Exploring Instance Correlation for Advanced Active Learning

Yifan Fu

A Thesis submitted for the degree of Doctor of Philosophy

Faculty of Engineering and Information Technology

University of Technology, Sydney 2013

Certificate of Authorship and Originality

I certify that the work in this thesis has not previously been submitted for a degree nor has it been submitted as part of requirements for a degree except as fully acknowledged within the text.

I also certify that the thesis has been written by me. Any help that I have received in my research work and the preparation of the thesis itself has been acknowledged. In addition, I certify that all information sources and literature used are indicated in the thesis.

Signature of Student

Production Note:
Signature removed prior to publication.

Acknowledgements

On having completed this thesis, I am especially thankful to my supervisor Prof. Xingquan Zhu and co-supervisor Prof. Chengqi Zhang, who had led me to an at one time unfamiliar area of academic research, and trusted me and given me as much as possible freedom to purse my own research interests. Prof. Zhu has taught me how to think and study independently and how to solve a difficult scientific problem in flexible but rigorous ways. He has sacrificed much of his precious time for developing my academic research skills. Prof. Zhang has also given me great help and support in life.

I am thankful to the group members I met in the University of Technology, Sydney, including Bin Li, Shirui Pan, Guodong Long, Lianyang Ma, and many others. I learned a lot from these smart people, and I was always inspired by the interesting and in-depth discussions with them. I enjoyed the wonderful atmosphere,being with them, of both academic research and daily life.

I am incredibly grateful to my mother for her generosity and encouragement. This thesis is definitely impossible to be completed without her constant support and understanding. I am also thankful to my friends who have companied me, though not always at my side, through the arduous journey of three and a half years.

Table of Contents

List of Figures

List of Tables

Abstract

Active learning (AL) aims to construct an accurate classifier with the minimum labeling cost by actively selecting a few number of most informative instances for labeling. AL traditionally relies on some instance-based utility measures to assess individual instances and label the ones with the maximum values for training. However, such approaches cannot produce good labeling subsets. Because instances exist some explicit / implicit relations between each other, instance-based utility measure evaluates instance informativeness independently without considering their interactions. Accordingly, this thesis explores instance correlation in AL and utilizes it to make AL's more accurate and applicable. To be specific, our objective is to explore instance correlation from different views and utilize them for three different tasks, including (1) reduce redundancy for optimal subset selection, (2) reduce labeling cost with a nonexpert labeler and (3) discover class spaces for dynamic data.

First of all, the thesis introduces existing works on active learning from an instance-correlation perspective. Then it summarizes their technical strengths/ weaknesses, followed by runtime and label complexity analysis, discussion about emerging active learning applications and instance-selection challenges therein.

Secondly, we propose three AL paradigms by integrating different instance correlations into three major issues of AL, respectively. 1) The first method is an optimal instance subset selection method (ALOSS), where an expert is employed to provide accurate class labels for the queried data. Due to instance-based utility measures assess individual instances and label the ones with the maximum values, this may result in the redundancy issue in the selected subset. To address this issue, ALOSS simultaneously considers the importance of individual instances and

the disparity between instances for subset selection. 2) The second method introduces pairwise label homogeneity in AL setting, in which a non-expert labeler is only asked "whether a pair of instances belong to the same class ". We explore label homogeneity information by using a non-expert labeler, aiming to further reducing the labeling cost of AL. 3) The last active learning method also utilizes pairwise label homogeneity for active class discovery and exploration in dynamic data, where some new classes may rapidly emerge and evolve, thereby making the labeler incapable of labeling the instances due to limited knowledge. Accordingly, we utilize pairwise label homogeneity information to uncover the hidden class spaces and find new classes timely. Empirical studies show that the proposed methods significantly outperform the state-of-the-art AL methods.

Chapter 1

Introduction

1.1 Motivation

Electronic data management systems have rapidly emerged in the past decades. All these systems computerize the data on operations, activities, and performances. Due to rapid development of storage, sensing, networking, and communication technologies, recent years have witnessed a gigantic increase in the amount of daily collected data. For decision-making purposes, these systems typically rely on domain experts to manually analyze a database. While data are becoming rapidly available, it becomes very difficult, or impossible, to manually extract all the useful knowledge from a huge amount of data. For many sophisticated learning tasks, sample annotation requires costly expert efforts, which raises significant issues for some large scale or domain specific problems as follows:

- *Fraud Detection* [107]. A banking expert needs to manually inspect each credit transaction to properly label a transaction as either a fraud or a normal transaction. With manual inspection and labeling, it may take an expert several years to inspect all transaction records in a month and annotate a small amount of fraud transactions.

- *Webpage Classification* [109]. When query results are returned by a search engine based on a specific keyword, we need to identify whether a web page is relevant to the keyword or not. It has shown that less than 0.0001% of all web pages have topic labels. Therefore, annotating thousands of web pages can be tedious and redundant.

- *Protein Structure Prediction* [23]. Protein structure prediction is to find a protein's secondary, tertiary, and quaternary structures from the protein's amino acid sequence. However, less than 1.2% of all proteins have known structures. For a specific protein, it takes months for a crystallographer to identify its structure in experiments.

When discovering patterns or rules from such large scale data sets, traditional data mining algorithms that labeling the whole data set for model construction is impossible in practise. This is mainly because labeling cost is very expensive and time consuming. If a very limited number of instances is labeled in random, these labeling information is insufficient to learn models with a good generalization capability. To address these issues, we need solve two essential questions: (1) Identification and collection of the relevant data from a huge information search space, and (2) Promptly reacting to situation changes and giving necessary feedback to both data collection and mining steps. The former is equivalent to find the most valuable instances to label from a very large instance space; the latter is concerned about how to present queried instances to a labeler, and collect their labeling information to update current model. Accordingly, active learning is proposed as a solution to these requirements, which explores and collects information based on current model, so that more information can help to update the model. The goal of active learning is to achieve high prediction accuracy by using as few labeled instances as possible [99]. Fig. 1.1 illustrates the general learning process of active learning. Suppose we are given a small labeled sample set as well as a relatively large unlabeled sample set, and we are allowed to interactively label a portion of unlabeled instances during the learning process. The key tasks of AL are (1) to identify most informative data and (2) to get information feedback in terms of the most informative data, such that the inclusion of these data into the labeled set can help improve models.

Generally speaking, including most informative instances to a labeled set can help improve the model performance with least labeling costs, or reduce the computational cost for succeeding mining procedures [141]. In practice, informativeness of a sample can be assessed by using uncertainty of instance based on models trained from current labeled sample set. For example, uncertainty is a common utility measure assessing a model's uncertainty in classifying an unlabeled sample. If a

sample's uncertainty is high, it implies that current models do not have sufficient knowledge in classifying the sample, and, presumably, including this sample into the training set can thus help improve the underlying models. Following this heuristic, the key challenge for most informative data identification is to design proper uncertainty metrics to evaluate the utility of an unlabeled sample. A large number of methods have been proposed to quantify and assess sample uncertainty in various ways. From an instance-selection perspective, these methods can be classified into the following two categories, with a progressive relationship.

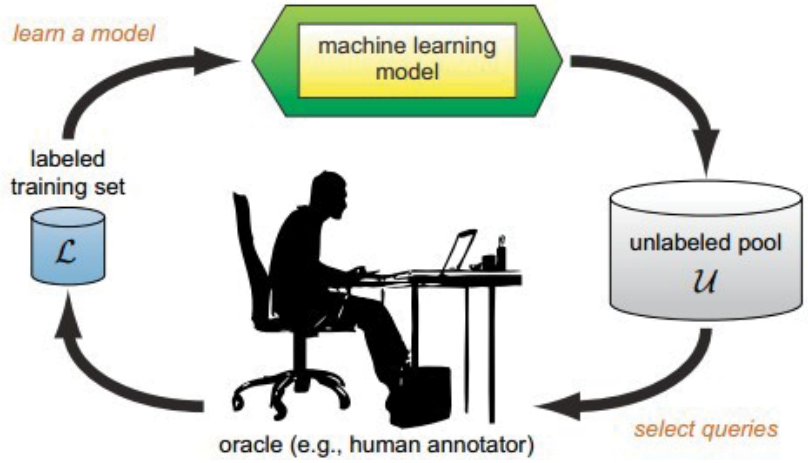


Figure 1.1: The general learning process of Active Learning [98]

1. **Utility metrics merely based on uncertainty of IID instances**: Methods in this category treat samples as independent and identically distributed (IID) instances, where the selection criteria only depend on the uncertainty values computed with respect to each individual instance's own information. Accordingly, one possible problem is that this type of approach may select similar instances in a candidate set, which results in redundancy in the candidate set.

2. **Utility metrics further taking into account instance correlations**: To take sample redundancy into consideration, utility metrics based on

instance correlation utilize some similarity measures to discriminate differences between instances. By uncovering inherent relationships between instances, instance utility calculated by this scheme integrates sample correlations, through which a selected candidate set may not always contain the "most uncertain "instances. Whereas, together, the selected instances form an optimal candidate set by balancing instance uncertainty and diversity.

A number of studies [35, 36, 85, 94, 111] 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, the labeler may be incorrect sometimes and give noisy answers though. 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 non-expert labelers scenarios. From the labeler's perspective, existing solutions mainly follow two directions: an expert can provide the truth grounds to the queries with their domain knowledge, whereas, a non-expert labeler may give simple or noisy answers to the questions.

In this thesis, we explore instance correlation in advanced active learning settings, where both expert labeler and non-expert labeler are taken into account for different AL issues, respectively.

From the labeler's perspective, existing solutions mainly follow two categories:

- **Active Learning with Expert Labeler**: Many methods [75] 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 considers the labeling cost on a large size of data set, which is addressed by selecting a subset of data with maximum utilities. This strategy explicitly provides ground truth 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 [66, 54] assumes all the (labeled and unlabeled) instances can be observed as a candidate pool. It first measures sample utility in the pool to decide which ones can maximally improve the performance of current model; then a learner queries their class memberships from a domain expert. Stream-based active learning [142, 143], 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 stream data. Query construction based active learning [4, 71] 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 Non-expert Labeler**: This strategy is used in the scenarios where no expert labeler exits for a certain learning task. Instead, a labeler or a group of cheap and noisy labelers is employed to address the labeling cost issue. Due to noisy labels provided by non-expert labelers, this strategy may not be appropriate in active learning. The tradeoff between labeling noise and labeling cost is a big challenge for active learning with non-expert labelers. Based on the pioneering work [103], [130] employs a group of non-expert labelers in the active learning setting by incrementally relabeling the most important instances. [115] formulates a budgeted selection task as a continuous optimization problem where the optimal selected subset maximizes the improvement to the classifier's objective, with a labeling cost budget constraint. Proactive learning [37] focuses on selecting an optimal labeler as well as an optimal instance at the same time using a decision theoretic approach.

In summary, taking instance correlation into account helps reduce sample redundancy for the most informative data identification. Moreover, employing non-expert labeler in AL setting is a promising solution to further reduce labeling cost, without a large performance loss.

1.2 Problem Statement and Solutions

In many real-world applications, instances are correlative in some fashion. Instance correlation may be presented explicitly or implicitly according to different learning tasks. Some learning tasks on networked data can provide explicit instance correlation straightforward (e.g. documents citation, co-authorship). While in other tasks, instance correlation is hidden information, which is needed to be exploited from an original data set. Take video object classification as an example, the inherent characteristics (i.e. the sequential continuity and multi-modalities of video streams) allow us to identify whether persons appearing in the consecutive frames are the same one or not. No matter instance correlation is explicit or implicit, incorporating instance correlation into AL can help boost system performance.

Accordingly, we are interested in utilizing instance correlation to solve some advanced active learning problems. To be more specific, we investigate different pairwise instance correlations from different views and utilize them to solve three challenges:

- **Exploring Optimal Instance Subset Selection with minimal redundancy** Traditional AL relies on some instance-based utility measures (such as uncertainty [112]) to assess individual instances and label the ones with the maximum values for training. It is argued that such approaches cannot produce good labeling subsets mainly because instances are evaluated independently without considering their interactions, the selected subset has a large number of redundant information. Alternatively, we proposed to achieve AL with optimal subset selection (ALOSS), where the key is to find an instance subset with a maximum utility value. To achieve the goal, we introduced disparity between a pair of data to our utility measure, which explores examples' similarity from feature space and prediction space simultaneously, namely, pairwise correlation in this context. Through using pairwise correlation, we guarantee the subset satisfies the requirement of data diversity and informativeness.

- **Exploring Instance Label Homogeneity to Reduce Labeling Cost**

Several studies have shown that active learning greatly helps reduce the labeling effort in various domains. However, traditional active learning depends on some unrealistic assumptions about labelers. For instance, active learning assumes there is a unique omniscient oracle. In real life, it is more possible and general to have multiple annotators with different areas of expertise. Active learning also assumes that the unique oracle is perfect, always providing a correct answer when requested. In reality, though, an "oracle"may be incorrect with a probability that may give noisy and erroneous answer of the question. Furthermore, an oracle is not always indefatigable, that is, it may refuse to answer if it is too uncertain or too busy. Active learning presumes the oracle is either free or inexpensive, and charges uniform cost in labeling tasks. Such an assumption is naive since cost is likely to be regulated by difficulty (amount of work required to formulate an answer) or other factors. To reduce labeling cost, we develop a pairwise label homogeneity query approach , in which a non-expert labeler is only asked "whether a pair of instances belong to the same class", namely, pairwise label homogeneity (i.e. pairwise correlation). Under such circumstances, our active learning aims to solve the following two challenges: (1) decide which pairs of instances should be selected for query, and (2) how to make use of the pairwise homogeneity information to improve an active learner.

- **Explore Class Spaces for Dynamic Data in Active Learning** Active learning traditionally focuses on labeling the most informative instances to formulate an accurate learner for some predefined learning tasks with known class labels. In dynamic data environments, some new classes may rapidly emerge and evolve, thereby making the labeler incapable of labeling the instances due to limited knowledge. In an extreme case, the whole active learning task may start from a "cold"state where the labeler does not know the number of classes exist in the data, and the actual class label for each queried instance. We refer to this problem as Cold-Start active learning, where no randomly labeled instances exist to kick-off the learning circle and the labeler only can provide pairwise label homogeneity information. The major challenges of cold-start active learning are to: (1) identify part (or all)

of the existing classes in the underlying data, (2) capture new (or unidentified) classes, and (3) label the most informative instances to train an accurate classifier for prediction.

1.3 Contributions

This thesis focuses on exploring and utilizing instance correlation to solve three important problems in AL. We list our contributions to each of them below:

- **Exploring Optimal Instance Subset Selection with minimal redundancy** Active learning (AL) traditionally relies on some instance-based utility measures (such as uncertainty) to assess individual instances and label the ones with the maximum values for training. One possible problem with individual-assessment-based approaches is that they may label similar instances, which, in turn, results in labeling redundancy. This is mainly because instances are evaluated independently without considering their interactions, and individuals with maximal ability do not necessarily form an optimal instance subset for learning. Alternatively, we propose to achieve AL with optimal subset selection (ALOSS), where the key is to find an instance subset with a maximum utility value. To achieve the goal, ALOSS simultaneously considers the following: 1) the importance of individual instances and 2) the disparity between instances, to build an instance-correlation matrix. Our contributions are as follows:

    1. Optimal subset v.s. the best individual: Individuals with the best capacity do not necessarily form an optimal subset, even if utility measures do take instance correlations into consideration. Our approach provides a new paradigm for AL by taking a group of instances as a whole for consideration.
    2. A new utility measure for instance characterization: By considering instance correlations, the proposed measure combines instance uncertainty and disparity to capture instance level correlations. As a result, the selected subset will contain best individuals with minimum redundant knowledge.

3. A general framework for AL: In addition to the clear optimization objective and theoretical basis, the theme of using instance correlations to form a matrix and employing semidefinite programming (SDP) to select an optimal subset can be applied to any utility measure and any learning algorithms.

- **Exploring Instance Label Homogeneity to Reduce Labeling Cost** Traditional active learning methods request a labeler to provide a class label to each queried instance. Such an active learning paradigm requires the labeler to be a highly skilled domain expert 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 non-expert labelers to do the labeling work without explicitly telling the class label of each queried instance. We propose a new active learning paradigm, named Pairwise Homogeneity based Active Learning (PHAL), in which a non-expert labeler is only asked "whether a pair of instances belong to the same class". This intuition is motivated by noise tolerate and label cost reduction. Even if the non-expert tells the wrong answer for a hard pair, 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. The contributions are as follows:

  1. Pairwise Label Homogeneity v.s. Specific Labels: Querying class labels of individual instances is expensive in the traditional active learning paradigm, even if the size of 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 a non-expert labeler.
  2. Max-flow based Pair Selection v.s. Random Pair Selection: To decide which pairs 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.

3. A Utility Measure for Instance Selection: Based on the prediction results of a Min-cut based classifier ensemble, the proposed utility measure uses a confidence based selection criterion to choose instances with high prediction confidence to extend the labeled data set.

- **Explore Class Spaces for Dynamic Data in Active Learning** Active learning traditionally focuses on labeling the most informative instances to formulate an accurate learner for some predefined learning tasks with known class labels, and a labeler (or an oracle) is provided to label each queried instance. In dynamic data environments, some new classes may rapidly emerge and evolve, thereby making the labeler incapable of labeling the instances due to limited knowledge. In an extreme case, the whole active learning task may start from a "cold"state where the labeler does not know how many classes exist in the data, and what is the actual class label for each queried instance. In this method, we refer to this problem as Cold-Start active learning, where no randomly labeled instances exist to kick-off the learning circle and the labeler only has weak knowledge to answer whether a pair of instances belong to the same class or not. The contributions are as follows:

  1. Class Discovery: An active learner needs to initially identify a set of candidate class labels in a given unlabeled data set, where the labeler does not know the genuine number of classes and true class label for each queried instance. A labeled training set is built to kick-off the active learning process.

  2. Class Exploration: An active learner further explores the undiscovered classes in the given data set. Moreover, whenever a new class emerges, our algorithm needs to accurately detect the new class and properly expands the training set to cover instances representing the new class.

  3. Instance Labeling: Our method will select and label the most informative

instances by asking a labeler who does not know the genuine class label of each instance, but can only answer whether a pair of instances belong to the same class or not.

1.4 Publications

- Yifan Fu, Xingquan Zhu, and Ahmed Elmagarmid. Active Learning with Optimal Instance Subset Selection. *IEEE Transactions on Systems Man and Cybernetics, Part B*, vol.43, no.2, 2013, pp.464-465.

- Yifan Fu, Xingquan Zhu, and Bin Li. A Survey on Instance Selection for Active Learning. *Knowledge and Information Systems*, vol. 35, iss. 2, 2013, pp. 249-283 .

- Yifan Fu, Xingquan Zhu. Optimal Subset Selection for Active Learning. *In Proceedings of the 25th AAAI Conference on Artificial Intelligence (AAAI-11)*, San Francisco, USA, 2011 [student poster].

- Yifan Fu, Bin Li, Xingquan Zhu and Chengqi Zhang. Do they belong to the same class: active learning by querying pairwise label homogeneity. *In Proceedings of the 20th ACM Conference on Information and Knowledge Management (CIKM-11)*, pp. 2161-2164, Glasgow, UK, 2011 [poster].

- Yifan Fu, Bin Li, Xingquan Zhu, and Chengqi Zhang. Active Learning without Knowing Individual Instance Labels: A Pairwise Label Homogeneity Query Approach. *IEEE Transactions on Knowledge and Data Engineering*, accepted, 2013.

- Yifan Fu and Xingquan Zhu. Active Learning with Unknown Class Space and Weak Labeling Knowledge. *IEEE Transactions on Neural Networks and Learning Systems*, under first round review, 2012.

1.5 Thesis Structure

The rest of thesis is summarized as follows:

Chapter 2: This chapter provides preliminary and definitions for the proposed models. It also summarizes major notations in the thesis.

Chapter 3: This chapter is a literature review that surveys existing works on active learning from an instance selection perspective. It summarizes major approaches in the field, along with their technical strengths/weaknesses, followed by a simple runtime performance comparison, theoretical label complexity analysis, discussion about emerging active learning applications and instance-selection challenges therein.

Chapter 4: This chapter presents an optimal instance subset selection method in active learning (ALOSS). It provides algorithm details, theoretical proof, time complexity and comparative experiments to valid its superiority to state-of-art methods.

Chapter 5: This chapter describes a pairwise homogeneity query approach in active learning (PHAL)in details. It explains the motivations and the principles of graph ensemble construction, provides theoretical basis and interpretations for the proposed work, analyzes the time complexity , and shows comparative experimental results with baseline methods and benchmark data sets.

Chapter 6: This chapter introduces a cold-start active learning method (CSAL). A detail of analysis of the comparisons results of experiments performed is also presented in this chapter.

Chapter 7: This chapter concludes this thesis and outlines the direction for future work.

Chapter 2

Preliminary and Notations

2.1 Definitions

Given a set of instances $D = \{e_1, e_2, \ldots, e_n\}$, where each sample $e_i = \{x_i; y_i\}$ is denoted by its feature values x_i and label values y_i, x_i is in a q dimensional feature space $\mathcal{F}$ and y_i is in an l dimensional label space $\mathbb{Y}$, so $e_i \in \mathcal{F} \times \mathbb{Y}$. Depending on the number of labels an instance contains, an instance can be divided into two types: a "single-label instance "and a "multi-label instance ".

Definition 1. ***Single-label Instance:*** *For a single-label instance e_i, it can be denoted by $e_i = \{x_i; y_i\}$, where $x_i =< x_{i1}, x_{i2}, \ldots, x_{iq} >$, and x_{ik} is the k^{th} feature value of e_i, and y_i is the class label of e_i.*

Definition 2. ***Multi-Label Instance:*** *For a multi-label instance , it can be denoted by $e_i = \{x_i; < y_{i1}, \ldots, y_{il} >\}$, where $x_i =< x_{i1}, x_{i2}, \ldots, x_{iq} >$, and x_{ik} is the k^{th} feature value of e_i,, and y_{ij} is the j^{th} class label of e_i.*

In typical active learning scenarios, users are given a data set D comprising a handful of labeled data sub-set $D^L = \{(x_1; y_1), (x_2; y_2) \ldots, (x_n; y_n)\}$ and a relatively large amount of unlabeled data $D^U = \{(x_1; ?), (x_2; ?) \ldots, (x_u; ?)\}$, with $D = D^L \bigcup D^U$. With limited labeling information, an accurate model can hardly be learned. To learn an accurate model, we need to label extra instances to get additional information. In an active learning environment, it is considered costly and time consuming to label all instances in D^U. Alternatively, an active labeler utilizes evaluation metrics to measure instance utility, and further selects instances with maximal utility values for labeling. There are two important concepts used in

utility metrics: uncertainty and correlation. The former is an evaluation criterion to measure the uncertainty of each single instance, whereas the latter measures the "correlations "between instances.

