*. Dover, 1998.

[25] P. Donmez and J.G. Carbonell, "Paired Sampling in Density-Sensitive Active Learning," *Proc. Int'l Symp. Artificial Intelligence and Math.*, 2008.

[26] V.C. Raykar, S. Yu, L. Zhao, A. Jerebko, C. Florin, G. Hermosillo-Valadez, L. Bogoni, and L. Moy, "Supervised Learning from Multiple Experts: Whom to Trust when Everyone Lies a Bit," *Proc. 26th Ann. Int'l Conf. Machine Learning (ICML)*, 2009.

[27] N. Roy and A. McCallum, "Toward Optimal Active Learning through Sampling Estimation of Error Reduction," *Proc. Int'l Conf. Machine Learning (ICML'01)*, pp. 441-448, 2001.

[28] R. Schapire, M. Rochery, M. Rahim, and N. Gupta, "Incorporating Prior Knowledge into Boosting," *Proc. Int'l Conf. Machine Learning (ICML '02)*, 2002.

[29] B. Settles, "Active Learning Literature Survey," Technical Report 1648, 2009.

[30] B. Settles and M. Craven, "An Analysis of Active Learning Strategies for Sequence Labeling Tasks," *Proc. Conf. Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1070-1079, 2008.

[31] H. Seung, M. Opper, and H. Sompolinsky, "Query by Committee," *Proc. Int'l Conf. Learning Theory (COLT '02)*, pp. 287-294, 2002.

[32] V.S. Sheng, F. Provost, and P.G. Ipeirotis, "Get Another Label? Improving Data Quality and Data Mining Using Multiple, Noisy Labelers," *Proc. 14th ACM SIGKDD Int'l Conf. Knowledge Discovery and Data Mining (KDD)*, 2008.

[33] S. Tong and D. Koller, "Support Vector Machine Active Learning with Applications to Text Classification," *Proc. Int'l Conf. Machine Learning (ICML '00)*, pp. 999-1006, 2000.

[34] S. Vijayanarasimhan, P. Jain, and K. Grauman, "Far-Sighted Active Learning on a Budget for Image and Video Recognition," *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, pp. 3035-3042, 2010.

[35] B.C. Wallace, K. Small, C.E. Brodley, and T.A. Trikalinos, "Who Should Label What? Instance Allocation in Multiple Expert Active Learning," *Proc. SIAM Int'l Conf. Data Mining (SDM)*, 2011.

[36] I. Witten and E. Frank, *Data Mining: Practical Machine Learning Tools and Techniques*. Morgan Kaufmann, 2005.

[37] X. Wu and R.K. Srihari, "Incorporating Prior Knowledge with Weighted Margin Support Vector Machines," *Proc. 10th ACM SIGKDD Int'l Conf. Knowledge Discovery and Data Mining (SIGKDD)*, 2004.

[38] Y. Yan, R. Rosales, G. Fung, and D. Dy, "Active Learing from Crowds," *Proc. Int'l Conf. Machine Learning*, 2011.

[39] L. Zhao, G. Sukthankar, and R. Sukthankar, "Incremental Relabeling for Active Learning with Noisy Crowdsourced Annotations," *Proc. IEEE Int'l Conf. Social Computing*, 2011.

[40] Y. Zheng, S. Scott, and K. Deng, "Active Learning from Multiple Noisy Labelers with Varied Costs," *Proc. IEEE 10th Int'l Conf. Data Mining (ICDM)*, pp. 639-648, 2010.

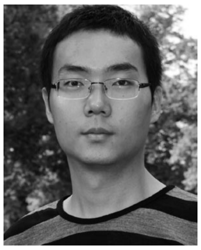
[41] X. Zhu and X. Wu, "Class Noise vs. Attribute Noise: A Quantitative Study of their Impact," *Artificial Intelligence Rev.*, vol. 22, pp. 177-210, 2004.

[42] X. Zhu, P. Zhang, X. Lin, and Y. Shi, "Active Learning from Data Streams," *Proc. IEEE Seventh Int'l Conf. Data Mining (ICDM)*, pp. 757-7627, 2007.

[43] X. Zhu, P. Zhang, Y. Shi, and X. Lin, "Active Learning from Stream Data Using Optimal Weight Classifier Ensemble," *IEEE Trans. Systems, Man, and Cybernetics, Part B*, vol. 40, no. 6, pp. 1607-1621, Dec. 2010.

[44] X. Zhu, "Semi-Supervised Learning Literature Survey," Computer Sciences TR 1530, 2008.

Yifan Fu received the ME degree in software engineering from Northeast Normal University, Changchun, China, in 2009, and the PhD degree in computer science from University of Technology Sydney, Sydney Australia, in 2013. She is currently a research associate in the School of Computing and Mathematics, Charles Sturt University, Australia (since August, 2013). Her research mainly focuses on machine learning and data mining, including active learning, ensemble methods, graph mining, and tensor decomposition.



Bin Li received the PhD degree in computer science from Fudan University, Shanghai, China, in 2009. He is currently a lecturer and previously was a postdoctoral research fellow in the Center for Quantum Computation & Intelligent Systems (QCIS), University of Technology, Sydney (UTS), Australia (since 2011). Prior to this, he was a postdoctoral research fellow at the Institut TELECOM SudParis, France (2009-2010). His research interests include machine learning and data mining methods and their applications to social media mining, recommender systems, and ubiquitous computing.

Xingquan Zhu received the PhD degree in computer science from Fudan University, Shanghai, China. He is an associate professor at the Department of Computer & Electrical Engineering and Computer Science, Florida Atlantic University. Prior to that, he was with the Center for Quantum Computation & Intelligent Systems, University of Technology, Sydney, Australia. His research mainly focuses on data mining, machine learning, and multimedia systems. Since 2000, he has published more than 160 refereed journal and conference papers in these areas, including two Best Paper Awards (ICTAI-2005 and PAKDD-2013) and one Best Student Paper Award (ICPR-2012). He was an associate editor of the *IEEE Transactions on Knowledge and Data Engineering* (2008-2012), and a program committee co-chair for the 23rd IEEE International Conference on Tools with Artificial Intelligence (ICTAI 2011) and the Ninth International Conference on Machine Learning and Applications (ICMLA 2010). He also served as a conference co-chair for ICMLA 2012. He is a senior member of the IEEE.

Chengqi Zhang (M'90-SM'95) received the PhD degree in computer science from The University of Queensland, Brisbane, Australia, in 1991, and the DrSc degree from Deakin University, Geelong, Australia, in 2002. He is currently with the University of Technology, Sydney (UTS), Sydney, Australia, where he is a research professor of information technology and the director of the UTS Priority Investment Research Center for Quantum Computation and Intelligent Systems. He has published more than 200 refereed research papers. His main research interests include data mining and its applications. He is a fellow of the Australian Computer Society. He is a senior member of the IEEE.

▷ **For more information on this or any other computing topic, please visit our Digital Library at** www.computer.org/publications/dlib.