Definition 3. ***Uncertainty Metric:***. *Given an unlabeled sample set D^U and a label space $\mathbb{Y}$, uncertainty metric is a function f_u mapping from the instance space, D^U or $D^U \times \mathbb{Y}$, to a real number space R, where the "sample view"means the uncertainty metrics calculated based on the sample features, and "sample-label view"means the uncertainty metric calculated from both features and labels.*

$$f_u : \begin{cases} D^U \mapsto R, & sample \quad view \\ D^U \times \mathbb{Y} \mapsto R, & sample - label \quad view \end{cases} \tag{2.1}$$

Most of the previous algorithms evaluate the uncertainty only from the sample view. More recently, research work has focussed on evaluating the uncertainty from both sample-label views, which is considered to be more effective. In general, an uncertainty metric usually borrows information technology and statistical theory, such as "entropy "and "margin ", to measure instance utility. Different functions used in the selection metrics prefer different types of instances. For example, an "entropy "function tends to select instances minimizing the log loss of the model, whereas a "margin "function intends to choose the ones reducing the error rate by refining the decision boundary.

In addition to the above discussed "uncertainty "metric, one can also take "diversity "of selection into consideration, which can be enabled by evaluating the correlation of instances. By uncovering correlations between the instances, the selected labeling set helps generate a boundary much closer to the true decision boundary, compared to algorithms where only uncertainty is considered. Accordingly, properly estimating correlation among instances is important for selecting most informative instances in active learning.

Definition 4. ***Correlation Metric:*** *Given an unlabeled sample set D^U and a label space $\mathbb{Y}$, a correlation metric is a function f_c used to measure the correlation between a pair of instances x_i and x_j, where the correlation between any instance*

pair, $f_c(x_i, x_j)$, can be defined from three views:

$$f_c : \begin{cases} D^U \times D^U \mapsto R, & feature \quad view \\ \mathbb{Y} \times \mathbb{Y} \mapsto R, & label \quad view \\ (D^U, \mathbb{Y}) \times (D^U, \mathbb{Y}) \mapsto R, & both \quad views \end{cases} \tag{2.2}$$

By uncovering the pair-wise correlation in the instance set, two instances with a large correlation value are considered similar to each other, while the two with a small value are different. With Eq. (2.2), we can define the correlation between x_i and any other instances in D^U, denoted by $f_c(x_i)$, which is the mean of the correlation $f_c(x_i, x_j)$ for all $j \neq i$

$$f_c(x_i) = \frac{1}{|D^U|} \sum_{x_j \in D^U / x_i} f_c(x_i, x_j) \tag{2.3}$$

Eq. (2.3) represents the instance density in an unlabeled sample set. The larger the value $f_c(x_i)$, the higher the density around the instance x_i is. Therefore, the most representative instances in a set have the largest correlations. Intuitively, they are the centrex points with the highest density. On the other hand, the instances with smallest correlation values are located at the edge of the set, and are considered as outliers.

Based on the above "uncertainty metric "and "correlation metric", the "utility metric "for active learning is defined as follows.

Definition 5. ***Utility Metric:*** *Given an uncertainty metric f_u and/or a correlation metric f_c, utility metric is a function f used to evaluate the worth of labeling for unlabeled instances in D^U:*

$$f = \begin{cases} f_u, & if \quad not\ given\ f_c \\ f_u \bullet f_c, & if \quad given\ f_c \end{cases} \tag{2.4}$$

As the definition of f_u also explores the utility from two granularity levels: sample and sample-label pair. When utility metric integrates the correlation metric f_c, the instance utility is evaluated from both uncertainty and correlation views. As shown in Eq. (2.4), if f_u increases, uncertainty becomes larger, and so does u. However, only taking uncertainty into account results in a redundancy issue, while we assess instance utility based on correlation that may select diverse instances. To

this end, Eq. (2.4) is a trade-off function between the two views to assess instance utility.

Definition 6. ***Query Strategy:*** *By choosing a certain utility metric, a query strategy evaluates the informativeness of unlabeled instances based on the prediction result of current model (to calculate uncertainty) and/or data distributions (to calculate correlations), then selects the most informative instances for labeling.*

With a specific query strategy, one can rank instances according to their utility values. The instances on the top of the queue are the most ambiguous ones for the current model, whereas the ones at the bottom of the queue are the most certain instances for the model. The top b (where b is the size of an optimal sampling sub-set) form a maximal utility sub-set to be included in the training set. Following this approach, general procedures for active learning process are described in Algorithm 1.

Algorithm 1 General Process of Active Learning

Input: Initial labeled instance set D^L, Unlabeled instance set D^U, size of the training set m
Output: Model $\hbar$

while training size $\leq m$ **do**
  $\hbar \leftarrow$ learn a model based on D^L;
  $D^U \leftarrow D \setminus D^L$;
  for each instance x_i in the D^U **do**
    $f_i \leftarrow f(x_i, \hbar)$;
  end for
  $x^\star \leftarrow \arg\max_i(f_i)$;
  $D^L \leftarrow D^L \bigcup x^\star$;
  $D^U \leftarrow D^U \setminus x^\star$;
  $\hbar \leftarrow$ update the model based on D^L;
end while

In Algorithm 1, a model is trained from an initial small labeled training set D^L. After that, all instances in the unlabeled pool D^U are queried by a learner. On the basis of the query results evaluated by a utility metric, the learner requests to select most potential instances to be labeled by an oracle (*i.e.* a labeler). After that, the new labeled instances are directly added to the training set D^L to update the model. This process repeats until the model achieves the desired prediction accuracy or the pre-set number of instances are labeled in the training set.

2.2 Notations

Major notations used in this paper are summarized in Table 2.1.

Table 2.1: Notations used in the paper

Symbols	Explanations
D^L, D^U and D^T	the labeled, unlabeled and test data sets, respectively
n	the number of instances in D^U
$\mathcal{F} = \{\alpha_1, \alpha_2, \ldots, \alpha_q\}$	a q dimensional feature space
$\mathbb{Y}$ and Y	the total (unknown) and discovered class space
$e_i = \{x_i; y_i\}$	a single- label instance in a data set
$x_i =< x_{i1}, x_{i2}, \ldots, x_{iq} >$	the feature space of e_i, where x_{ij} the j^{th} feature value of e_i
y_i	the class label of e_i
$e_i = \{x_i; < y_{i1}, \ldots, y_{il} >\}$	a multiple-label instance, where y_{ij} the j^{th} class label of e_i
$\hbar$	a prediction model
Υ	the budget (total number of queried pairs during active learning)
Δ	the data set selected for labeling in each iteration, where $\|\Delta\|$=b
$P_\hbar(y_i\|x)$	the probability of x belong to class i given a model $\hbar$
η	the number of parameters in a model $\hbar$
$\hbar(.)$	a prediction function mapping from the feature space to the class label space
$E_\hbar = \{\hbar_1, \hbar_2, \ldots, \hbar_m\}$	a classifier ensemble
$g = (V, \mathfrak{E})$	a graph, where each node $v_i \in V$ denotes an instance x_i, each edge $\mathfrak{e}_{ij} =< v_i, v_j >\in \mathfrak{E}$ describes some relationship between two nodes v_i and v_j
V^L and V^U	the labeled and unlabeled vertex sets, respectively

Symbols	Explanations
$V = V^L \cup V^U$	the vertex set of a graph
$\mathfrak{E}_i$	the edge set of the ith graph
$g_i = (V, \mathfrak{E}_i)$	the ith graph
$\mathbb{G} = \{g_1, \ldots, g_m\}$	a set of graphs
$P_c(v_i)$	the confidence of the predicted label for v_i
$d_{P_{i,j}}$ and $d_{\mathcal{F}_{i,j}}$	the prediction distance and the feature distance between a pair of instances (x_i *vs.* x_j), respectively
$w_{i,j}$	the edge weight between vertices i and j
D^t, $D^{\triangle t}$and $D^{t+\triangle t}$	the data set at a specific time t, the new data set after a time interval $\triangle t$ and the data set at a specific time $t + \triangle t$, respectively
Γ and k	the optimal subset used in the process of *Class Discovery* and its size, respectively
$\mathfrak{T}_i, i = 1, 2, \ldots, n$	the ith Minimum Spanning Tree learned on Γ
$\coprod_{\mathfrak{T}} = \{\mathfrak{T}_1, \ldots, \mathfrak{T}_n\}$	a set of Minimum Spanning Trees
$\mathbb{E}^s$ and $\mathbb{E}^u$	a classifier ensemble and a cluster ensemble, respectively
$\mathbb{E} = \mathbb{E}^s \cup \mathbb{E}^u$	a consensus ensemble learning model
β, β_s, β_u	the size of $\mathbb{E}, \mathbb{E}^s$ and $\mathbb{E}^u$, respectively
$\hbar_a \in \mathbb{E}$	the ath model in $\mathbb{E}$
$\hbar_a^s$ and $\hbar_a^u$	the ath model is supervised and unsupervised, respectively
κ^a	the number of cluster generated on each model $\hbar_a$
$\mathcal{G}_l^a (1 \leq l \leq \kappa^a)$	the$l$th group generated on the model $\hbar_a$
$P(y\|x, \mathbb{E}), P(y\|x, \hbar_i), P(y\|x, \mathcal{G}_l^a)$	the probability of x belonging to the class y given $\mathbb{E}$, $\hbar_a$ and $\mathcal{G}_l^a$, respectively
$\widetilde{P}(y = q\|x, \mathcal{G}_l^a), \widehat{P}(y = q\|x, \mathcal{G}_k)$	the initial and finial probability of x belonging to the class q if x is in $\mathcal{G}_l^a$
Λ	the total groups generated by $\mathbb{E}$
$J(\mathcal{G}_i, \mathcal{G}_j)$	the similarity between group $\mathcal{G}_i$ and $\mathcal{G}_j$

Symbols	Explanations
$P(\hbar_a\|x)$	the weight of the model $\hbar_a$
ϕ_a^i	the probability of x belong to class i given $\hbar_a^s$
$\Phi_j = (\phi_a^1, \ldots, \phi_a^L)$	the prediction distribution of an instance x_j
$\delta(\hbar_a, \hbar_b\|x)$	the similarity between $\hbar_a$ and $\hbar_b$ on x's label prediction
$P_e(x\|y)$ and $P(x)$	the occurring probability of x in the class y and the prior probability of x, respectively
$P_t(y \in Y \cup y_{new}\|x)$	the probability of x belonging to each existing class and the new class
$P^m(x)$	the misclassification probability of the instance x
$P_{error}(y\|x)$	the posterior probability of x belonging to y
Θ	the most utility subset added at each iteration in the process of *Class Exploration and Instance Labeling*

Chapter 3

Literature Review

This literature review provides an in-depth study on how existing active learning methods explore uncertainties and correlations to select instances for labeling. Our main objective is to (1) summarize and categorize instance-selection methods to provide a big picture for active learning; and (2) compare and analyze the strengths and deficiencies of existing approaches.

3.1 Instance Correlation

Instance correlation represents a broad class of relationships in different contexts. It may refer to dependency between two random variables or two sets of data, such as the correlation between the physical statures of parents and their offsprings, and the correlation between the demand for a product and its price. It also may explain similarity between two examples in many classification tasks, such as image recognition and recommendation systems. Instance correlation usually exists in either an explicit or an implicit way in practise. Explicit correlation can be collected directly without further exploration, while hidden correlation have to be exploited from an original data set with various data mining technologies. Many studies have suggested that integrating instance correlation into active learning can effectively improve system performance. When reviewing existing research regarding incorporating instance correlation into AL scenario, involved instance correlation can be divided into three groups:

1. **Content-based Instance Correlation** explores instance relationships from their feature and/or label spaces. Based on the different contents exploration,

this kind of correlation can be further categorized into three subgroups:

- *Feature based instance correlation* measures pairwise instance similarity between two vectors of feature spaces. Usually a similarity measurement [133] or a correlation matrix [109] on features is utilized to compare pair-wise similarities of instances, so the informativeness of an instance is weighted by average similarity on its neighbors. The algorithms rely on clustering algorithms to group instances, and select the most representative instances in each cluster to form an optimal sub-set with maximum uncertainty. This strategy integrates the information density based metrics and the traditional uncertainty measures to evaluate the potential of an instance.
- *Label based instance correlation* exploits complex label relevancy among the class labels. This category is widely used in multi-label learning tasks [12], where an instance can have more than one label. In general, the labels have constraints or relations between each other. Therefore, if we are given a portion of labels for an instance, active learning can automatically infer its additional labels using the constraints.
- *Feature and label based instance correlation* Besides comparison on the feature similarity, this scheme further explores correlations based on the neighbor's prediction information for a specific instance. Therefore, it integrates the result from feature and label dimensions to assess correlations. This setting is very suitable for mining tasks on a data set with a complicated structure. However, traditional similarity metrics compute average similarity over all the pairs of instances, so that the computational cost is expensive as the size of a data set grows rapidly. Some variance approaches [79, 42] are developed following the idea that each cluster has a density center. A simplified version is to compute the similarity between each instance and the center instance by considering the whole set as a cluster, and calculate the average of the cumulative result [25].

2. **Link-Based Instance Correlation** utilizes the link information to explore

relationships between neighbor nodes in a graph/ or a complicate network. It have been broadly used for classifying networked data, for example, the linked web pages and the friendship network. A number of models have been proposed to explore link information, such as Conditional Random Fields [65], Continuous Bayesian network [86], Factor Graph models [110], Collective Learning [59], and Semi-supervised Learning over graphs [135].

3. **Content and Link based Instance Correlation** takes not only link information but also individual instance content information into consideration to AL, which is considered as a combination of above two categories. It has been applied to networked data classification as well. Some works like [140] combine semi-supervised learning and active learning based on Gaussian random fields and harmonic functions; [77] integrates graph-metrics and empirical risk minimization. [10] proposes an active learning method for networked data built upon uncertainty sampling, committee-based sampling, and clustering. Also, there have been explorations of active learning on special graphs, such as trees [21].

Although instance correlation contains three different subgroups, this thesis only focuses on exploring various content-based instance correlations in active learning. Our objectives are to explore these instance correlations from different views and utilize them for three different tasks, that are (1) reduce redundancy for optimal subset selection, (2) reduce labeling cost with an nonexpert labeler and (3) discover class space for dynamic data sets.

3.2 Categorization for active learning methods

For instance selection in active learning, there are two major research issues: 1) "how to select unlabeled instances for labeling "and 2) "how to evaluate selected unlabeled instances ". By combining the two research issues into a single view, we obtain a hierarchical structure of our categorization for active learning in Fig. 3.1. It is worth noting that *how to select* can be divided into two major categories and seven subcategories (the solid rectangle on the top); while *how to evaluate* only comprises two categories (the solid rectangle at the bottom). The two dimensions

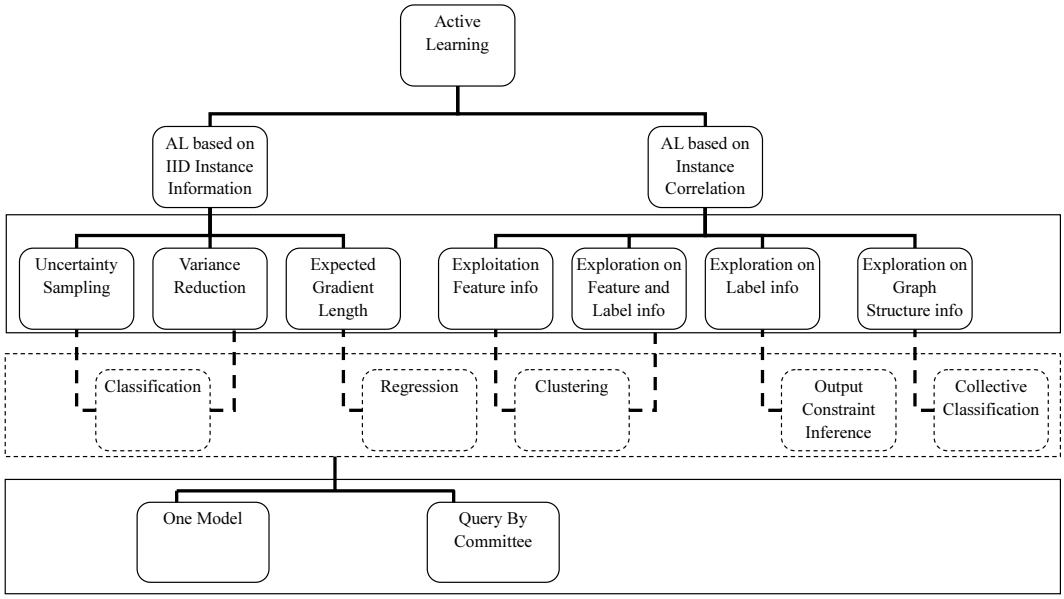


Figure 3.1: Hierarchical structure of the categorization for active learning. The query strategies in the top solid rectangles demonstrate "how to select unlabeled instances for labeling ", and the models in the bottom solid rectangles demonstrate "how to evaluate selected unlabeled instances ". The methods in the dash rectangle boxes explain the implementations of a query strategy.

are coupled in certain machine learning tasks such as classification, regression, and clustering (middle dashed rectangle). In the next two sections, we give a high level introduction of AL based on IID Instance Information, and highlight AL based on Instance Correlation, respectively.

3.3 Active learning based on IID instance information

Definition 7. ***Active Learning based on IID instance information:*** *Given an unlabeled sample set D^U, a labeled training set D^L, and an uncertainty function $f_u(.)$ for instance utility evaluation, i.e., $f(.) = f_u(.)$, active learning based on IID instance uncertainty aims to help construct an accurate model by labeling the most informative individual instances in D^L to form the training set D^L, according to $f(.)$.*

3.3.1 How to Select Unlabeled Instances for Labeling

Active Learning based on IID instance information is commonly applied in single-label learning tasks, where a utility function is designed based on the input feature space. Moreover, the assumption that instances in the unlabeled set are independent in this scheme makes an uncertainty evaluation function immediately available to calculate instance utility. Accordingly, methods in this category normally rank instances simply based on an uncertainty metric, and choose the ones with the largest uncertainty values for labeling. We categorize query strategies into three groups based on "how to select ":

- *Uncertainty Sampling* emphasizes on labeling the most uncertain instances, by using diverse uncertainty schemes such as least confidence, margin, and entropy [55].
- *Expected Gradient Length* focuses on querying instances that cause the maximal change to the current model [100].
- *Variance Reduction* favors instances that minimize the square loss of a learner [3].

3.3.2 How to Evaluate Selected Unlabeled Instances

Another important issue is how to evaluate instance utility value with the above query strategies. Some algorithms employ a single model, so an instance utility relies on the model prediction result, where the most "ambiguous "instance is the most uncertain one for the model. Others use a set of models to form a "query committee "[101]. In this approach, class label prediction for an instance rests on the majority voting result in the committee. The most informative instance is the one with the most disagreement prediction from the classifier ensemble.

3.4 Active Learning Based on Instance Correlations

Many studies suggest that active learning based on single instance information tends to select outliers [94]. Moreover, the selected instances may also have redundancy. These are because that the uncertainty of a specific instance is calculated based on its own utility with instance correlation inherently ignored. In order to address this issue, instance correlations are further taken into account to avoid information redundancy.

Definition 8. ***Active learning based on instance correlations:*** *Given a domain D, consisting of an unlabeled dataset D^U and label space $\mathbb{Y}$, an utility function $f(.) = f_u(.) \times f_c(.)$, where $f_u(.)$ is an uncertainty metric function and $f_c(.)$ is a correlation metric function, instance correlation based active learning tries to select most informative sample/sample-label pairs according to $f(.)$.*

3.4.1 How to Select Unlabeled Instances

In instance correlation based active learning setting, a correlation function is integrated into a utility function with the assumption that instances are dependent on each other. Compared with Definition 3, this strategy selects instances from both correlation and uncertainty views rather than from the uncertainty aspect only. For some multi-label tasks, we are given a portion of labels of an instance, so a sample-label pair can be considered as an object to assess its utility instead of using feature values of the instance only. Based on different types of correlations, we categorize this setting into three sub-categories as shown in Table 3.1, and summarize the relationship between traditional machine learning and various sub-settings in this strategy.

3.4.1.1 Exploration on Feature Correlation

In this scheme, most algorithms exploit feature correlations by using clustering methods. Many studies have shown that semi-supervising algorithms can facilitate the clustering process. In active learning setting, a small labeled training set and a

Table 3.1: Different settings of active learning based on instance correlation

Sub settings	Related Areas	Instance View	
		Feature	Feature-label pair
Exploration on Feature Correlation	Clustering correlation matrix analysis	√	
Exploration on Label Correlation	Multi-label learning Constraint inference	√	√
Exploration on Feature and Label Correlation	Clustering	√	
Exploration on Graph Structure	Collective classification	√	

large unlabeled pool provide a semi-supervised learning environment, which makes it possible to incorporate clustering algorithms to group instances before an active query starts. A simple clustering partitions instances based on features information. In the clustering algorithms, a similarity measure is developed as grouping criteria. Thus, the utility of a single instance is weighted by the average similarity over all other instances in the unlabeled set, which is described as follows:

$$x^{\star} = \underset{x}{argmax}\, f_u(x) \times (\frac{1}{n}\sum_{u=1}^{n} sim(x, x^u))^{\beta} \tag{3.1}$$

Where $f_u(.)$ is the uncertainty function of a single instance, which is calculated according to the definitions given in Section 2.1, n denotes the size of unlabeled data set, $sim(x, .)$ denotes the similarity function to evaluate the distance between two instances, and β controls the importance of density term. The utility of an individual instance is weighted by average similarity over the unlabeled data set. The most representative instances selected from each group form an optimal sub-set with maximum uncertainty.

Traditional similarity metrics compute the average similarity for each instance. For a data set with large volumes, the computational cost can be expensive. Some alternative approaches [79, 42] have been developed based on the idea that each cluster can be considered as a dense region in the input space. Chen & Subramani [25] implemented a simplified version of information density function by computing the similarity between instances and the mean point. This function requires less computation than Eq. (3.1) under assumption that there is only one cluster in the data set. Depending on different corpus used in real-world applications, the

first term of Eq. (3.1) can be replaced by other uncertain sampling schemes, such as least confidence, margin and so on. In addition to the exploration on features correlation using clustering algorithms, [109] utilizes a feature correlation matrix to measure the property difference between pairwise examples.

Furthermore, the above approaches for feature correlation fall into two cases based on "how to evaluate selected unlabeled instances ". Traditional algorithms represent instances using a feature vector, so that a similarity function evaluates instance correlation in terms of feature values. In fact, correlation exploration on feature values is insufficient. Take text classification as an example, conventionally, similarity comparison on pairwise documents is measured by accumulating the occurrence rate of each term/word. This approach can only group documents with many identical words, while semantic relations like acronyms and synonyms are ignored. To this end, the *content (or semantic)-based similarity* measure has been proposed. Huang et al [57] utilized Wikipedia to design a concept-based representation for a text document, and evaluated the content correlation of pairwise documents at the granularity of Wikipedia concepts to find instance-level constraints. Nguyen & Smeulders [85] proposed a representative active learning algorithm considering the prior distribution of the instance set. However, one weakness of this method is that it is only effective in linear logistic regression, which means all the clusters are modeled with the same parameters. To address this issue, Yan et. al. [126] introduced a new framework for semi-automatic annotation of home video under the active learning strategy. In their design, an off-line model is built on a small labeled data set. The initial model is adjusted with the local information of a specific home video obtained online. In the paper, they used four semantic concepts to present the labeling process.

The relationships between the algorithms used in this strategy and evaluation granularity are summarized in Fig. 3.2.

In the previous work, various similarity measures are designed for feature correlation exploration. There are three commonly used similarity functions.

A) Cosine Similarity

Cosine similarity is a measure which calculates similarity between two vectors by

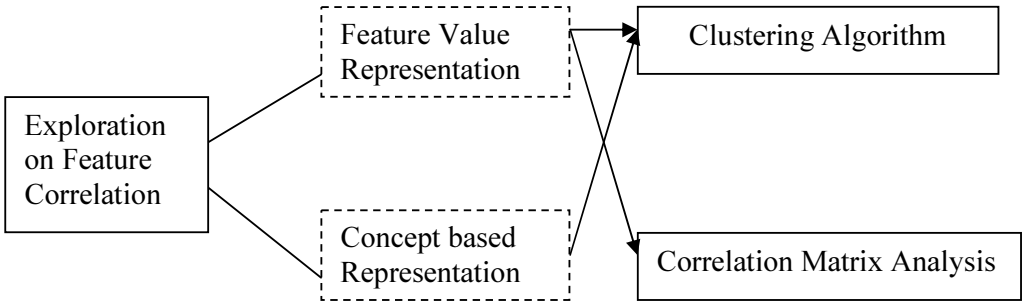


Figure 3.2: Relationships between the algorithms used in exploration on feature correlation and evaluation granularity

finding the cosine of the angle between them. The formula is defined as follows:

$$sim_{cos}(x, x^u) = \frac{\overrightarrow{x} \cdot \overrightarrow{x_u}}{\| \overrightarrow{x} \| \times \| \overrightarrow{x_u} \|} \tag{3.2}$$

Where $\overrightarrow{x}$ is a fixed-length feature vector of an instance x, $\cdot$ represents inner dot product of two vectors, and $\| \cdot \|$ is the vector norm.

Cosine similarity is widely used in sequence instance classification, especially in sequence classification tasks. For instance, an improved k-Nearest Neighbor Algorithm based Cosine Similarity is applied for text classification [68]. A multi-criteria-based active learning for name entity recognition exploits the utility of an instance from three criteria, including formativeness, representativeness, and diversity [102]. Meanwhile, it employs two criteria combinations to select instances. Cosine similarity is also used to measure similarity between words in the representativeness strategy.

Cosine similarity is very effective for instances with low dimensional features, which avoids unnecessary computation. It evaluates the similarity on the original input space without sub-space transition or matrix connection. Nguyen & Li [84] proposed a cosine similarity metric for face verification. Unlike comparing two faces based on the traditional Euclidean distance metric in a transformed sub-space, they used cosine similarity on the original input space. This similarity metric also plays an important role on text-independent speaker verification. Classical variable score normalization techniques define speaker sub-space and channel factors separately, and estimate them jointly. To reduce computing complexity, Shum et al. [105] proposed a new score normalization scheme with cosine similarity, effectively

reducing the additional computation at each adaptation update process.

B) KL Divergence Similarity

Kullback-Leibler divergence is a non-symmetric measure capturing difference between two instances. The utility of each instance is weighted by the summation of the difference over the rest instances in an unlabeled pool. The exponential KL divergence similarity function is defined as

$$sim_{KL}(x, x^u) = exp(-\gamma_1 \sum_{j=1}^{J} P(\alpha_j|\overrightarrow{x}) log \frac{P(\alpha_j|\overrightarrow{x})}{\gamma_2 P(\alpha_j|\overrightarrow{x_u}) + (1-\gamma_2)P(\alpha_j)}) \tag{3.3}$$

The smoothing parameters γ_1 and γ_2 control the divergence speed and encoded distribution in the denominator, respectively. $\overrightarrow{x_u}$ is a J-dimensional feature space, $P(\alpha_j|\overrightarrow{x})$ denotes posterior probability of containing a feature α_j. $P(\alpha_j)$ is simply a marginal probability of feature α_j over all instances in the pool.

KL-divergence has been applied to active learning to evaluate different class output distributions between the classifiers in the ensemble, such as named entity recognition [7] and information extraction [61]. Because KL-divergence has a non-negative value, the larger the value, the more different the pair is, and a zero KL-divergence value indicates two identical distributions. When taking a peaked distribution as a benchmark of certainty, KL-divergence is very similar to cost-testing [81].

Due to the non-symmetric property of the measure, the similarity between pairwise instances should be computed twice, implying a high computational cost. In order to improve the computational complexity, Zhao et al. [131] developed an active learning model based on this strategy for telecom client credit risk prediction.

C) Gaussian Similarity

Another exponential similarity measure used to estimate information density is called Gaussian Similarity, which is an exponential Euclidean distance aggregating all the distances on each feature. The formula is represented as

$$sim_{Gauss}(x, x^u) = exp(-\sum_{j=1}^{J} \frac{(\overrightarrow{x}^j - \overrightarrow{x_u}^j)^2}{\alpha^2}) \tag{3.4}$$

where α^2 is the variance in the Gaussian distribution. Different variance can be

set for different feature, but there is a challenge for setting appropriate parameters. Moreover, it has been suggested that a model with several parameters does not improve the effort of representing the similarity. A semi-supervised learning using Gaussian fields and harmonic functions integrates this query scheme to select most informative instances [136]. [139] further introduces a combination of active learning and semi-supervised learning under the above framework. Moreover, in [64], graph kernel functions based on the inverted square Euclidean distance and Gaussian similarity, respectively, are evaluated in the context of rating prediction problems. The experimental results show that Gaussian functions outperforms other kernel functions in most cases.

3.4.1.2 Exploration on Label Correlation

For multi-label and multi-task problems, the output space contains multiple class related labels, which means that outputs are subject to some inherent correlations, such as agreement, inheritance, exclusive and so on. These correlations provide valuable information for reducing prediction cost. To this end, many studies have leveraged output constraints to improve the learning process [23, 24, 20], where label correlation is used for model parameter estimation or inference on unlabeled instances. [128] introduces a cross task information metric for multi-task active learning, whose utility is measured by all the relevant tasks reachable through task output constraints. This framework combines uncertainty sampling metric with inconsistency of prediction on coupled tasks. The main procedures of the algorithm are as follows.

(1) *Choose a reward function for a single task*: Given an unlabeled sample x, a utility function UI is used to compute its importance for improving model performance, which can be denoted by

$$UI(Y, x) = \sum_{y} \hat{p}(Y = y|x)\varrho(\hat{p}, Y = y, x) \tag{3.5}$$

Where $\hat{p}$ represents the posterior probability of sample x belonging class y, $\varrho()$ is a regard function. This formula accumulates the reward on each possible label y.

(2) *Specify the constraint set between task outputs* By constructing the propagation rules, the algorithm computes the set of propagated outcomes for each possible

label. The set of propagated outcomes $Propc(Y_i = y_i)$ is defined as the inferred outcome labels from task $Y_i = y_i$ based on constraint.

$$Propc(Y_i = y_i) = \{Y_j = y_j | Y_i = y_i \underset{Constraint}{\rightarrow} Y_j = y_j\} \tag{3.6}$$

Based on the propagated outcomes, the reward function $\varrho(Y_i = y_i, x)$ is defined as follows:

$$\varrho(Y_i = y_i, x) = \sum_{\substack{Y_j = y_j \in Prop_c(Y_i = y_i) \\ Y_j \in UL(x)}} \varrho(\hat{p_j}, Y_j = y_j, x) \tag{3.7}$$

(3) *Compute cross-task value of information for a sample-task pair*: With Eq.3.7, the cross-task utility for a sample-task pair is defined as follows.

$$UI(Y_i, x) = \sum_{y_i} \hat{p_i}(Y_i = y_i | x) \sum_{\substack{Y_j = y_j \in Prop_c(Y_i = y_i) \\ Y_j \in UL(x)}} \varrho(\hat{p_j}, Y_j = y_j, x) \tag{3.8}$$

With the above three steps, the algorithm selects the most informative sample-label pair which maximizes the UI value.

Qi et al. [92] proposed a two dimensional active learning algorithm for multi-label problems, which explores the uncertainty of sample and label correlation concurrently. The novel method requests the annotation on the sample-label pair, once added into the training set, is expected to minimize generalization error. In their paper, they derived a Multi-labeled Bayesian Error Bound for the sample-pair selection.

Given a sample x and its labeled and unlabeled parts $U(x)$ and $L(x)$. Once y_s is activated to ask for labeling, the Bayesian classification error $\varepsilon(y|y_s; y_L(x), x)$ for an unlabeled $y_i \in U(x)$ is bounded as:

$$\varepsilon(y|y_s; y_L(x), x) \leq \frac{1}{2m} \sum_{i=1}^{m} \{H(y_i|y_{L(x)}, x) - MI(y_i; y_s|y_{L(x)}, x)\} \tag{3.9}$$

Where

$$H(y_i; y_s|y_{L(x)}, x) = \sum_{t,r\in\{0,1\}} \{-P(y_i = t, y_s = r|y_{L(x)}, x) log P(y_i = t, y_s = r|y_{L(x)})\} \tag{3.10}$$

denotes the entropy of the sample-label pair, and $MI(X;Y) = H(X) - H(X|Y)$ denotes the mutual information between y_i and y_s given unlabeled part $U(x)$. Accordingly, the algorithm selects the most informative sample-label pairs, which are expected to reduce the Bayesian classification error over the unlabeled pool to the greatest extent. Before selecting a sample-label pair (x_s, y_s), the expected Bayesian classification error is denoted by

$$\epsilon^b(P) = \frac{1}{|P|} \sum_{x\in P} \epsilon(y|y_{L(x)}, x) \tag{3.11}$$

After the pair is selected, the expected error is calculated as follows

$$\epsilon^a(P) = \frac{1}{|P|} \{\epsilon(y|y_s; y_{L(x)}, x_s) + (\sum_{x\in P-x_s} \epsilon(y|y_{L(x)}, x))\} \tag{3.12}$$

Therefore, the goal of the algorithm is to select a best (x_s^*, y_s^*), which maximizes the error reduction $\triangle\varepsilon(P)$, that is

$$\begin{aligned} (x_s^\star, y_s^\star) &= \arg \max_{x_s\in P, y_s\in U(x_s)} \triangle\varepsilon(P) \\ &= \arg \min_{x_s\in P} -\triangle\varepsilon(P) \end{aligned} \tag{3.13}$$

Where $\triangle\varepsilon(P) = \varepsilon^b(P) - \varepsilon^a(P)$. Applying Eq. (3.10) to Eq. (3.13), we have

$$(x_s^\star, y_s^\star) = \arg \max_{x_s\in P, y_s\in U(x_s)} MI(y_i; y_s|y_{L(x_s)}, x_s) \tag{3.14}$$

Consequently, the method selects the sample-label pair for labeling according to Eq. (3.14), by maximizing the mutual information at each loop.

3.4.1.3 Exploration on Feature and Label Correlation

Some algorithms exploit both feature and label correlations simultaneously. In addition to comparison on the feature similarity, this scheme further explores the neighbor's prediction information for a specific instance. [45] designs a hidden multi-class representation to capture intra-class variability for object detection for an image, i.e., a binary classification task discriminating foreground from background. In their design, they applied a classifier-based bootstrapping with online multi-class classifier and generated virtual classes, which are separated into negative and positive classes. Based on these label correlations, each modality decides whether to generate a new virtual class. Then we utilize clustering to find a context background, which explores sample correlation from the view of features. The main process is as follows: In the first step, it trains an initial classifier to discriminate object from background, and then applies it to the current scene. For each sample, misclassified by the model or close to the decision boundary, a new virtual class is added to the multi-class models. In this model, Gradient Boost algorithm is used to combine a number of selector ξ_i, to a strong one

$$\Xi(x) = \sum_{i=1}^{\imath} \xi_i(x) \tag{3.15}$$

Each selector ξ_i is formed by a number of classifiers $\{\xi_{i,1}(x), \cdots, \xi_{i,j}(x), \cdots, \xi_{i,N}(x)\}$, and $\xi_i(x)$ is represented by its best classifier which minimizes the generalization error. For each $\xi_{i,j}(x)$, online histogram is used to evaluate their confidence on prediction. For example, one can use symmetric multiple logistic transformation in Eq.(3.16), where p_j, the prediction confidence of x belonging to class j, can be calculated in the online histogram. Since the model is built on the context of scene, it can also handle changing context.

$$\xi_j(x) = logp_j(x) - \frac{1}{\Im} \sum_{k=1}^{\Im} logp_k(x) \tag{3.16}$$

3.4.1.4 Exploration on Structural Correlation

A graph is a good data structure to present instance correlation in a data set. Given a graph $g = (V, \mathfrak{E})$, each node v_i denotes an instance x_i, which is denoted by a vector

as introduced in Definition 1. Each edge $\mathfrak{e}_{ij} =< v_i, v_j >$ describes some relationship between two nodes v_i and v_j. Take web page link as an example, the web page is likely to have similar topics with its link pages, therefore, each web page is denoted by a node, and the link relationship between the web pages is denoted by the edge. Consequently, we can explore instances correlation from the graph structure. When the label of a node is annotated, the labels of its neighbors can also be inferred, which reduces labeling cost.

In this graph structure setting, collective classification is a key method used for predicting labels of nodes in the graph simultaneously. Generally, the label y_i of a node x_i depends on its own features as well as the labels y_j and features of other nodes x_j.

Various collective classification methods have been proposed with regard to the structural correlations. Bilgic , Mihalkova and Getoor [11] proposed a novel active learning for network instances. They constructed a local collective classification model on a complex object, which is a vector including its local features x_i , an aggregation of features and labels of its neighbors $aggr(N_i)$. The collective classifier CC is used to learn $P(y_i|x_i, aggr(N_i))$, and a content-only classifier CO based on local node information is used to learn $P(y_i|x_i)$. After clustering the nodes, the algorithm calculates the disagreement score of each cluster C_j ,which is defined as

$$Disagreement(CC, CO, C_j, D^L) = \sum_{V_i \in C_j \cap D^U} LD(CC, CO, V_i, D^L) \tag{3.17}$$

Where $LD(CC, CO, V_i, D^L)$ is the entropy value of node V_i's labels over the output spaces.

After computing the disagreement score of each cluster, one ranks them according to their disagreement score, selects the first k clusters with the largest score, and randomly samples an item in each cluster for labeling.

Notice that when a given collective classification misclassifies a node, an island of nodes is likely to be misclassified. To address this issue, Bilgic, and Getoor [10] added a "reflect and correct "scheme under a collective classification model based on their previous work. They developed an active inference for collective classification, with general process illustrating in Fig. 3.3. The method constructs a traditional collective model on the graph, and then makes predictions on the test instances. To

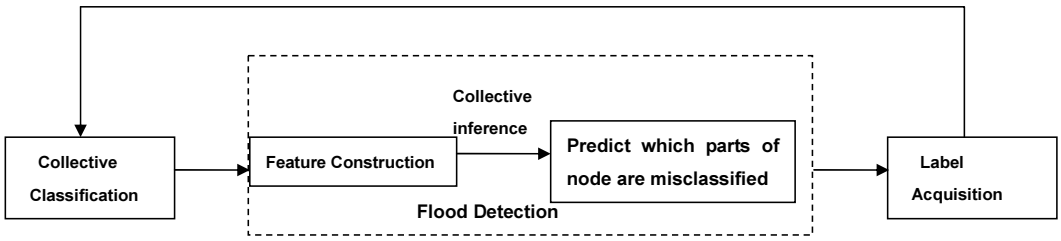


Figure 3.3: Active inference using Reflect and Correct Method. We iteratively label the nodes
with a collective model. To predict node mis-classification, we build a classifier on the features containing node content and its neighbor information, and then predict possible mis-classified instances with the classifier, tagging their mis-classified labels as True. After that, they are corrected with RAC strategy.

find out whether a node is misclassified or not, it constructs a feature table that is a possible indicator for judging whether a node is misclassified, and builds a classifier on the features to predict the possible misclassified ones. After that, it acquires a label for the central node among the potentially misclassified ones. The process repeats until the system's requirements are satisfied. This iterative process is called reflect and correct process.

3.4.2 How to Evaluate Selected Unlabeled Instances

After exploiting instances correlation from four different views, we utilize the same framework introduced in Section 3.3.2 to study the evaluation of selected unlabeled instances.

Indeed, there are some extra model construction methods for instance correlation based active learning algorithms. For Query by Single model, the application has been expanded into multi-label tasks. Suppose given a labeled data set D^L , each instance e_i is an multi-label instance as denoted in Definition 2, which is denoted as $x_i = \{< x_{i1}, x_{i2}, \ldots, x_{iq} >; < y_{i1}, \ldots, y_{il} >\}$; and label constraints rules $\mathcal{C}$. However, an instance may have incomplete label information, that is, only a portion of labels $\mathbb{Y}^L$. With such incomplete information, an accurate model still can be built by taking advantage of label constraints. To this end, a new model based on feature information and label constraints is developed for multi-label prediction tasks. A model can be constructed according to feature information $\mathcal{F}$ and known label information $\mathbb{Y}^L$, because the unknown labels $\mathbb{Y}^U$ can be inferred with constraint

rules $\mathcal{C}$, which effectively reduces the labeling cost. Therefore, the target function is represented as follows:

$$\hbar(.) : (\mathcal{F}, \mathbb{Y}^L) \underset{\mathcal{C}}{\rightarrow} \mathbb{Y}^U \tag{3.18}$$

While for the Query by Committee model, algorithms exploring graph structure correlations have a different committee construction method. They build two classifiers to form a classifier committee, including a collective classifier and a context-only classifier. The former predicts an instance class label according to its own feature information, as well as its neighbor's feature and labels information; whereas the latter is built based on its own information. Query by Committee favors the instance maximizing the disagreement between the two classifiers. Compared with the committee used in the active learning based on IID information, the committee member consists of different kinds of classifiers rather than the same types of classifiers.

3.5 Algorithm Performance Comparison

In this section, we first summarize all the reviewed instance selection methods in Table 3.2, with respect to the two dimensions: "how to select "and "how to evaluate ". Then, we select some representative methods from the two major categories: active learning based on IID instance uncertainty and active learning based on instance correlations, to conduct an experimental study and compare their performance, as well as analyze their strengths and weakness.

3.5.1 Time Complexity Analysis

Most papers have shown that active learning gains improvements compared to passive learning. In the literature, since algorithms are tested and evaluated in different experimental settings, it is difficult to make a fair comparison across various active learning methods. In this subsection, we focus on the computational time complexity of some representative algorithms in each category introduced above. Because the time complexity for various algorithms relies on the component learner used in a method, which has a different computation complexity, we cannot make fair

Table 3.2: Summary of all reviewed instance-selection methods in terms of two dimensions: "how to select "and "how to evaluate ".

How to Select			How to Evaluate	
			One Model	Committee
IID Instance Uncertainty	Uncertainty Sampling [UNG] [67]	Least Confidence [UNGLC]	[134][70]	[127]
		Margin[UNGMA]	[18][26]	[18]
		Entropy[UNGEN]	[17][63] [78]	[132][55] [29]
	Expected Gradient Length [EGL]		[68][100]	[74]
	Variance Reduction[VAR]		[99][50] [53][58]	[80][119] [69][95] [142]
Instance Correlation	Exploiting Feature Correlation[EFC]	Cosine Similarity [EFCCS]	[68][102] [84]	[105]
		KL Divergence [EFCKL]	[7][61]	[81][131]
		Gausssian Similarity [EFCGS]	[136][139] [64]	
	Exploiting Label Correlation [ELC]		[23][24] [20]	[128][92]
	Exploiting Feature and Label Correlation [EFLC]		[45]	
	Exploiting Structure Correlation [ESC]			[11]

comparisons. In this subsection, we evaluate the time complexity of different algorithms based on the time cost for a query process over an unlabeled data set. We simply summarize the above query strategies in Table 3.2 from the dimensions of time complexity and instance correlation, as shown in Fig. 3.4. From Fig. 3.4, the time complexity and correlation values of the algorithms taking instance correlation into account are much higher than the ones based on IID information, which suggests algorithms taking correlation into consideration require more time to explore correlation information. For active learning based on IID information, we can easily conclude that Uncertainty Sampling strategies have the lowest time complexity, whereas Variance Reduction and Expected Gradient length algorithms have a higher time cost among the four strategies. The observations suggest that simple query strategy costs less time than complex strategies. For instance, Fisher algorithms need a K dimensional matrix in the calculation, whereas uncertainty sampling just uses the output distribution to evaluate instances uncertainty. The more details considered in the algorithm, the higher time complexity it requires. For Expected Gradient length scheme, its application on binary classification has the comparable performance with Uncertainty Sampling. However, for the multi-class prediction and sequence mining tasks, the time complexity becomes higher as the

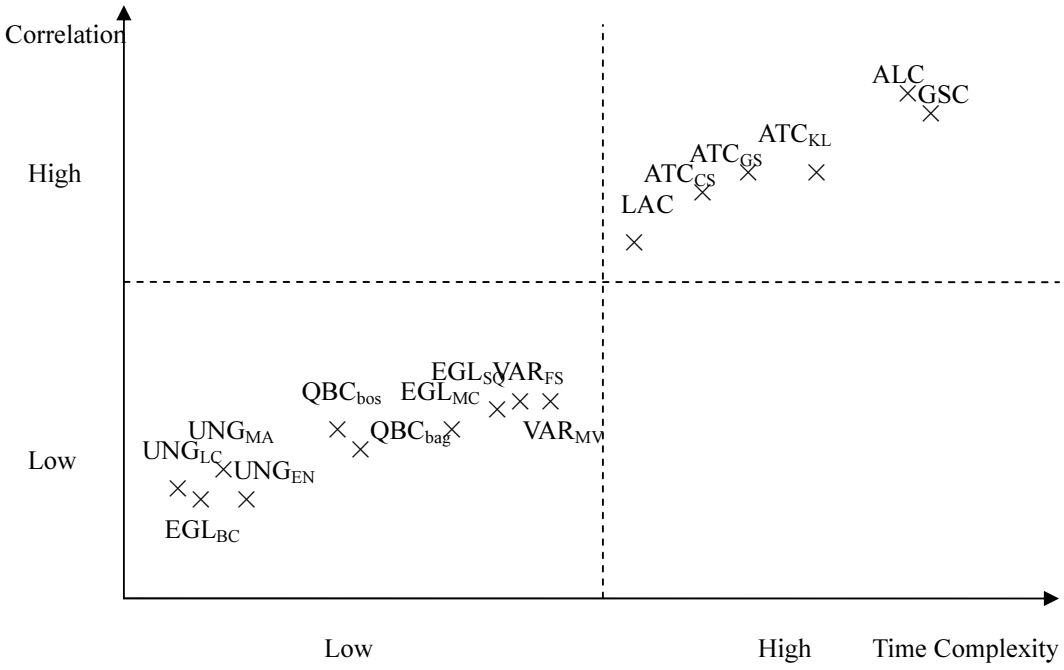


Figure 3.4: Query strategy comparison on representative algorithms from the instance correlation and time complexity perspectives. The x-axis denotes the time complexity and the y-axis denotes the instance correlations.

number of class labels or the length of sequence grows. The performance of Query By Committee is superior to EGL and VR, but is inferior to UNG, the main time cost depend on the component learner it chooses.

When taking instance correlation into consideration, the algorithms exploiting both feature and output information have almost the same time complexity as the ones exploiting correlation based on graph structure. The algorithms mining feature information with clustering algorithms have second highest time complexity. The above observations are consistent with our expectation. The more information explored, the more time the algorithm needs. For feature based algorithms, the method with KL divergence has a relative higher time cost than the other two similarity metrics. This is because KL divergence is an asymmetric metric, which costs more in computation cost. For output relation exploited algorithms, the time complexity relies on the number of class labels, which is much less than the number of data sets. Therefore, these kinds of algorithms have lower time complexity than the algorithms based feature information.

3.5.2 Label Complexity Analysis

One contribution of active learning is reducing labeling cost in terms of model construction. In this section, we analyze some sort of bounds on the number of queries required , which theoretically guarantee that this number is less than in the passive supervised setting.

According to theoretical analysis of the QBC algorithm by Freund et al.[123], to achieve the same generalization error ϵ, passive learning and active learning request to label $O(\frac{d}{\epsilon})$ and $O(d \log \frac{1}{\epsilon})$ instances respectively, where d is the Vapnik-Chervonenkis (VC) dimension [113] of the model space. This is an exponential improvement over the typical $O(\frac{d}{\epsilon})$ sample complexity of the supervised setting. However, this result can be tempered somewhat by the computational complexity of the QBC algorithm in certain practical situations, GiladBachrach et al. [44] offer some improvements by limiting the version space via kernel functions.

Dasgupta [32] also provides some theoretical lower bounds for active learning. For example, the labeling complexity of linear classifiers can grow to $O(\frac{1}{\epsilon})$ in the worst case, which offers no improvement over standard supervised learning, but is also no worse. The standard perceptron algorithm needs $O(\frac{1}{\epsilon^2})$ labels as a lower bound to learn a linear separator within the generalization error ϵ. To reduce the labeling complexity of perception classifiers, Dagupta et al. [31] present a variant of perceptron, which can achieve the same label complexity bound as reported for QBC.

Most of above theoretical results are under some limitations that a learner can perfectly classify the instances, and data are noise-free. To address these limitations, there has been some recent theoretical work in *agnostic active learning* [5], which only requires that unlabeled instances are drawn i.i.d. from a fixed distribution, and even noisy distributions are allowed. Hanneke [49] provides upper bounds on label complexity for the agnostic setting. Dasgupta et al. [33] propose a general agnostic active learning algorithm that works for any hypothesis class of bounded VC dimension, and any data distribution. These agnostic active learning approaches explicitly use complexity bounds to determine which hypotheses still "look viable", and queries can be assessed by how valuable they are in distinguishing among these viable hypotheses.

3.5.3 Lessons Learned

3.5.3.1 Lessons from IID Based Active Learning

IID based active learning employs an uncertainty evaluation measure to calculate instance utility values by treating instances as IID samples. All unlabeled instances are ranked based on their uncertainty values, and the subset with the largest uncertainty values are selected for labeling.

IID based approaches are commonly used in single-label learning tasks. The three representative subgroups are suitable for different types of applications.

- *Uncertainty sampling* [2, 39, 55] is often straightforward for probabilistic learning models. For example, *least confidence* has been popular with statistical sequence model in information extraction [2, 99]. In addition, *entropy*, a general uncertainty sampling measure, is appropriate to minimize the log-loss of the objective function[70, 134]. The other two uncertainty sampling measures, *least confidence and margin*, are more suitable for reducing model error [15, 18, 17, 117, 78], because they favor instances helping to discriminate specific classes.

- *Expected Gradient Length* is widely used in applications involving ranking functions [106, 68], such as information retrieval and text classification. Moreover, *Expected Gradient Length* strategy can be applied to discriminative probabilistic models by using gradient-based optimization, where the "change "of the model is evaluated by the length of the training gradient [74, 100].

- *Variance Reduction* can avoid model retraining process by taking advantage of Fisher Information Function: the information matrices simulate retraining process with an approximation of output variance [99, 53, 54]. The setting for variance reduction has been applied in the dual control problems as well. These approaches [58, 80, 119] either add a variance term or an innovation process, or consider it as a constraint to perform the active law selection process.

When taking individual instance uncertainty value into consideration, the selected instance subset may contain redundant knowledge and therefore cannot form an

ideal candidate set. In addition, for *Expected Gradient Length* and *Variance Reduction* based methods, there are some practical disadvantages in terms of computational complexity. For high dimensional feature spaces or large data sets, *Expected Gradient Length* is computationally expensive and its performance can deteriorate significantly if features are not appropriately scaled. In other words, the instance utility value calculated by expected gradient length can be over-estimated simply as a result of either one or multiple feature values or the corresponding parameter estimation is quite large, both resulting in a gradient of high magnitude. Meanwhile, the biggest challenge of *Variance Reduction* is its computational complexity. Each new instance requires a $\eta \times \eta$ matrix inversion for output variance estimation, where η is the number of model parameters, resulting in a time complexity of $O(n\eta^3)$ (n denoting the size of unlabeled instances). Therefore, for complex models involving a large number of parameters (η), the computational complexity of variance reduction based approaches can be very large. As a result, *Expected Gradient Length* and *Variance Reduction* are empirically much slower than simple uncertainty measuring strategy like *Uncertainty sampling*.

3.5.3.2 Lessons from Instance Correlation Based Active Learning

Comparing with *IID Based Active Learning*, instance correlation based active learning explores relationship between instances to calculate utility values of unlabeled samples. A utility metric is a combination of both an uncertainty function and a correlation function. Therefore, the selected candidate set balances the instance uncertainty and diversity for active learning. According to different correlation exploration views, existing solutions in this category are further be categorized into four groups: *Exploiting on feature correlation*, *Exploiting on label correlation*, *Exploiting on both feature and label correlation* and *Exploiting on structure correlation*.

Different from IID based active learning, which is mainly used for single-label learning tasks, instance correlation based active learning has been used for single-label [79, 25], multiple-label tasks [128], and for data with complex structures [45].

- *Exploiting on feature correlation* is the most common way to calculate instance correlations through feature based similarity measurements. Among all types

of similarity measures, *Cosine Similarity* is adequate for sequence classification tasks, such as text classfication [68] and name entity recognition [102]. Meanwhile, Cosine similarity is very effective for instances with high dimensional features, such as face recognition and text classification [84, 105], because it evaluates the similarity on the original input space without sub-space transition or matrix connection. *KL Divergence Similarity* is very unique and useful for evaluating the similarity tetween class distributions generated from different classifiers [7, 81]. *Gaussian Similarity* works well in semi-supervised learning frameworks [139, 136] and graph kennel function [64].

- *Exploration on Label Correlation* aims at solving multi-label learning and multi-task learning problems by exploring output constraints to improve the learning process [23, 24, 20], as well as to reduce the prediction cost.

- *Exploiting on both feature and label correlation* mainly handles data sets with multiple labels by considering feature and label correlations at the same time. For example, one can capture multi-class correlation to represent inter-class variability for visual object detection and tracking [45].

- *Exploiting on structure correlation* denotes instance correlation using a graph representation, assuming that an instance' neighbors share the same labels as the instance. Collective classification is a key method employed for predicting the labels of nodes in the graph simultaneously. This setting is applicable to networked data [11] and for active inference problems [10].

While instance correlation based active learning is effective to reduce redundancy in the selected candidate set, the computational cost for instance correlation calculation is expensive, especially for data sets with a large number of instances. For non-symmetric similarity measures, such as *KL Divergence Similarity*, the computation cost is twice high as the symmetric measures. The parameter settings, especially for *Gaussian Similarity*, can also be a big challenge. In addition, a common assumption in *Exploiting on structure correlation* is that an instance's neighbor nodes share the same labels as the instance. In reality, collecting enough labeled nodes are difficult (or impossible), so that prediction results mainly depend on the

network structures, which may reduce the prediction accuracy. Meanwhile, clustering is often the first step for collective classification, where the quality of the clustering results can bring a big impact to the final results and result in sampling redundancy in the selected candidate set.

3.6 Emerging Applications: Challenges and Trends

The methods reviewed in the previous sections are all in the standard active learning setting, in the senses that both labeled and unlabeled data sets are available before training, samples are assumed to be independent and identically distributed in the feature space, and labels are assumed to be provided by domain experts. However, in many emerging applications, these conditions can hardly be satisfied. Many challenges are posed for active learning in various complicated scenarios. The instance selection methods for active learning in these complicated scenarios are urgent to explore. In the following, we summarize a number of emerging active learning scenarios. For each scenario, we will analyze their challenges and discuss the research trends.

3.6.1 Active Learning on Streaming Data Platform

In many real-world applications, a large number of unlabeled instances arrive in a streaming manner, making it difficult (or even impossible) to maintain all the data as a candidate pool, such as email spam detection, malicious/ normal webpage classification[72]. This type of applications faces two issues. First, it generates diverse and massive data volumes in a short period of time, making it impractical for domain experts manually examining every datum. Second, the data stream evolves over time, therefore, the traditional training methods on a static data set may fail. A natural solution to tackle the two issues is to employ active learning by selecting a small amount of informative data for labeling to help build a model. However, traditional active learning does not fit for the dynamically changing candidate pool. Thus the challenges of active learning on streaming data platform is threefold: (1)

In the streaming data platform, the data volumes come continually, the candidate pool is dynamically changing, which also leads to the data distribution and decision boundary is evolving consecutively; whereas traditional active learning can deal with only static data sets. (2) Because of increasing data stream, storing all the data is very costly and impossible, whereas traditional active learning uses a candidate pool to store all the data in a data set. (3) In the data stream framework, because of the drifting/ evolving data volumes, building a model based on all the labeled data may be not reasonable, while, traditional active learning replies on a model build from all the previously labeled data.

To tackle these challenges, several algorithms have been proposed recently[40, 28, 9, 138, 88]. A Minimal Variance principle [142] is introduced to guide instance selection from data stream, coupled with a dynamic weight updating rule for data stream with drifting/evolving concepts. Following the same principle, Chu, Zinkevich, and Li [28] considered unbiased property in the sampling process in data streams, designed optimal instrumental distributions in the context of online active learning. Zhang etc.[127] presented a weighted ensemble classifiers and clusters model to mine concept drifting data streams. Most existing work on streaming data platform rely on building accurate ensemble models [9]. They share the same basic idea: using divided-and-conquer techniques to handle large volumes of data stream with concept drifting. Specifically, the data stream is partitioned into several small chunks, with each ensemble member model constructed from one chunk. The member models are eventually combined in different ways for prediction.

3.6.2 Active Learning with Complex Data Presentations

The massive data set collections of networked data in various domain applications (*e.g.*, social network, information network, and document citation) drive the research of flexible and accurate graph-based prediction models. Vertices classification in a graph is an important topic in graph-based models. A simple graph is designed based on a relation measure (*e.g.*, distance or similarity), where each node denotes a data element, and the edge denotes the relation between the corresponding pair of nodes. Given a graph with unlabeled vertices and a subset of labeled vertices, a model infers the unlabeled nodes memberships, on the strength of labeled

training set and graph structure. A common assumption which a model depends on for vertices classification is that similar data should be assigned the same class labels. Thus unlabeled vertices are given the labels of their nearest labeled neighbors in a simple way. However, in most cases, gathering sufficient labeled vertices is very expensive, so prediction results depend mainly on network structure, which may reduce performance. To address this issue, active learning focuses on reducing labeling cost by selecting an optimal utility vertices in a graph for the purpose of constructing a superior model.

In general, existing vertices selection criteria falls into two categories. The first type of approaches is to find an optimal solution for a designed objective function. For Instance, [47] proposes a function which seeks an optimal labeling vertices V^L that disconnect most regions of the graph by cutting minimal edges. However, there is no general algorithms for minimizing the function, and the method may perform worse than random algorithms in some experiments [47]. To address this issue, [21] employs active learning algorithm to find the minimization of the objective function on a spanning trees. Unfortunately, there is no experiments showing its effectiveness on a general graph. They conduct experiments with random spanning tree (RST) and breadth-first spanning tree (BST). RST may hide the cluster structure of graph, while, BST are likely affected by parameters like starting node.

Another kind of algorithms selects vertices corresponding to the disagreement between classifiers. In their design, they employ clustering algorithm to group vertices based on graph topology. Then they make predictions on each cluster by using a classifier community, and select samples with the most disagreement in each cluster to form an optimal subset. [11] effectively exploits the prediction difference between a classifier and a collective classifier, where the former is built with vertices information, while the latter also takes edges between vertices and neighbor's information into consideration. However, a fixed number of clusters are likely to destroy the actual data class distribution.

3.6.3 Active Learning with Crowdsourcing Labelers

Traditional active learning asks an omniscient expert to provide ground truths to the queried instances, so that labeled instances can help build an accurate model.

By doing so, the expert is assumed to be accurate (never wrong), indefatigable (always answers the queries), unique(only one oracle), and insensitive to costs(inexpensive/free annotation cost). However, labeling an optimal utility subset is still costly and expensive in many cases. To reduce labeling cost, crowdsourcing labelers, which are composed of some cheap and noisy labelers, have now been considered for active learning. Unfortunately, a direct application of crowdsourcing labelers on traditional active learning is problematic for two reasons. (1) Since 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. (2) Since active learning is consisted of multiple learning iterations, the errors induced in each round will be passed onto the following rounds and will be amplified. Thus, asking crowdsourcing labelers to directly provide noisy class labels may not be appropriate in active learning. The tradeoff between the labeling noise and labeling cost is a big challenge for active learning with crowdsourcing labelers.

To address the above challenges, existing work on active learning with crowdsourcing labelers mainly follow two directions. One research direction utilizes relabeling strategy to obviate the effect of noise. Follow this idea, Sheng et al. [103] proposed a crowdsourcing resolution in supervised learning scenarios. Based on Sheng et al.'s work [103], Zhao et al.[130] applied crowdsourcing labelers in active learning framework by incremental relabeling only the most import instances. Fu et al.[41] proposed a new active learning paradigm, in which a nonexpert labeler is only asked whether a pair of instances belong to the same class. To instantiate the proposed paradigm, it adopts the MinCut algorithm as the base classifier, and repeatedly updates the unlabeled edge weights on the max-flow paths in the graph. Finally, an unlabeled subset of nodes with the highest prediction confidence is added into labeled nodes.

Besides relabeling strategy, taking labeling cost into consideration is the other research direction. Integrating a cost budget into instance selection metrics, it guarantees that the selected optimal subset is subject to budget constraint, where budget is the total time cost avaliable on annotation. [115] formulates a budgeted selection task as a continuous optimization problem where the optimal selected subset maximizes the improvement to the classifier's objective, with a labeling cost

budget constraint. Proactive learning [37] focuses on selecting an optimal oracle as well as an optimal instance at the same time using a decision theoretic approach.

3.6.4 Active Learning for Domain Adaptations

It is desirable that a model built based on plenty labeled instances in one domain should perform reasonably well on data from different but similar domains [137]. For example, a classifier trained to classify "indoor" vs. "outdoor" images should be beneficial for training classifier to classify "building" vs. "natural scenery" images. A model on the source domain straightly applies in the target domain may result in a serious accuracy reduction. Therefore, we need to additionally label some instances in the target domain, so as to help leverage the original model apply in the target domain without performance compromise, which is known as domain adaptation. However, labeling a large amount of instances in the target domain is a costly and expensive process. So a promising resolution is seeking to minimize the amount of new annotation effort required to achieve good performance in the target domain. To reduce labeling cost, active learning can be employed to select instances to annotate from the target domain of interest.

Active learning in a domain adaptation setting has received little attention so far, where existing work mainly follows either pool based or online active learning settings for domain adaption. In the pool based active learning setting, [22] proposes to combine active learning with domain adaptation for word sense disambiguation system. [104] employs an initial pool of labeled target domain to help train an in-domain model. In the online active learning setting, [96] presents a novel approach that uses a domain-separator hypothesis in the active query process, and further leverages inter-domain information. Meanwhile, Zhu et. al. [144] proposed a transfer active learning approach which actively selects (and labels) samples from auxiliary domains to improve the learning for a target domain. Both approaches in [96, 144] can be used in pool-based or online active learning settings.

3.7 Summary

With the goal of labeling the most informative instances to achieve high prediction accuracies with minimum cost, active learning is a continuously growing area in machine learning research. In previous work, the emphasis has been on the design of new query strategies for instance selection criteria. In this section, we categorized existing query strategies in active learning into two groups: (1) Active learning based on IID instance uncertainty, and (2) Active learning based on instance correlations. We surveyed the two types of query strategies, analyzed and compared the time complexity of some representative methods, and briefly discussed some potential issues in the existing designs. A number of emerging active learning scenarios and new approaches are also discussed in this section. Our survey, which mainly emphasizes on instance selection, provides a high-level summarization for interested readers to take instance correlations into consideration for designing effective active learning solutions.

Chapter 4

Active Learning with Optimal Instance Subset Selection

4.1 Introduction

From a technical perspective, the key of AL is to find instances mostly needed for labeling, such that the inclusion of the instances into the labeled set can help improve learning model.

When assessing instances for labeling, existing active learning methods can be roughly categorized into two groups: 1) individual assessment based and 2) set assessment based. The former treats unlabeled instances as independent and identically distributed (I.I.D.) samples, and each instance's utility value is calculated without taking others into consideration, whereas the latter intends to select an optimal subset with a maximal utility value, by using sample correlations/distributions to estimate utility value.

One possible problem with individual-assessment-based approaches is that they may label similar instances, which, in turn, results in labeling redundancy. Take Fig. 4.1 as an example, when only considering the uncertainty of the samples for labeling, a labeling set may contain most uncertain samples, each of which has the maximum uncertainty value from a single-instance perspective, but samples in the set may contain redundant knowledge and do not form an ideal candidate set, as shown in Fig. 4.1(b). Utility metrics based on a set assessment, on the other hand, utilizes some similarity measures to discriminate samples so selected instances may not be the most uncertain ones, whereas together, they form a good labeling set. As shown in Fig. 4.1(c), by considering sample correlations, decision boundaries

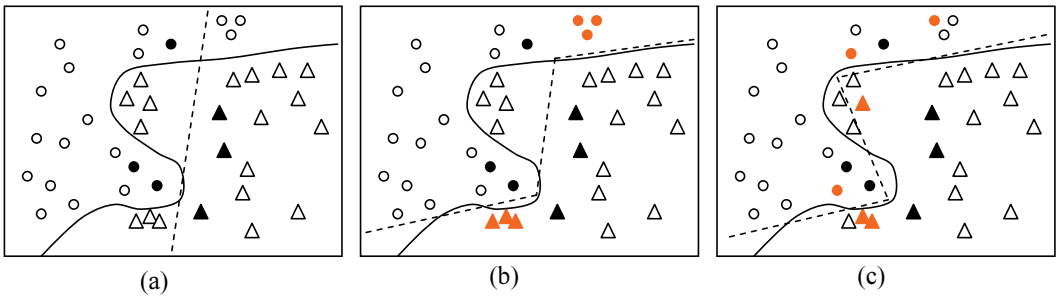


Figure 4.1: A toy example to demonstrate the tradeoff between uncertainty and diversity for sample selection in active learning. Circles and triangles denote the instances from two classes, respectively; solid circles and triangles denote labeled instances and the rest denote unlabeled instances. The solid lines denote the true decision boundaries and the dashed lines denote the decision boundaries learned by the learners based on the selected instances. (a) Decision boundary learned from six labeled training instances. (b) By labeling six most uncertain instances, the learner refines its decision boundary, which becomes more approximate to the true decision boundary. (c) By taking sample diversity into consideration, a method chooses the most informative candidate instances with low redundancy between them, based on which the learned decision boundary is significantly improved, compared to the approach which considers uncertainty only.

generated from six selected candidates are much closer to the genuine boundaries, compared to the approach in Fig. 4.1(b). Batch mode AL , represents the typical set-assessment-based methods.

For set assessment based active learning, although such methods do take sample correlations into consideration for active learning, in practice, they have at least the following two disadvantages:

- For clustering based methods [16, 122], they normally separate instances into groups, and select centroid of each cluster as representative instances for labeling. Accordingly, active learning crucially relies on clustering results whereas most clustering methods can only generate convex groups and real-world data can distribute arbitrarily. In addition, because instance selection depends on cluster numbers and sizes, and clustering is typically not a transparent process, these methods are inflexible for active learning.

- For batch mode active learning [54], instance selection is made by using a search process, *e.g.* hill-climbing search, to choose instance, one at a time, to maximize the objective function, and iterate several times to greedily include instance, one at a time, into the selected set to form a subset. Therefore,

these works, in essence, still are based on single instance selection without considering each subset as a whole for active learning.

In order to address the above issues, we propose a completely new active learning paradigm under the context of an oracle labeler by employing instance correlation for selecting unlabeled samples for labeling. To capture instance correlations, we explore the pairwise correlation from feature spaces and prediction distribution simultaneously. we combine instance uncertainty and instance correlations to form a matrix with each element denoting correlation of instances indexed by the corresponding row and column. Using correlation matrix, active learning can be regarded as an optimal subset selection problem to select b out of n samples, such that the selected subset has the maximum utility value.

The problem definition and system overview are introduced in Sec. 4.2. The algorithm details are introduced in Sec. 4.3, followed by experiments in Sec. 4.4.

4.2 Problem Definition & System Overview

4.2.1 Problem Definition

Given a dataset $\mathcal{D}$ consisting of n instances with each instance e_i denoted by $e_i = \{x_{i1}, \cdots, x_{im}; y_i\}$, where x_{ij} denotes the j^{th} attribute value of the instance and y_i denotes x_i's class label. If e_i is unlabeled, we denote it by $e_i = \{x_{i1}, \cdots, x_{im}; ?\}$. Assume at any stage, a labeled sample is moved from $\mathcal{D}$ to a subset $\mathcal{D}^L$, and the remaining unlabeled samples in $\mathcal{D}$ form an unlabeled subset $\mathcal{D}^U$, with $\mathcal{D} = \mathcal{D}^L \cup \mathcal{D}^U$ and $\mathcal{D}^L \cap \mathcal{D}^U = \emptyset$. The **aim** of optimal subset based active learning is to select and label a batch (*i.e.* a subset Δ) of instances, one batch at a time, from $\mathcal{D}^U$, such that when user requested number of instances are labeled, a classifier trained from $\mathcal{D}^L$ has the maximum prediction accuracy in classifying test samples D^T.

4.2.2 System Overview

In order to train classifiers from $\mathcal{D}^L$ with the maximum prediction accuracy, a commonly employed principle for active learning is to label samples with high uncertainty. Following this principle, assume a matrix $\mathcal{M}$ exists to capture each single

instance's uncertainty as well as the disparity between any two instances x_i and x_j, the above active learning **goal** can be regarded as the selection of an optimal subset of unlabeled samples Δ, such that the summation of instance uncertainty and disparity over Δ can reach the maximum (compared to any alternative subsets with the same size).

Accordingly, we employ an iterative procedure with the following three major steps:

1. Construct Classifier Ensemble. Using bootstrap sampling to train a committee of classifiers $E_\hbar$ from $\mathcal{D}^L$.

2. Build Correlation Matrix. Applying $E_\hbar$ to an unlabeled sample set $\mathcal{D}^U$ and building a correlation matrix $\mathcal{M}$ to capture instance uncertainty and disparity between instances.

3. Select an optimal subset with maximum utility value. Using correlation Matrix $\mathcal{M}$ to select an optimal subset Δ which has the maximum utility value among all candidate sets with the same size. This problem can be formulated as a quadratic integer programming problem as follows,

$$\begin{aligned} &\max_{\mathbf{x}} \mathbf{x}^T \mathcal{M} \mathbf{x} \\ s.t. \sum_{i, x_i \in \mathbf{X}} x_i = b \; ; \quad &x_i \in \{0, 1\} \end{aligned} \tag{4.1}$$

where $\mathbf{x}$ is an n-dimensional column vector and n is the size of unlabeled set $\mathcal{D}^U$. The constraint b defines the size of the subset for labeling, with $x_i = 1$ denoting that instance x_i is selected for labeling and $x_i = 0$ otherwise.

Assume the objective function in Eq.(4.1) is properly solved, the optimal subset Δ will contain b instances with the maximum summation of the uncertainty and disparity, through which the instance labeling redundancy is reduced. Depending on the number of instances for labeling, the above process repeats until the dataset is suitably labeled.

4.3 ALOSS: Optimal Instance Subset Selection for AL

We now discuss technical details on correlation matrix construction and optimal subset selection for active learning.

4.3.1 Correlation Matrix Construction

An important step of ALOSS is to construct a correlation matrix $\mathcal{M}$ with elements in $\mathcal{M}$ properly capturing each single instance's uncertainty as well as correlations between each instance pair. To build a correlation matrix $\mathcal{M} \in \mathbb{R}^{n \times n}$, where n denotes the number of instances in the unlabeled set $\mathcal{D}^U$, we separate elements in $\mathcal{M}$ into two parts. More specifically, assume $\mathcal{U}_{i,i}$ defines the uncertainty of instance e_i and $\mathcal{I}_{i,j}, i \neq j$ defines the disparity between instances e_i and e_j, the correlation matrix $\mathcal{M}$ is constructed using Eq.(4.2)

$$\mathcal{M}_{i,j} = \begin{cases} \mathcal{U}_{i,j}, & \text{if } i = j \\ \mathcal{I}_{i,j}, & \text{if } i \neq j \end{cases} \tag{4.2}$$

Classifier Weighting Matrix: To calculate instance uncertainty, we build a classifier ensemble $E_\hbar$ with m heterogeneous members, $\hbar_1, \cdots, \hbar_m$, each of which is trained from labeled sample set $\mathcal{D}^L$. Meanwhile, assume users intend to use a specific type of learning algorithm, say decision trees, to the final labeled samples and train a classifier to predict test samples, we also apply the same learning algorithm to $\mathcal{D}^L$ to train a benchmark classifier $\hbar^*$. Because we use ensemble $E_\hbar$ with diverse base learners, the purpose of employing $\hbar^*$ is to make sure that active learning process indeed favors samples with respect to user selected learning algorithm.

Given a classifier ensemble $E_\hbar = \{\hbar_1, \cdots, \hbar_m\}$ and a benchmark classifier $\hbar^*$, for each ensemble member $\hbar_j$, we compare its prediction with the benchmark classifier $\hbar^*$ on each unlabeled sample in $\mathcal{D}^U$ (which contains n instances), and generate an n by m matrix $H \in \mathbb{R}^{n \times m}$ as follows:

- $H_{i,j}$=1, if $\hbar_j$ and $\hbar^*$ has the same prediction on x_i.
- $H_{i,j}$=0, otherwise.

Table 4.1: A toy example demonstrating classifier matrix construction. (a) the prediction from a benchmark classifier $\hbar^*$ and predictions from the three classifiers of the Ensemble E on a four-instance dataset (b) the $H_{i,j}$ values for matrix H (c) The normalized matrix $\mathcal{H} = \frac{H^T \times H}{n}$

(a)

Classifier	x_1	x_2	x_3	x_4
$\hbar^*$	T	F	F	T
$\hbar_1$	F	T	F	T
$\hbar_2$	T	T	F	T
$\hbar_3$	F	T	F	F

(b)

Classifier	$\hbar_1$	$\hbar_2$	$\hbar_3$
x_1	0	1	0
x_2	0	0	0
x_3	1	1	1
x_4	1	1	0

(c)

Classifier	$\hbar_1$	$\hbar_2$	$\hbar_3$
$\hbar_1$	$\frac{2}{4}$	$\frac{2}{4}$	$\frac{1}{4}$
$\hbar_2$	$\frac{2}{4}$	$\frac{3}{4}$	$\frac{1}{4}$
$\hbar_3$	$\frac{1}{4}$	$\frac{1}{4}$	$\frac{1}{4}$

By using H matrix, which records agreements between each ensemble member and the benchmark classifier, we calculate a m by m matrix $\mathcal{H} \in \mathbb{R}^{m \times m}$ with $\mathcal{H} = \frac{H^T \times H}{n}$. Clearly, each diagonal term $\mathcal{H}_{j,j}$ denotes the percentage of same predictions between $\hbar_j$ and $\hbar^*$, whereas each off-diagonal term $\mathcal{H}_{i,j}, i \neq j$, denotes the common agreements between classifiers $\hbar_i$ and $\hbar_j$. Table 4.1 demonstrates the correlation matrix construction process using a toy example.

Instance Uncertainty: To calculate uncertainty for each single instance x_i in $\mathcal{D}^U$, we apply each ensemble classifier $\hbar_j, j = 1, \cdots, m$, to x_i, and build a vector $\mathbf{u}_i$ with each element $u_{i,j}, j = 1, \cdots, m$, recording uncertainty of classifier $\hbar_j$ on instance x_i (in our experiments, we use *entropy* to measure $\hbar_j$'s uncertainty on x_i). As a result, we can build an n by m matrix $\mathbf{u} \in \mathbb{R}^{n \times m}$, and calculate weighted instance uncertainty as follows.

$$\mathcal{U} = \mathbf{u} \times \mathcal{H} \times \mathbf{u}^T \tag{4.3}$$

According to Eq.(4.3), each diagonal term in $\mathcal{U}_{i,i}$ contains weighted entropy of x_i with respect to all ensemble classifiers in E, which are trained from labeled samples $\mathcal{D}^L$.

Pairwise Correlation: The purpose of calculating correlation between each pair of instances x_i and x_j is to capture difference between x_i and x_j such that an optimal subset can contain instances with high uncertainty and high disparity (so there is low redundancy in the labeled samples). To achieve the goal, we employ two

types of distance measures, prediction distance and feature distance, for correlation assessment.

Prediction Distance **(d_P)** intends to compare prediction dissimilarity of a same set of classifiers on two instances. The purpose is to assess the behavioral difference between a pair of instances (x_i *vs.* x_j) with respect to some classifiers. Given an instance x_i and a classifier $\hbar_\kappa$, assume the labeling space has $|\mathbb{Y}|$ labels in total, so x_i can be predicted, by $\hbar_\kappa$, as any label $y_l \in [1, |\mathbb{Y}|]$. Denote $P_\hbar(y_l|x)$ the probability of x_i belonging class y_l, as per classifier $\hbar_\kappa$'s prediction. For a pair of instance x_i and x_j, their prediction difference with respect to a classifier $\hbar_\kappa$ is denoted by

$$\Phi_{i,j}^{\hbar_\kappa} = (|P_\hbar(y_1|x_i) - P_\hbar(y_1|x_j)|, \cdots, |P_\hbar(y_{|\mathbb{Y}|}|x_i) - P_\hbar(y_{|\mathbb{Y}|}|x_j)|) \tag{4.4}$$

When combining prediction distance over all class labels $y_l \in [1, |\mathbb{Y}|]$ and all classifiers $\hbar_\kappa, \kappa = 1, \cdots, m$, we have

$$d_{P_{i,j}} = \sum_{\kappa=1, \hbar_\kappa \in E}^{m} \sum_{l=1}^{\mathcal{Y}} |\hbar_\kappa^l(x_i) - \hbar_\kappa^l(x_j)| \times \mathcal{H}_{\kappa,\kappa} \tag{4.5}$$

where $\mathcal{H}_{\kappa,\kappa}$ denotes the weight of the classifier $\hbar_\kappa$, as we discussed in Sec. 4.3.1.

Feature Distance **($d_\mathcal{F}$)**, as its name suggests, intends to capture the disparity of a pair of instances in the feature space. Given instance $e_i = \{x_{i,1}, \cdots, x_{i,m}; y_i\}$ where $x_{i,\kappa}$ denotes the κ^{th} feature value of x_i, the feature distance between x_i and x_j is calculated as follows,

$$d_{\mathcal{F}_{i,j}} = \sqrt{\sum_{\kappa=1}^{m} (x_{i,\kappa} - x_{j,\kappa})^2} \tag{4.6}$$

Pairwise Correlation **($\mathcal{I}$):** Because prediction distance ($\mathcal{P}$) and feature distance ($\mathcal{F}$) each denotes the difference between instances x_i vs. x_j from different perspectives, the final pairwise correlation between x_i and x_j is the product of the two distances as follows.

$$\mathcal{I}_{i,j} = d_{P_{i,j}} \times d_{\mathcal{F}_{i,j}} \tag{4.7}$$

By using the product of the prediction distance and feature distance to calculate the disparity between instances, as shown in Eq.(4.7), our intention is to simultaneously consider instances' behaviors (prediction distance) and their distance in

feature space (feature distance). Assume prediction distance and feature distance each assess instance distribution from one dimension, the product therefore assess the join distribution from both dimensions and is a proper way of assessing instance disparity.

4.3.2 Optimal Subset Selection

Given a correlation matrix $\mathcal{M}$ with n instances, the purpose of optimal subset selection, as defined in the objective function Eq.(4.1), is to select a subset with b instances, such that the summation of all instances' uncertainty and their disparities is the maximum among all alternative subsets with the same size. This problem is actually a standard 0-1 optimization problem, which is NP hard in general. Semi-Definite Programming (SDP) [46], fortunately, provides an approximate solution to solve similar NP-hard maximization problems with polynomial complexity. Accordingly, we transform the original problem in Eq.(4.1) to a "Max cut with size b "(MC-b) problem [46, 129], whose objective is to partition an edge-weighted graph (which contains N vertices) into two subsets, with one of which containing b vertices, such that the total weight of edges cross the cut (*i.e.* partitioning) is maximized. A formal definition of the MC-b problem is given in Eq.(4.8), where N denotes the number of vertices in the graph, and $w_{i,j}$ is the edge weight between vertices i and j.

$$\begin{aligned} \max_{y} \frac{1}{2} \sum_{i\in[1,N],j\in[1,N],i<j}^{N} w_{i,j} \times (1 - y_i y_j) \\ s.t. \sum_{i} y_i = N - 2b; \quad y_i \in \{-1, 1\}. \end{aligned} \tag{4.8}$$

To transform original problem, defined in Eq.(4.1), into the MC-b problem as defined in Eq.(4.8), we transform variable x_i in Eq.(4.1) as follows,

$$x_i = \frac{y_i + 1}{2}, \tag{4.9}$$

where $y_i \in \{-1, 1\}$. Replacing x_i in Eq.(4.1) using its form in Eq.(4.9), we have,

$$\begin{aligned} \max_{\mathbf{y}} \frac{1}{4}(\mathbf{y} + \mathbf{e})^T \mathcal{M}(\mathbf{y} + \mathbf{e}), \\ s.t. \ (\mathbf{y} + \mathbf{e})^T I(\mathbf{y} + \mathbf{e}) = 4b; \quad y_i \in \{-1, 1\}. \end{aligned} \tag{4.10}$$

where $\mathbf{y}$ is an n-dimensional vector with values either 1 or -1, and $\mathbf{e}$ is the same size column vector with all 1s. The original cardinality constraint $\sum_i x_i = b$ is rewritten as a quadratic form $\mathbf{x}^T I\mathbf{x} = b$, where I is an identify matrix.

To put transformed objective function in Eq.(4.10) and its cardinality constraints into quadratic form, we expand the vector $\mathbf{y} = (y_1, \cdots, y_n)$ into an extended form $\mathbf{y} = (y_0, y_1, \cdots, y_n)$ with $y_0 = 1$, and construct a new matrix $\mathcal{Q} \in \mathbb{R}^{(n+1)\times(n+1)}$ as follows,

$$\mathcal{Q} = \begin{pmatrix} \mathbf{e}^T\mathcal{M}\mathbf{e} & \mathbf{e}^T\mathcal{M} \\ \mathcal{M}\mathbf{e} & \mathcal{M} \end{pmatrix} \tag{4.11}$$

Similarly, we can apply same extension to the cardinality constraints and build a new constraint matrix $\mathcal{C} \in \mathbb{R}^{(n+1)\times(n+1)}$ as follows,

$$\mathcal{C} = \begin{pmatrix} n & e^T \\ e & I \end{pmatrix} \tag{4.12}$$

As a result, the original instance selection problem in Eq.(6.3) is transformed into an MC-b problem as follows,

$$\begin{aligned} \max_{\mathbf{y}} \mathbf{y}^T\mathcal{Q}\mathbf{y} \\ s.t.\ \ \mathbf{y}^T\mathcal{C}\mathbf{y} = 4b \\ y_0 = 1;\ y_i \in \{-1, 1\}; \forall\ i \neq 0 \end{aligned} \tag{4.13}$$

Solving MC-b using SDP programming: To solve Eq.(4.13), we denote $Y = \mathbf{y} \times \mathbf{y}^T$, where $Y \in \mathbb{R}^{(n+1)\times(n+1)}$ and have a SDP form for Eq.(4.13) as follows,

$$\begin{aligned} \max_{\mathbf{y}} \mathcal{Q} \bullet Y \\ s.t.\ \ \mathcal{C} \bullet Y = 4b \\ y_0 = 1; y_i \in \{-1, 1\};\ \forall\ i \neq 0 \end{aligned} \tag{4.14}$$

In Eq.(4.14), $\bullet$ defines dot product given as $A \bullet B = \sum_{i,j} A_{i,j} \times B_{i,j}$. We integrate the constraints on binary variable y_i. Because y_i has only two values 1 and -1, together with the constraint y_0=1, the diagonal terms in Y are all 1s. Consequently, the constraints $y_i \in \{-1, 1\}$ can be expressed as Diag(Y)=I, where I is an (n+1)-dimensional identity matrix.

Therefore the SDP relaxation of Eq.(4.13) is denoted by

$$\begin{aligned} &\max_{\mathbf{y}} \mathcal{Q} \bullet Y \\ s.t.\ \ &\mathcal{C} \bullet Y = 4k \\ &Diag(Y) = I;\ \ Y \succeq 0 \end{aligned} \tag{4.15}$$

where $Y \succeq 0$ defines that symmetric matrix Y is positive semidefinite (*i.e.* all its eigenvalues are nonnegative). Following SDP problem formulation defined in Eq.(4.15), one can employ publicly available open source packages to solve Eq.(4.15). In our experiments, we use SDPA [19], which is based on interior point method, to find solutions for Eq.(4.15).

4.3.3 ALOSS: System Framework

The whole process of ALOSS, as listed in Algorithm 2, is an iterative procedure. In each iteration, a small optimal subset Δ with b samples are selected for labeling. The labeled samples are then used to update models, including an ensemble E and a benchmark classifier $\hbar^*$, and help select another optimal subset Δ for labeling. The whole iterative process continues until the number of labeled instance *labeledsample* reaches the users defined value t.

Accelerated Process: In Algorithm 2, optimal subset selection is carried out on $\mathcal{M}$ which is a square matrix and its size is determined by the size of $\mathcal{D}^U$. For a large size matrix, finding solutions for SDP is computationally expensive. Alternatively, one can build a small matrix by removing samples whose uncertainty values are hopelessly small. So the instance subset selection only works on the remaining samples, which will, in turn, significantly reduce ALOSS's computational complexity. To achieve the goal, the key issue is to determine a proper threshold value and separate samples, according to their uncertainty values $\mathcal{H}$, into to subsets. For this purpose, we employ a histogram [87] based automatic thresholding method to divide unlabeled samples into two groups: low uncertainty group and high uncertainty group, with the goal of separating samples into two groups such that their combined spread (intra-group variance) is minimal.

Given an unlabeled instance subset $\mathcal{D}^U$ with n instances, the uncertainty of

each individual instance is denoted by $\mathcal{U}_{i,i}, i = 1, \cdots, n$, as given in Eq.(4.3). Assuming that the uncertainty values of all instances $\mathcal{D}^U$ are distributed between $[E_{min}, E_{max}]$, one can partition the range into L intervals with equal width, as defined by Eq.(4.16), and the number of samples whose entropies belonging to the jth interval is denoted by n_j.

$$\varpi = (E_{max} - E_{min})/L \tag{4.16}$$

Assume a threshold τ exists to separate samples in $\mathcal{D}^U$ into two groups $\mathcal{G}_{low}(\tau)$ vs. $\mathcal{G}_{high}(\tau)$, as follows

$$\begin{cases} \mathcal{G}_{low}(\tau) = \{x_i | x_i \in \mathcal{D}^U;\ \mathcal{U}_{i,i} \leq E_{min} + \tau \cdot \varpi\} \\ \mathcal{G}_{high}(\tau) = \{x_i | x_i \in \mathcal{D}^U;\ \mathcal{U}_{i,i} > E_{min} + \tau \cdot \varpi\} \end{cases} \tag{4.17}$$

Denoting the percentage of samples in the l^{th} interval by

$$P_l = \frac{\sum_{i=1;\ x_i \in \mathcal{D}^U;\ E_{min} + l \cdot \varpi < \mathcal{U}_{i,i} \leq E_{min} + (l+1) \cdot \varpi}^{n} 1}{n} \tag{4.18}$$

The respective percentage of samples in the groups $\mathcal{G}_{low}(\tau)$ and $\mathcal{G}_{high}(\tau)$, with respect to a given threshold τ, are then denoted by

$$\omega_{low}(\tau) = \sum_{j=0}^{\tau} P_j; \qquad \omega_{high}(\tau) = \sum_{j=\tau+1}^{L-1} P_j \tag{4.19}$$

The weighted mean for each of group $\mathcal{G}_{low}(\tau)$ and $\mathcal{G}_{high}(\tau)$, with respect to the threshold τ, is then defined by. Eqs.(4.20) and (4.21), respectively.

$$\mu_{low}(\tau) = \sum_{j=0}^{\tau} \frac{(j+1) \cdot P_j}{\omega_{low}(\tau)} \tag{4.20}$$

$$\mu_{high}(\tau) = \sum_{j=\tau+1}^{L-1} \frac{(j+1) \cdot P_j}{\omega_{high}(\tau)} \tag{4.21}$$

Assume the mean uncertainty level over the whole unlabeled data set $\mathcal{D}^U$ is calculated using Eq.(4.22)

$$\mu = \sum_{j=0}^{L-1} (j+1) \cdot P_j \tag{4.22}$$

The weighted inter-group variance, with respect to the given threshold τ, is then defined by

$$\sigma^2(\tau) = \omega_{low}(\tau) \cdot [\mu_{low}(\tau) - \mu]^2 + \omega_{high}(\tau) \cdot [\mu_{high}(\tau) - \mu]^2 \tag{4.23}$$

The main objective of the automatic thresholding method is to exhaustively search for an optimal threshold τ^* that maximizes in Eq.(4.23), which will, in turn, separate samples into two groups, $\mathcal{G}_{low}(\tau^*)$ and $\mathcal{G}_{high}(\tau^*)$, with maximum inter-group variance. The algorithm details for finding an automatic threshold to partition unlabeled samples $\mathcal{D}^U$ are shown in Algorithm 3.

Algorithm 2 ALOSS: Active Learning with Optimal Subset Selection

Input: (1) an unlabeled sample set $\mathcal{D}$; (2) m: # of classifiers to form an ensemble E; (3) t: # of samples selected for labeling; (4) $\hbar$: a learning algorithm for training final classifiers; and (5) b: the size of optimal subset (or batch size).

Output: an labeled sample set $\mathcal{D}^L$ with t labeled samples.

1: *labeledSample* $\leftarrow$ A small random number;
2: $\mathcal{D}^L \leftarrow$ Randomly label *labeledSample* instances from $\mathcal{D}$;
3: $\mathcal{D}^U \leftarrow \mathcal{D} \setminus \mathcal{D}^L$;
4: **while** *labeledSample* $< t$ **do**
5: $\quad \hbar^* \leftarrow$ Apply $\hbar$ to D^L to train a benchmark classifier;
6: $\quad E_\hbar\{\hbar_1, \cdots, \hbar_m\} \leftarrow$ Apply bootstrap samplings to D^L and build ensemble E with heterogeneous classifiers;
7: $\quad \mathcal{H} \leftarrow$ Apply E and $\hbar^*$ to $\mathcal{D}^U$ and build instance uncertainty matrix;
8: $\quad \mathcal{D}^{U'} \leftarrow$ Refining $\mathcal{D}^U$ using accelerated search process in Algorithm 2;
9: $\quad \mathcal{I} \leftarrow$ Calculate disparity matrix for instances in $\mathcal{D}^{U'}$;
10: $\quad \mathcal{M} \leftarrow$ Build instance correlation matrix;
11: $\quad \Delta \leftarrow$ Apply optimal subset selection to $\mathcal{M}$ and select a subset Δ with b instances;
12: $\quad D^L \leftarrow D^L \bigcup \Delta; \qquad D^U \leftarrow D^U \setminus \Delta;$
13: $\quad$ *labeledSample* $\leftarrow$ *labeledSample* $+ b$;
14: **end while**

4.3.4 Time Complexity Analysis

The total time complexity of ALOSS includes two major parts: 1) building instance-correlation matrix $\mathcal{M}$ and 2) applying SDP to $\mathcal{M}$ to select a b-instance subset. We assume that a learning algorithm with quadratic complexity $O(z^2)$ is used, where z is the maximum number of labeled instances. In each iteration, there are one benchmark classifier and m ensemble members that need to be trained. In addition, the calculation of the disparity matrix needs pairwise instance correlation, which requires $O(n^2)$time complexity, where n denotes the number of unlabeled instances. Therefore, in total, the time complexity for building instance-correlation matrix is

$$O(\mathcal{M}) = (m+1)O(z^2) + O(n^2) \tag{4.24}$$

Algorithm 3 AP: Accelerated Process using instance selection

Input: (1) weighted uncertainty values for all unlabeled instances in $\mathcal{D}^U, \mathcal{U}_{i,i},\ i = 1, \cdots, n$; (2) L: # of levels in separating instance uncertainty values $[0, L-1]$
Output: selected instance subset

1: $\varpi \leftarrow$ determining the step value for separating uncertainty values as shown in Eq.(4.16).
2: $\tau^* \leftarrow 0$; $\sigma(\tau^*) \leftarrow 0$; initializing optimal threshold and corresponding maximum variance value as 0.
3: **for** $j = 0$ to $L-1$ **do**
4: $\tau \leftarrow j$; setting current threshold.
5: $\mathcal{G}_{low}(\tau), \mathcal{G}_{high}(\tau) \leftarrow$ separating two groups as shown in Eq.(4.17).
6: $\sigma^2(\tau) \leftarrow$ calculating inter-group variance as shown in Eq.(4.23)
7: **if** $\sigma^2(\tau^*) < \sigma^2(\tau)$ **then**
8: $\tau^* \leftarrow \tau$; updating threshold value.
9: $\sigma^2(\tau^*) \leftarrow \sigma^2(\tau)$; updating optimal variance.
10: $\mathcal{G}_{high}(\tau^*) \leftarrow \mathcal{G}_{high}(\tau)$; updating optimal subset.
11: **end if**
12: **end for**
13: **return** $\mathcal{G}_{high}(\tau^*)$.

Due to the accelerated search process (Algorithm 3), we can reduce the correlation matrix from $\mathcal{M} \in R^{n \times n}$ to $\mathcal{M} \in R^{(\alpha \times n) \times (\alpha \times n)}$, where $\alpha \in [0, 1]$ is the percentage of reduction. The SDP process requires $O(SDP) = O(b^2 + (\alpha \times n)^3)$[124] complexity to solve the $n \times n$ matrix and selects a b-instance subset. Because the size of the instance subset b is much smaller than$\alpha \times n$, we can regard that SDP's time complexity is bounded by $O((\alpha \times n)^3)$.

Because the whole AL process requires the repetitive training of the classifiers, the while loop between steps 4 and 14 in Algorithm 2 needs to repeat z/b times, so the total time complexity of ALOSS is given as follows:

$$O(ALOSS) = \frac{z}{b}[(m+1)O(z^2) + O(n^2) + O((\alpha \times n)^3)] \tag{4.25}$$

The aforementioned complexity analysis indicates that the bottleneck time complexity of ALOSS is asymptotically bounded by the SDP process.

4.4 Experiments

We implement ALOSS and a number of baseline approaches using Java and WEKA data mining tools, and comparatively study their performance on 10 benchmark

datasets collected from UCI data repository [82]. A simple description of the benchmark data sets is summarized in Table 4.2. To comparatively study the algorithm performance, we compare classifiers trained from sample sets labeled different methods. If a classifier trained from a sample set labeled by A has a higher accuracy than the classifier trained from a sample set labeled by B, we conclude that A has a better active learning performance than B. To make fair comparisons, all methods are compared based on the same training and test samples (the initial randomly labeled samples are also the same for all methods). All experiments are based on 10 times 10-fold cross validation. To build ensemble $E_\hbar$ with diverse classifiers, we employ four learning algorithms, including (1) Decision Trees, (2) Naive Bayes, (3) ZeroR (a classifier predicting samples to the majority class), and (4) Multilayer Perception, to build an ensemble with $m = 8$ classifiers, with each of them contributing two classifiers.

Table 4.2: A simple description of the benchmark data

ID	Dataset	Instances	Features	Classes
1	horse	368	23	2
2	auto-mpg	398	8	3
3	balance	625	5	3
4	pima	768	9	2
5	vehicle	846	19	4
6	german	1000	21	2
7	cmc	1473	10	3
8	car	1728	7	4
9	segment	2310	19	7
10	abalone	3196	37	2

4.4.1 Benchmark Methods

In addition to the proposed ALOSS approach, we also implement a number of mainstream active learning methods [8].

Random is a simple approach, which randomly selects user requested number of instances for labeling.

Entropy is a query by uncertainty active learning method, which uses entropy as uncertainty measure. Each instance x_i's entropy is calculated by using class distributions predicted from a classifier, as defined by Eq.(4.26), where $P(y_l|x_i)$ is

the probability of x_i belonging to class y_l.

$$x_H^* = argmax_{x_i} - \sum_l P(y_l|x_i) \log P(y_l|x_i) \tag{4.26}$$

All unlabeled samples are sorted according to their uncertainty values. A number of samples ranked on the top of the list form a batch for labeling. The labeling process repeats until the user requested number of instances are labeled.

Margin is identical to Entropy except the uncertainty measure. Instead of using entropy, Margin aims to seek the difference between the two most likely class labels on a specific instance, as defined by Eq.(4.27)

$$x_H^* = argmin_x P(y_1|x) - P(y_2|x) \tag{4.27}$$

where y_1 and y_2 are the first two most probable class labels x being classified by a classifier. Intuitively, an instance with the least margin value is the one which is mostly ambiguous (*i.e.* uncertain). Similar to Entropy, Margin also employs a batch based iterative process for active learning.

SIB is a Single Instance Batch method which is identical to Entropy except that for SIB the batch size is 1, which means that SIB selects and labels instance one at a time. The main purpose of using SIB as a baseline method is to check that if we remove the redundancy during the batch based active learning process, what is the best performance the existing method can possibly achieve. Because SIB has the smallest batch size, it is the most computationally expensive method.

IW is a most recently developed instance weighting based method [8]. For each unlabeled data x_t, IW generates a weight value for x_t according to its maximum loss value on a set of benchmark classifiers, as defined in Eq.(4.28), where $\mathcal{L}()$ defines a loss function, f and g each denotes a classifier in a set $\hbar$. The higher the p_{x_t}, the more likely x_t is going to be selected for labeling.

$$p_{x_t} = max_{f,g \in \hbar; y \in Y} (\mathcal{L}(f(x_t), y) - \mathcal{L}(g(x_t), y)) \tag{4.28}$$

ALOSS$_p$ and **ALOSS**$_f$ are variants of the proposed ALOSS approach, where instance disparity $\mathcal{I}$ only considers prediction distance (ALOSS$_p$) or feature distance (ALOSS$_f$), respectively, as introduced in Section 4.3.1.

For the purpose of fairness of comparison, all the baseline algorithms are designed as " set assessment" mode by selecting top b data with the largest utility at a time, except "SIB", labeling one data at each time.

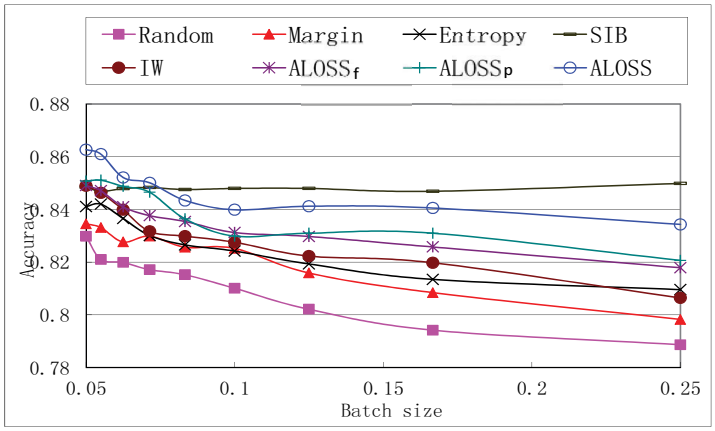
In the following sections, we first study algorithm performance in different parameter settings, including different batch sizes, different labeling portions, etc., and then report their performance on all benchmark datasets.

4.4.2 Experimental Results with Different Batch Sizes

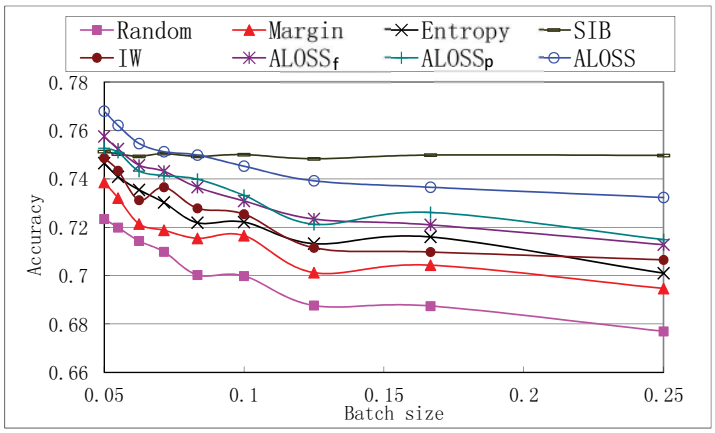
In Fig. 4.2, we report the performance of different algorithms on three datasets, where x-axis shows the batch size and y-axis reports the accuracies of the classifiers trained from final labeled samples (by using different active learning methods). In our experiments, we use decision trees (using J4.8 implementation) as the benchmark learner, and active learning is used to label 50% of samples. For example, for batch size 0.05 (which means 5% of samples), an active learning algorithm needs to repeat 10 times in order to label 50% of samples (In our experiments, we vary number of labeling iterations from 10 to 2 with step -1 to label 50% of samples, which correspond to batch sizes, 0.5/10=0.05, 0.5/9=0.056, ..., 0.5/2=0.25).

As the batch size increases, the performance of all methods, except SIB, deteriorates (the batch size of SIB is fixed to 1 so its performance remains stable across all batch sizes). This is because the labeled samples is fixed to 50% and for a smaller batch size, an active learner will have more iterations to update its instance selection process. Interestingly, the results in Fig. 4.2 also show that as the batch size increases the performance gain of ALOSS, in comparison with other methods, continuously improves. This asserts that the optimal subset selection procedure in ALOSS does play an effective role for avoiding redundancy and selecting informative samples for labeling. For small batch sizes, the effect of redundant samples may be marginal as if a batch only contains several samples the redundancy among them may not be significant and a learning algorithm may also need the redundancy to build correct decision logics. For large batch sizes, simply sorting all samples according to their uncertainty, without considering their correlations, will introduce significant redundancy in the labeled samples.

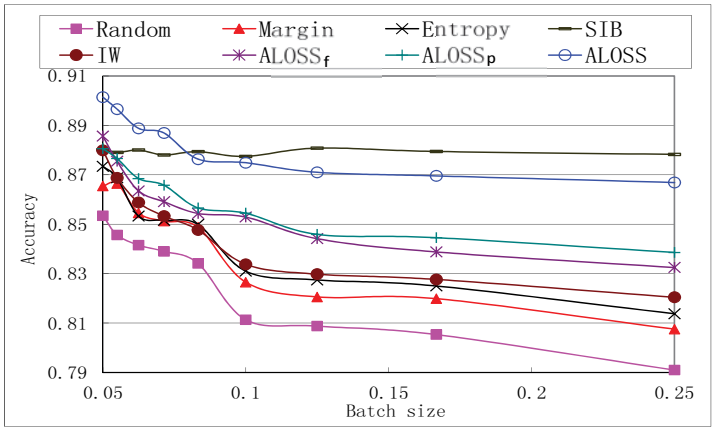
To demonstrate algorithm performance in most rigorous conditions, in following experiments we use batch size 0.05 for all experiments.



(a) Horse (2 classes)



(b) Auto-mpg (3 classes)



(c) Car (4 classes)

Figure 4.2: Accuracy comparisons with different batch sizes for active learning

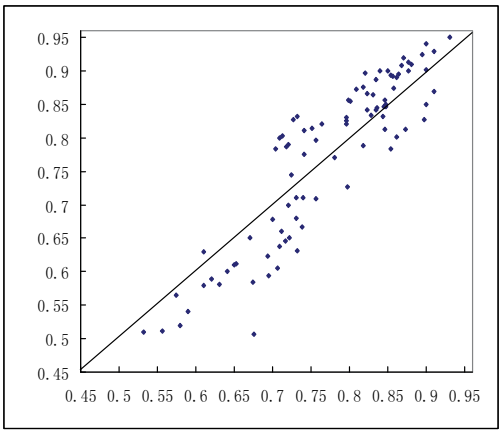
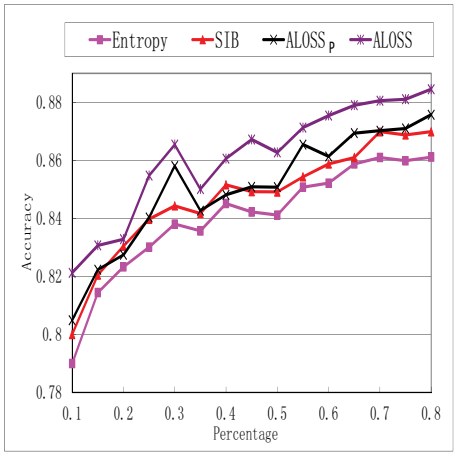
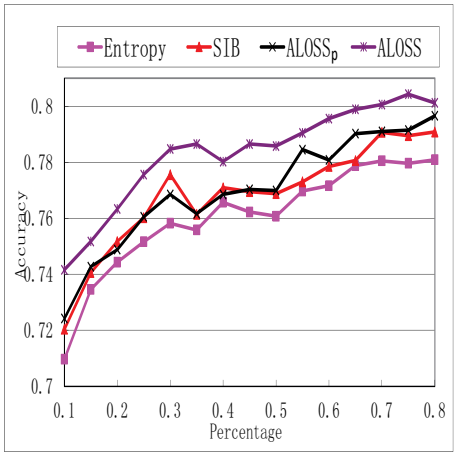


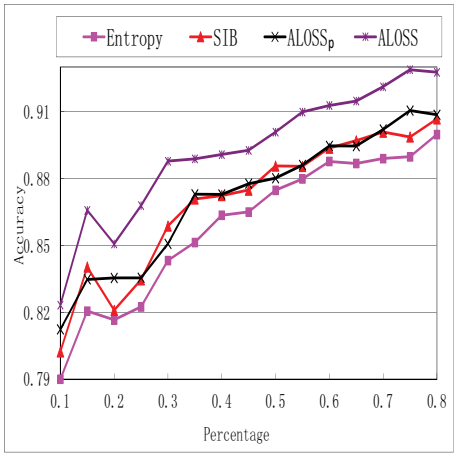
Figure 4.3: Head-to-Head comparisons between prediction distance and feature distance based instance disparity measures



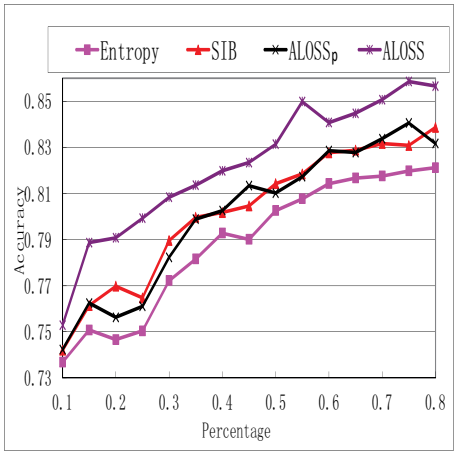
(a) Horse (Decision Trees)



(b) Horse (Naive Bayes)

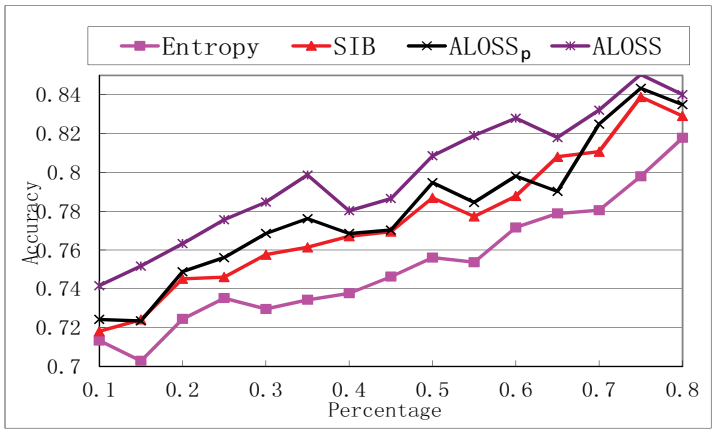


(c) Car (Decision Trees)

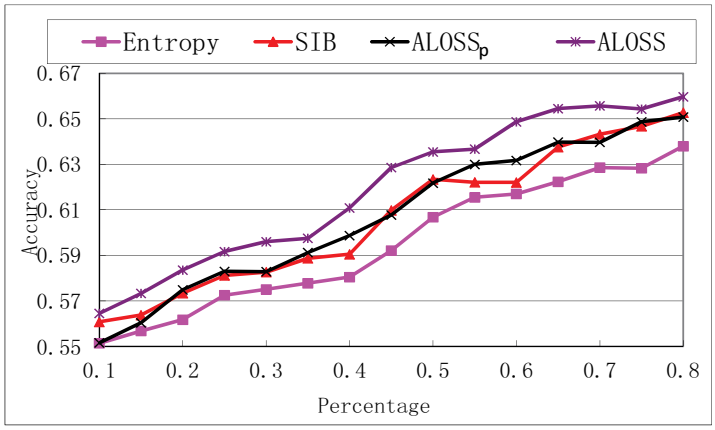


(d) Car (Naive Bayes)

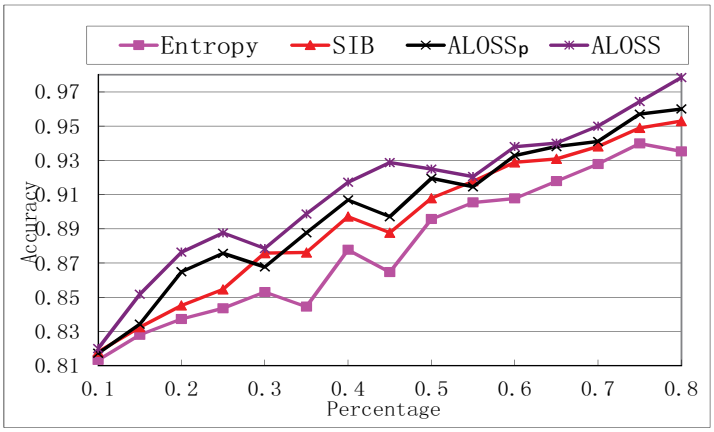
Figure 4.4: Accuracy comparisons *w.r.t.* different portions of labeled samples



(a) Pima (2 classes)



(b) Auto-mpg (3 classes)



(c) Segment (7 classes)

Figure 4.5: Accuracy comparisons *w.r.t.* different portions of labeled samples, the batch size is fixed to 0.05

4.4.3 Comparisons between Different Distance Measures

In Fig. 4.3, we compare the performance of ALOSS$_p$ and ALOSS$_f$ on all benchmark datasets. In our experiments, we fix the batch size to 0.05, and the initial randomly labeled samples is set as 0.05. Then we apply ALOSS$_p$ and ALOSS$_f$ to label different percentages of samples, with the labeling percentages varying from 10% to 50%. Because batch size is 0.05, for each dataset, it will generate 9 results, which correspond to the labeling percentages 10%, 15%, 20%,..., 50%. For 10 benchmark datasets, we will have 90 pairs of accuracy values in total. We report all 90 pair accuracies in Fig. 4.3, where a value above $y = x$ line indicates that ALOSS$_f$ outperforms ALOSS$_p$, and vice versa.

Among all 90 observations, ALOSS$_p$ outperforms ALOSS$_f$ on 55 cases, which asserts that instead of considering feature values to assess instance correlations, like existing correlation-based active learning algorithms do [85], using behaviors of instances with respect to different classifiers is an effective way to assess instance correlations. Interestingly, for classifiers with relatively high accuracies, ALOSS$_f$ appears to has a better chance to outperform ALOSS$_p$. Indeed, because ALOSS$_f$ replies on Euclidean distance, which is essentially a nearest neighborhood approach, to assess sample correlations. When classifier accuracy is low, it means that using feature based distance function is ineffective to differentiate samples from different classes.

4.4.4 Results with Different Percent of Labeled Samples

In Fig. 4.4, we report the algorithm performance with respect to different percentages of labeled samples, which vary from 10% to 80% as indexed by x-axis and y-axis shows the accuracies of the classifiers trained from the corresponding labeled samples. For each benchmark dataset, we use two types of benchmark learners, Decision trees and Naive Bayes, and report their accuracies in the figure. For all experiments, we fix the batch size to 0.05 and the initial randomly labeled subset is 0.05. Due to page limitations, we only report results for Horse and Car datasets. Detailed results of each method on all benchmark datasets are reported in the next section.

The Entropy method, without considering sampling redundancy, is the least effective algorithm mainly because the instance uncertainty is calculated based on each sample's own information, and the correlation between samples is ignored. By employing optimal subset selection, we observe that ALOSS constantly outperforms SIB. Because the batch size for SIB is set as 1, and a smaller batch results in better labeling quality, SIB represents the performance upper bound of individual assessment based active methods. By combing instance uncertainty and instance disparity together to select optimal subsets for active learning, ALOSS is shown to outperform the upper bound to a large extent.

4.4.5 Detailed Comparisons for All Methods

In Tables 4.3, 4.4 and 4.5, we report detailed comparisons of all methods on 10 benchmark datasets. In all tables, the batch size is fixed to 0.05.

Among all methods, ALOSS achieves the best performance gain, ALOSS_p and SIB are the second tier. Instance weighting (IW) based approach, however, marginally outperforms random sample selection. As we have discussed in Eq.(4.28), the instance weighting in IW is essentially an individual assessment based measure, where the importance of each sample (*i.e.* the weight value) is generated according to its loss values with respect to some classifiers. This observation again asserts that instance correlations play an important role for active learning methods to select informative and less redundant instances for labeling.

Intuitively, SIB intends to avoid sample labeling redundancy by selecting instance one at a time for labeling. Such a hill-climbing instance labeling approach, however, is still inferior to ALOSS, although SIB indeed outperforms most other methods (despite of the high computational costs of SIB). Indeed, although SIB intends to minimize redundancy in the sample selection process, it has no mechanism to ensure that samples selected in a consecutive number of iterations can form an optimal subset. Just like most hill-climbing search methods can be stuck to some local optimal without finding a good solution, ALOSS inherently avoids the problem through the selection of an optimal subset by taking sample correlations into consideration.

Table 4.3: Detailed algorithm performance comparisons (using Decision Trees as the benchmark learner and labeling percentage is 15%)

Dataset	Random	Margin	Entropy	SIB	IW	ALOSS$_p$	ALOSS
horse	$82.85_{\pm 1.20}$	$83.12_{\pm 1.09}$	$82.98_{\pm 1.70}$	$83.02_{\pm 1.28}$	$82.95_{\pm 1.76}$	$83.17_{\pm 1.69}$	$\mathbf{83.78}_{\pm 1.31}$
auto-mpg	$67.73_{\pm 2.16}$	$67.75_{\pm 2.96}$	$67.97_{\pm 1.12}$	$68.52_{\pm 1.68}$	$67.04_{\pm 2.62}$	$68.21_{\pm 2.13}$	$\mathbf{69.77}_{\pm 1.87}$
balance	$62.13_{\pm 1.19}$	$63.20_{\pm 2.9}$	$62.91_{\pm 1.28}$	$63.94_{\pm 1.94}$	$63.36_{\pm 1.2}$	$63.94_{\pm 1.23}$	$\mathbf{64.32}_{\pm 2.24}$
pima	$69.55_{\pm 1.48}$	$70.11_{\pm 1.88}$	$70.24_{\pm 1.68}$	$70.96_{\pm 1.34}$	$70.89_{\pm 1.96}$	$70.75_{\pm 1.94}$	$\mathbf{71.88}_{\pm 1.91}$
vehicle	$67.98_{\pm 1.12}$	$69.03_{\pm 1.93}$	$68.89_{\pm 1.14}$	$69.56_{\pm 1.98}$	$68.05_{\pm 1.43}$	$69.17_{\pm 1.45}$	$\mathbf{70.59}_{\pm 2.12}$
german	$68.54_{\pm 1.13}$	$67.21_{\pm 1.73}$	$68.76_{\pm 1.38}$	$69.64_{\pm 1.71}$	$68.56_{\pm 1.56}$	$70.09_{\pm 2.05}$	$\mathbf{70.13}_{\pm 1.57}$
cmc	$57.23_{\pm 1.16}$	$58.90_{\pm 1.78}$	$57.98_{\pm 2.06}$	$58.98_{\pm 1.76}$	$59.01_{\pm 2.19}$	$59.23_{\pm 1.82}$	$\mathbf{60.12}_{\pm 2.17}$
car	$83.35_{\pm 1.93}$	$84.76_{\pm 1.75}$	$84.68_{\pm 1.25}$	$85.07_{\pm 1.73}$	$83.58_{\pm 2.87}$	$85.45_{\pm 1.25}$	$\mathbf{86.78}_{\pm 2.15}$
segment	$82.29_{\pm 3.08}$	$82.75_{\pm 2.12}$	$83.84_{\pm 2.17}$	$\mathbf{85.06}_{\pm 1.25}$	$84.98_{\pm 1.25}$	$83.78_{\pm 1.69}$	$84.57_{\pm 1.95}$
analone	$80.78_{\pm 1.56}$	$82.85_{\pm 1.2}$	$81.67_{\pm 1.34}$	$82.56_{\pm 1.76}$	$80.21_{\pm 3.21}$	$83.23_{\pm 1.35}$	$\mathbf{84.72}_{\pm 2.18}$
Average	$72.24_{\pm 1.60}$	$72.96_{\pm 1.93}$	$72.99_{\pm 1.51}$	$74.63_{\pm 1.69}$	$72.86_{\pm 2.00}$	$73.72_{\pm 1.66}$	$\mathbf{75.31}_{\pm 2.04}$

Table 4.4: Detailed algorithm performance comparisons (using Decision Trees as the benchmark learner and labeling percentage is 30%)

Dataset	Random	Margin	Entropy	SIB	IW	ALOSS$_p$	ALOSS
horse	$83.31_{\pm 1.08}$	$83.96_{\pm 1.13}$	$84.07_{\pm 1.29}$	$84.31_{\pm 1.38}$	$84.11_{\pm 1.43}$	$84.53_{\pm 2.43}$	$\mathbf{85.47}_{\pm 1.76}$
auto-mpg	$68.36_{\pm 1.81}$	$71.08_{\pm 1.60}$	$69.97_{\pm 1.05}$	$70.88_{\pm 1.34}$	$69.04_{\pm 1.68}$	$71.85_{\pm 2.08}$	$\mathbf{72.06}_{\pm 1.45}$
balance	$63.54_{\pm 1.2}$	$64.73_{\pm 2.39}$	$64.85_{\pm 1.69}$	$65.09_{\pm 1.07}$	$65.18_{\pm 1.38}$	$65.25_{\pm 2.06}$	$\mathbf{66.25}_{\pm 1.46}$
pima	$70.91_{\pm 1.10}$	$71.15_{\pm 1.76}$	$70.96_{\pm 1.25}$	$71.68_{\pm 1.60}$	$71.72_{\pm 1.01}$	$71.97_{\pm 1.56}$	$\mathbf{72.78}_{\pm 1.38}$
vehicle	$69.06_{\pm 1.15}$	$70.87_{\pm 1.25}$	$70.08_{\pm 1.95}$	$71.51_{\pm 2.06}$	$71.06_{\pm 2.2}$	$71.37_{\pm 1.12}$	$\mathbf{72.56}_{\pm 2.56}$
german	$69.64_{\pm 1.01}$	$69.75_{\pm 1.02}$	$68.98_{\pm 1.60}$	$69.62_{\pm 1.71}$	$70.29_{\pm 2.03}$	$71.79_{\pm 1.45}$	$\mathbf{72.89}_{\pm 1.96}$
cmc	$58.01_{\pm 1.09}$	$60.07_{\pm 1.98}$	$59.80_{\pm 2.31}$	$61.52_{\pm 1.83}$	$\mathbf{62.43}_{\pm 2.32}$	$61.04_{\pm 2.24}$	$62.26_{\pm 2.12}$
car	$85.35_{\pm 1.85}$	$86.12_{\pm 1.22}$	$85.93_{\pm 1.85}$	$86.72_{\pm 1.74}$	$85.58_{\pm 2.87}$	$87.05_{\pm 2.12}$	$\mathbf{88.78}_{\pm 2.15}$
segment	$85.24_{\pm 1.03}$	$85.62_{\pm 2.29}$	$86.02_{\pm 1.05}$	$86.73_{\pm 1.60}$	$85.23_{\pm 2.83}$	$86.27_{\pm 1.56}$	$\mathbf{87.98}_{\pm 1.05}$
analone	$81.78_{\pm 1.43}$	$83.65_{\pm 1.47}$	$82.56_{\pm 1.82}$	$83.71_{\pm 1.91}$	$82.13_{\pm 2.21}$	$85.34_{\pm 1.72}$	$\mathbf{86.12}_{\pm 1.65}$
Average	$73.52_{\pm 1.27}$	$74.7_{\pm 1.61}$	$74.32_{\pm 1.58}$	$75.17_{\pm 1.62}$	$74.67_{\pm 1.99}$	$75.64_{\pm 1.83}$	$\mathbf{76.82}_{\pm 1.75}$

Table 4.5: Detailed algorithm performance comparisons (using Decision Trees as the benchmark learner and labeling percentage is 50%)

Dataset	Random	Margin	Entropy	SIB	IW	ALOSS$_p$	ALOSS
horse	$84.37_{\pm 1.51}$	$84.75_{\pm 1.07}$	$84.87_{\pm 1.45}$	$85.12_{\pm 1.47}$	$84.98_{\pm 2.56}$	$85.34_{\pm 1.60}$	$\mathbf{86.18}_{\pm 1.06}$
auto-mpg	$74.08_{\pm 1.98}$	$75.17_{\pm 1.69}$	$75.01_{\pm 2.08}$	$75.92_{\pm 1.69}$	$74.05_{\pm 1.2}$	$76.07_{\pm 1.12}$	$\mathbf{77.09}_{\pm 2.29}$
balance	$65.92_{\pm 1.03}$	$65.75_{\pm 2.35}$	$65.89_{\pm 1.48}$	$66.34_{\pm 1.19}$	$66.94_{\pm 1.39}$	$66.89_{\pm 2.36}$	$\mathbf{67.21}_{\pm 1.96}$
pima	$72.96_{\pm 1.10}$	$72.92_{\pm 1.29}$	$73.05_{\pm 1.12}$	$73.99_{\pm 2.01}$	$73.07_{\pm 1.19}$	$73.87_{\pm 1.23}$	$\mathbf{74.92}_{\pm 1.23}$
vehicle	$70.98_{\pm 1.13}$	$72.01_{\pm 2.8}$	$71.45_{\pm 1.34}$	$72.89_{\pm 1.65}$	$72.09_{\pm 2.1}$	$73.56_{\pm 1.18}$	$\mathbf{74.78}_{\pm 3.56}$
german	$70.78_{\pm 1.01}$	$70.96_{\pm 1.04}$	$71.08_{\pm 1.56}$	$71.78_{\pm 1.67}$	$71.09_{\pm 2.53}$	$72.67_{\pm 1.95}$	$\mathbf{73.27}_{\pm 1.65}$
cmc	$59.03_{\pm 1.75}$	$61.09_{\pm 2.08}$	$60.45_{\pm 1.34}$	$61.86_{\pm 1.12}$	$62.03_{\pm 1.23}$	$62.34_{\pm 1.71}$	$\mathbf{63.54}_{\pm 2.56}$
car	$86.97_{\pm 1.93}$	$88.15_{\pm 1.74}$	$87.37_{\pm 2.16}$	$88.36_{\pm 1.92}$	$87.09_{\pm 3.21}$	$88.96_{\pm 1.46}$	$\mathbf{90.13}_{\pm 2.96}$
segment	$86.24_{\pm 1.25}$	$86.02_{\pm 2.89}$	$87.12_{\pm 1.12}$	$87.89_{\pm 1.67}$	$87.23_{\pm 2.13}$	$88.31_{\pm 1.72}$	$\mathbf{89.98}_{\pm 1.34}$
analone	$83.78_{\pm 1.23}$	$84.87_{\pm 2.12}$	$84.65_{\pm 1.51}$	$85.71_{\pm 1.65}$	$84.23_{\pm 1.22}$	$86.12_{\pm 1.34}$	$\mathbf{88.12}_{\pm 1.89}$
Average	$75.51_{\pm 1.39}$	$76.17_{\pm 1.91}$	$76.09_{\pm 1.51}$	$76.98_{\pm 1.60}$	$76.28_{\pm 1.87}$	$77.41_{\pm 1.57}$	$\mathbf{78.52}_{\pm 2.05}$

Chapter 5

Pairwise Homogeneity Based Active Learning

5.1 Introduction

In most traditional active learning methods, an expert labeler (also called "oracle ") is required to provide ground truths to the queried instances and the model is updated by incorporating 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 because the selected subset can still be of large size and the active learning iterations can last for quite a long time. In addition, since 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 (non-expert) labelers, which are cheaper but can only provide noisy labels for unlabeled instances. Some works based on this idea have been proposed, such as [103, 93], 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 are 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 non-expert labelers to directly provide noisy labels may not be appropriate in active learning scenarios.

We propose a new active learning paradigm, named Pairwise Homogeneity based Active Learning (PHAL), in which a non-expert 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. 5.1, 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 actively label some of the remaining pictures to train a classifier. In traditional active learning, a domain expert may be needed to label these pictures since non-expert may find it difficult to tell the genuine label of (c), which seems unlike to either (a) or (f). In our proposed 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 non-expert.

The proposed PHAL not only imposes less requirements to the labelers and makes the labeling more affordable for real-world applications, but also shows good robustness to tolerate labeling errors, as we will soon demonstrate in the experiments. Indeed, in the example showing in Fig. 5.1, it does not matter even if the non-expert tell the wrong answer 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.

Pairwise homogeneity relationships can be commonly seen in many applications [125, 27], where this prior information can be easily obtained with very little efforts. For example, in Facebook, a popular social network site, a user favoring a pollical leader (*e.g.* Barack Obama) may publish interesting status in his/her personal page and other users can reply to these posts as "comments ". Any two

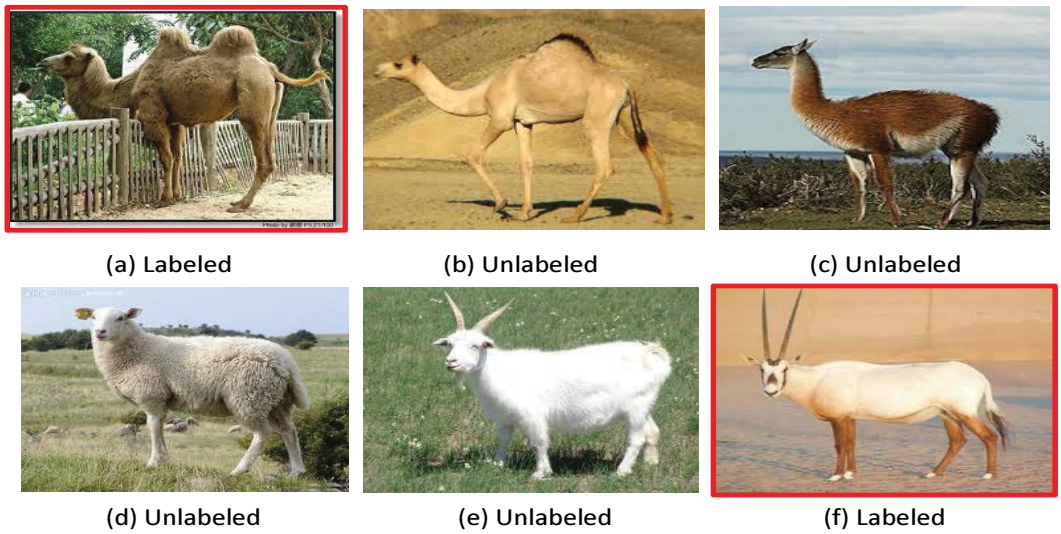


Figure 5.1: An example of using pairwise label homogeneity to ease a labeling task. (a) and (f) are labeled as "camel "and "sheep ", respectively, and the labeling task is to actively label remaining pictures to train a classifier. (c) is unlike to either (a) or (f), so its genuine label is difficult to obtain unless the labeler has sufficient knowledge (or a domain expert is involved). 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 non-expert.

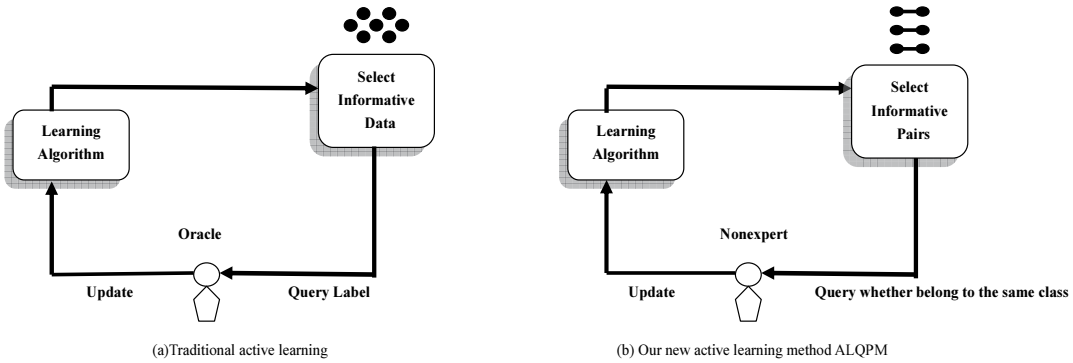


Figure 5.2: Traditional active learning paradigm *vs.* the proposed PHAL paradigm. Traditional active learning explicitly queries the class label of each instance, whereas our approach queries the class homogeneity between a pair of instances, i.e., whether a pair of instances belong to the same class or not. Since we do not require the labeler to provide class label of each queried instance, our query is much easier/cheaper to collect in reality.

comments corresponding to one post are in the same category (*i.e.* topic "political ") and therefore share the homogeneity relationship. Similar situation can also be observed in Twitter and Google News, where only a coarse web analysis script is needed to automatically detect the pair relationship between tweets or news. When collecting labeled instances for model training is difficult or even impossible, the pairwise information provides valuable knowledge for build learning models without knowing the genuine label of individual instances.

Based on the above assumption, the underlying queries in PHAL are to generate pairwise constraints between labels, which will be incorporated into the active learning procedure. Fig. 5.2 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 non-expert labeler based active leaning paradigm provides an opportunity to reduce labeling cost and make a labeling task easier to fulfill, 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 pairs 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.

The problem formulation and the proposed method for active learning by querying pairwise label homogeneity are presented in Sections 5.2 and 5.3, respectively. Experimental results are reported in Section 5.4.

5.2 Problem Formulation

5.2.1 Problem Setting

A given data set is denoted by $\mathcal{D}$, 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 y_i the class label. Meanwhile, the ith instance in $\mathcal{D}^U$ or $\mathcal{D}^T$ is denoted by $(\mathbf{x}_i, ?)$, where the label is unknown. In order to train a classifier on $\mathcal{D}^L$ with maximum prediction accuracy, a common 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 answering the class labels of queried instances in traditional active learning methods, we consider a pairwise label homogeneity query setting in this paper. Assume we employ a non-expert 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.

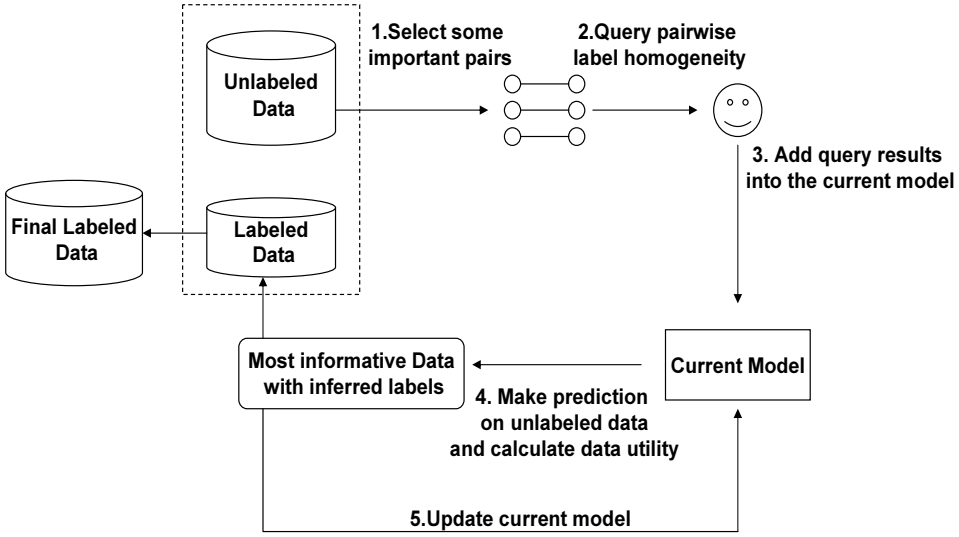


Figure 5.3: The proposed Pairwise Homogeneity based Active Learning (PHAL) framework.

5.2.2 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 non-expert 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. 5.3.

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 [13, 14] as a 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 and the other for negative instances). Since Max-flow paths play an important role

for graph bipartition, we select the 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 non-expert labeler is asked to provide label homogeneity information to the queried pairs. Based on the query results, we adjust the weights of 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 based on their label confidences, the top vertices are selected as the optimal subset Δ.

4. **Weight Update in Selected Subset**. We use the final prediction result of an unlabeled vertex in Δ as its class label. Then we further check the edges which have the vertices in Δ. 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 Δ, 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 Υ (the total number of queried pairs during active learning iterations). In each iteration,

the active learner selects a small optimal subset Δ and treats the predicted labels as their real labels. This information is then used to update both the graph ensemble $\mathbb{G}$ and the classifier ensemble $E_{\hbar}$, and also helps select the optimal subset in the next iteration.

5.3 The Proposed Method

In this section, we will introduce the proposed method in detail for active learning by querying pairwise label homogeneity. Sections 5.3.1 and 5.3.2 will address the first challenge mentioned in Section 5.2.1, and Sections 5.3.3 and 5.3.4 will address the second challenge. Finally, we will analyze the computational complexity of our method.

5.3.1 Graph Ensemble Construction

Given a distance metric, there exist a number of ways to construct graphs. In the following, we introduce design criteria for constructing an graph ensemble used in the Min-cut algorithm. First of all, it is expected that a graph has at least 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. Secondly, 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 [30] to construct graphs. We connect two vertices (instances) with an edge if one vertex is a member of the other one's top k nearest neighbors. This setting cater the first criterion with the assumption that the 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 the instances in $\mathcal{D}^L \cup \mathcal{D}^U$ to form the vertex set $V = V^L \cup V^U$ in the graph ensemble. That is to say, $v_i \in V$ 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 $\mathfrak{E}_1, \ldots, \mathfrak{E}_N \subseteq V \times V$, respectively. As a result, we can obtain a graph ensemble $\mathbb{G} = \{g_1 = (V, \mathfrak{E}_1), \ldots, g_N = (V, \mathfrak{E}_N)\}$. We use V_+^L to indicate the vertex set with positive labels and 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**. Since 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. Thus, the vertex set for constructing the graph ensemble becomes $V = V \cup \{v_+, v_-\}$. All the other vertices in 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 V_+^L and 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 $v_i \in V_+^L$ and $w(v_-, v_i) = \infty$ for all $v_i \in V_-^L$.

- **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,

$$w(v_i, v_j) = \exp(-\Upsilon(v_i, v_j)) \tag{5.1}$$

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

5.3.2 Weight Adjustment in Min-cut Sets

After graph construction, we use the Min-cut algorithm to bipartition the graphs for binary classification [13]. The Min-cut based classifiers are based on the Max-flow Min-cut theorem [89], which states that, given a flow network, the maximum flow passing from the source to the sink equals to the minimum cut of edge capacities (weights) in the network.

Theorem 1 (Max-flow Min-cut Theorem [89])**.** *Let χ be a flow passing from the source to the sink in a network g and $(\mathcal{A}, \mathcal{B})$ be a cut, where $g = \mathcal{A} \cup \mathcal{B}$. Then, for any cut, we have $\chi(g) \leq \varphi(\mathcal{A}, \mathcal{B})$, where $\varphi(\mathcal{A}, \mathcal{B})$ is the capacity of the cut. When $\chi(g) = \varphi(\mathcal{A}, \mathcal{B})$, χ is a maximum flow and $(\mathcal{A}, \mathcal{B})$ is a minimum cut of the network.*

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 [13, 14] belongs to a family of semi-supervised learning methods based on the manifold assumption [136], 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 [38].

To acquire pairwise label homogeneity information, we query "whether v_i and v_j belong to the same class"to the non-expert labeler. Based on the query result,

we adjust the weight of edge between v_i and v_j as follows

$$w(v_i, v_j) = \begin{cases} w(v_i, v_j) \times (1 + \partial) & \text{if } y_i = y_j \\ w(v_i, v_j) \times (1 - \partial) & \text{if } y_i \neq y_j \end{cases} \tag{5.2}$$

where ∂ is an adjustment factor, which determines the weight updating scale ($0 < \partial < 1$). According to Eq. (5.2), 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 (LOO) error of the underlying Min-cut based classifier. Lemma 3.4 in [13] 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 [60]

$$\iota_i = y_i \frac{\sum_{j \in kNN(j)} y_j w(v_i, v_j)}{\sum_{j \in kNN(i)} w(v_i, v_j)} \tag{5.3}$$

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

$$\epsilon_{LOO}(\mathcal{D}) \leq \sum_{i=1}^{|\mathcal{D}|} (1 - \iota_i) \tag{5.4}$$

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. (5.4) can become tighter if the edge weights are adjusted according to the pairwise label homogeneity query result using Eq. (5.2).*

Proof. To prove that the upper bound Eq. (5.4) can become tighter, we can equivalently prove that Eq. (5.5) will increase after adjusting the edge weights using Eq. (5.2).

$$\sum_{i=1}^{|\mathcal{D}|} \iota_i = \sum_{i=1}^{|\mathcal{D}|} \sum_{j \in kNN(i)} y_i y_j \frac{w(v_i, v_j)}{\sum_{j \in kNN(i)} w(v_i, v_j)} \tag{5.5}$$

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. (5.5) 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. (5.5) 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. (5.4). Over active learning iterations, the decision boundary of the classifiers will become clearer and the imbalanced cut issue will be relieved.

5.3.3 Confidence based Data Selection

By applying the Min-cut algorithm to the graph ensemble $\mathbb{G}$, an ensemble of Min-cut based classifiers $E_\hbar = \{\hbar_1, \ldots, \hbar_m\}$ are naturally derived on $\mathbb{G}$. Thus, the class labels of all the 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 the unlabeled vertices in $\mathbb{G}$, we employ the predictions of $E_\hbar$ on each vertex to calculate its class label distribution

$$p_\hbar(+, v_i) = \frac{1}{m}\sum_{n=1}^{m}\mathbb{I}(\hbar_n(v_i) = +) \tag{5.6}$$

$$p_\hbar(-, v_i) = 1 - p_\hbar(+, v_i) \tag{5.7}$$

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

$$P_c(v_i) = \max\{p_\hbar(+, v_i), p_\hbar(-, v_i)\} \tag{5.8}$$

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

5.3.4 Weight Update in Selected Subset

After incorporating the labeling information of the optimal subset Δ 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 Δ, 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. (5.2) introduced in Section 5.3.2

$$w(v_i, v_j) = \begin{cases} w(v_i, v_j) \times (1 + \varphi) & \text{if } y_i = y_j \\ w(v_i, v_j) \times (1 - \varphi) & \text{if } y_i \neq y_j \end{cases} \tag{5.9}$$

where φ is an adjustment factor, which determines the weight updating scale ($0 < \varphi < 1$). Since 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. (5.2), where the query answers are almost accurate.

The entire algorithm is described in Algorithm 4.

5.3.5 Time Complexity Analysis

In order to analyze the time complexity of our method, we 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: $B(V)$ and $U(V)$, where $B(V)$ denotes the time complexity for model training and $U(V)$ for active pair selection for label homogeneity queries.

The term $B(V)$ is further composed by the complexity of a graph ensemble construction $BG(V)$ and the complexity of the Min-cut based classifier ensemble training $BM(V)$. As aforementioned, we use the k-NN algorithm to construct N graphs with different number of neighbors. This procedure has a complexity of $O(|V|^2 + N|V|)$, where $O(|V|^2)$ is for computing the pairwise similarity matrix

Algorithm 4 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) $V^L = V_+^L \cup V_-^L \cup \{v_+, v_-\}$: a labeled vertex set (5) Υ: the budget of pairs queried during active learning.

Output: V^L

1: Construct a graph ensemble $\mathbb{G} = \{g_1, \ldots, g_N\}$ with different k values based on $\mathcal{D}^L \cup \mathcal{D}^U$;
2: **while** *the number of queried pairs* $< \Upsilon$ **do**
3: Learn the Min-cut classifier ensemble $E_\hbar = \{\hbar_1, \ldots, \hbar_N\}$ 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: Update edge weights $w(v_i, v_j) \leftarrow$ Eq.(5.2)
6: Compute prediction confidences $P_c(v_i) \leftarrow$ Eq.(5.8) for $v_i \in V^U$ and $c \in \{+, -\}$;
7: Predict the class memberships of $v_i \leftarrow arg_c$ Eq.(5.8)
8: Sort V^U according to their prediction confidences and select the top ones as the optimal subset $\mathfrak{E}$;
9: $\mathcal{V}^U \longleftarrow \mathcal{V}^U \setminus \mathfrak{E}$;
10: $\mathcal{V}^L \longleftarrow \mathcal{V}^L \cup \mathfrak{E}$;
11: Update the edge weight $w(v_i, v_j) \leftarrow$ Eq.(5.9) if exist $v_i \in \mathfrak{E}$;
12: **end while**

and $O(N|V|)$ 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 [38], which has a complexity of $O(|V||\mathfrak{E}|^2)$, for solving the Min-cut problem. Therefore, we retrain N Min-cut based classifiers for T times, which totally has a complexity of $T\sum_{n=1}^{N} O(|V||\mathfrak{E}_n|^2)$. The total time complexity of $B(V)$ is

$$B(V) = BG(V) + BM(V) \tag{5.10}$$

$$= O(|V|^2) + O(N|V|) + T\sum_{n=1}^{N} O(|V||\mathfrak{E}_n|^2) \tag{5.11}$$

In practice, since $|\mathfrak{E}_n|$ is larger than $|V|$ while N and T are usually small. Let $|\mathfrak{E}|$ be the average of $|\mathfrak{E}_n|$, we can further simplify Eq. (5.10) to be

$$B(V) = O(TNV|\mathfrak{E}|^2) \tag{5.12}$$

The term $U(V)$ is further composed by the complexity of pair queries $UP(V)$ and the complexity of optimal labeled data selection $UC(V)$. We assume that the average number of queried pairs in each iteration is M. Then the total complexity of T iterations is $UP(V) = O(TM)$. For $UC(V)$, the membership distribution estimation needs $O(N|V^U|)$ and the unlabeled data sorting based on confidences

needs $O(|V^U|^2)$, both of which are iterated for T times. As a result, the time complexity of $U(V)$ is

$$\begin{aligned} U(V) &= UP(V) + UC(V) && (5.13)\\ &= O(TM) + O(TN|V^U|) + O(T|V^U|^2) && (5.14)\\ &= O(TM) + O(T|V^U|^2) && (5.15) \end{aligned}$$

With the above analysis, we get the overall time complexity of our method

$$B(V) + U(V) = O(TNV|\mathfrak{E}|^2) \quad (5.16)$$

Eq. (5.16) shows that most computational cost of our method rests on the Min-cut based classifier ensemble training.

5.4 Experiments

In this section, we first investigate parameter settings and noise sensitivity of the proposed method to validate its robustness. The effectiveness of the proposed method is validated by extensive comparisons with a number of baseline methods.

5.4.1 Data Description and Experimental Settings

We conduct experiments on ten benchmark data sets listed in Table 5.1. All benchmark data sets except "lucas "are real-world binary data sets, which can be downloaded from the UCI Machine Learning Repository[1]. "lucas "is a synthetic data set[2] to simulate a medical application of lung cancer diagnosis, prevention, and treatment. It is generated using causal Bayesian networks with binary variables, where the target variable denotes whether a patient has lung cancer or not.

For fair comparisons, all experimental results are reported based on the average results of 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.

[1] http://archive.ics.uci.edu/ml/

[2] http://www.causality.inf.ethz.ch/challenge.php?page=datasets

All the compared methods are implemented using Java and WEKA [118] 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 their J48 classifiers 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 a method has a better active learning performance than its peers if it outperforms the baseline methods in terms of the classification accuracy.

Table 5.1: 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

5.4.2 Baseline Methods

To the best of our knowledge, there are no existing methods 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)** [41] 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 update 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 two vertices linked by the selected edge.

- **Uncertain Sample based Active Learning (USAL)** uses entropy [98] 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 the 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

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

good choice. Given an active learning task, if we observe an increasing performance gain from PHAL(α) as the value of α increases, it will validate that querying instance pairs on the Max-flow paths is, at least, a better choice than random query.

In addition to the above pairwise label homogeneity query based baseline methods, we also consider an individual instance query based baseline to compare the sensitivity of the proposed method and traditional active learning methods with respect to labeling noise.

- **Instance Label based Active Learning(ILAL)** employs a labeler who can provide class label for the queried instance. It uses entropy as the instance utility measure for instance selection.

For fair comparisons, all the 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 un-queried pairs on the Max-flow paths. However, due to the Max-flow paths often change as the active learning process iterates, the number of these pairs is not fixed in all iterations. Thus we guarantee that all the baseline methods query the same number of pairs as PHAL does in each iteration. For USAL, given n samples, 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

$$\varrho = \frac{\gamma(\gamma - 1)}{2} \tag{5.17}$$

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

5.4.3 Parameter Setting for k-NN Graphs

QHAL uses a predefined k 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. However, it cannot guarantee that the selected k achieves the best performance, unless we exhaustively search the optimal k, which is computationally expensive. Moreover, as explained in Section 5.3.1, it is difficult to find a general criterion to search the optimal k since it often depends on data

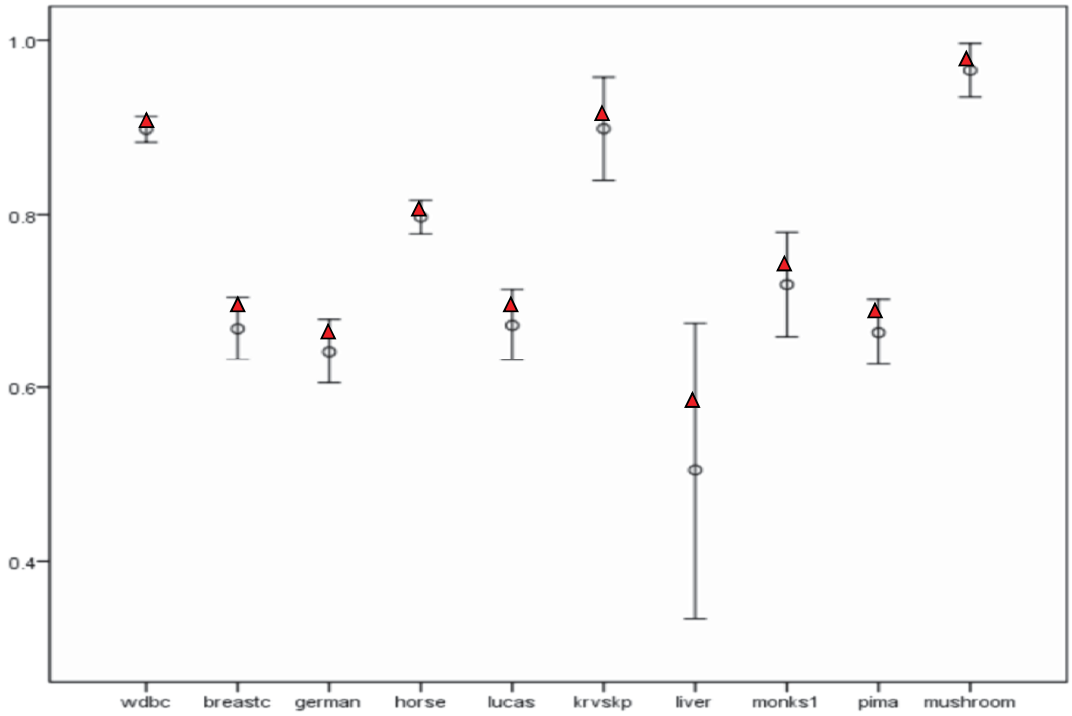


Figure 5.4: The accuracy comparison between QHAL and PHAL. Each vertical line segment denotes the accuracy range of QHAL with k varying from 3 to 24 with a step of 3, and the circle on the line denotes the average accuracy in the range; a red triangle denotes the accuracy of PHAL, which comprises an ensemble of k-NN graphs with k varying from 3 to 15 with a step of 3.

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.

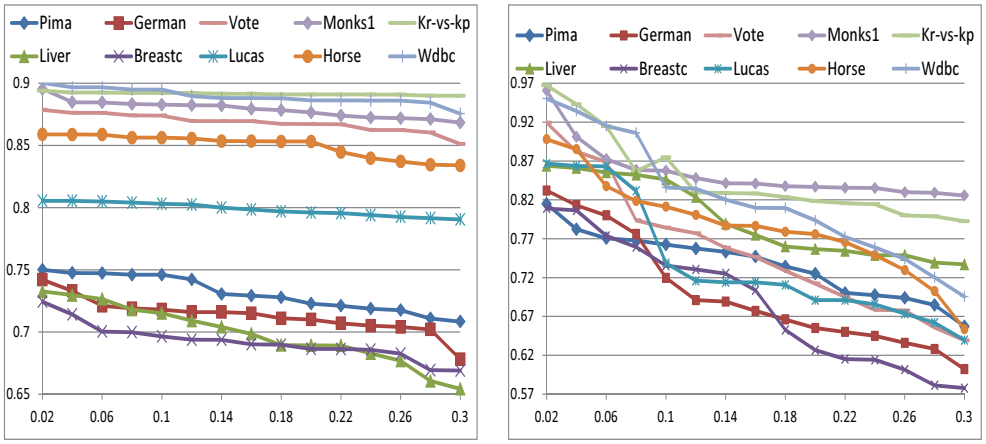
In Fig. 5.4, we report the results of QHAL and PHAL with respect to different k values on 10 benchmark data sets. The results show that the performance of PHAL is always better than the average performance of QHAL with different k values in a large range (ranging from 3 to 24 with a step of 3). Although 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.

5.4.4 Sensitivity w.r.t. Different Percentages of Labeling Noise

In real-world scenarios, labelers may provide noisy pairwise label homogeneity information for some uncertain cases. In this experiment, we comparatively study the sensitivities of PHAL (pairwise homogeneity query based) and ILAL (instance label query based) to the labeling noise. To measure the robustness of a model against noise, we can investigate its accuracy curve with respect to the increasing percentages of noisy labels. The more slowly the accuracy curve drops, the more robust the model is; otherwise, the model is sensitive to noise. By using this approach, we can compare the accuracy decreasing rates of PHAL and ILAL to validate whether PHAL is more robust than ILAL, or vice versa. 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 required percentage of queries as noisy labels. In particular, we randomly generate binary labels for the queried instance pairs in PHAL; while we randomly generate class labels for the queried individual instances in ILAL. Since PHAL queries the pairs on Max-flow paths on the graphs, the number of queried pairs may exceeds the number of the data set. Thus we cannot guarantee that ILAL queries the same number of instances as PHAL does at each time. Nevertheless, we guarantee that the same number of labeled instances are included into the labeled data set in each active learning iteration for both methods.

Fig. 5.5 reports the accuracy curves of PHAL and ILAL on 10 data sets, respectively, with the percentages of label noise ranging from 2% to 30% with a step of 2%. For PHAL, its accuracy curves on “kr-vs-kp ”, “monks1 ”, “vote ”, and “lucas ”slight drop within the whole range, while the curves on the other 6 data sets decline slowly as the noise increases. For ILAL, its accuracy curves on all 10 data sets decrease more quickly than the corresponding curves of PHAL. This phenomenon might be caused by the following reason: Instance label query results are directly used for supervised model training such that the noise will directly impact on the the learning model. PHAL, on the other hand, only uses pairwise label homogeneity information to update the affinity graphs. Because the classification results of



(a) PHAL (b) ILAL

Figure 5.5: The sensitivity of different methods to labeling noise. Accuracies (y-axis) of (a) PHAL and (b) ILAL on 10 data sets with respect to different percentages of noisy labels ranging from 2% to 30% with a step of 2% (x-axis).

PHAL depend on the overall structures of the affinity graphs, changes introduced to 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 much less sensitive to noise.

5.4.5 Comparison of Different Pair Selection Strategies

Fig. 5.6 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 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 increasing queried pairs.

As the number of queries increases, the performance of all the methods improves to some extent. 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 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 non-expert 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 non-expert labeler to give 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 big performance gap between the graph-based pair selection methods and USAL.

The last observation is that UPAL outperforms REAL in most cases. These

results 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.

5.4.6 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 5.2 (a), (b) and (c) report the detailed performance of all the compared methods on 10 benchmark data sets in the 3rd iteration, the 6th iteration, and the 9th 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 Section 5.4.2, UPAL integrates uncertainty measure in the pair selection strategy, which does help Min-cut based classifiers 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 Section 5.4.2, 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 with a graph topology. These results demonstrate that pairwise correlations do play an important role for pairwise label homogeneity based methods to select informative pairs for labeling.

Table 5.2: Detailed performance comparison

(a) After the 3rd iteration

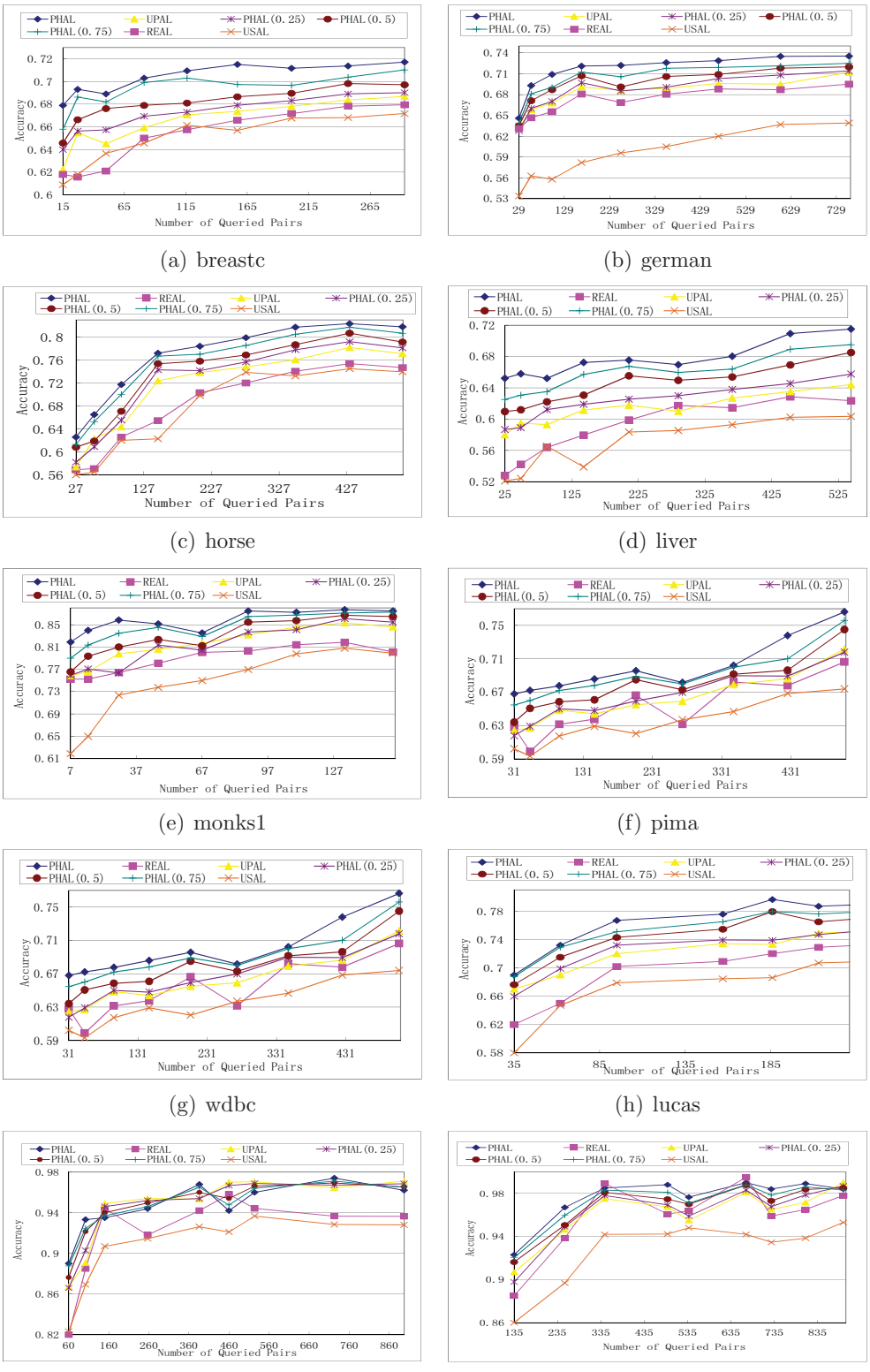
Dataset	♯ of labeled pairs	PHAL	REAL	UPAL	USAL
breastc	50	**0.689**	0.621	0.642	0.636
liver	88	**0.652**	0.570	0.593	0.568
wdbc	90	**0.950**	0.921	0.932	0.821
monks1	29	**0.858**	0.764	0.797	0.723
pima	96	**0.677**	0.636	0.645	0.617
horse	94	**0.717**	0.625	0.643	0.620
lucas	95	**0.767**	0.702	0.720	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



(a) breastc (b) german

(c) horse (d) liver

(e) monks1 (f) pima

(g) wdbc (h) lucas

(i) vote (j) kr-vs-kp

Figure 5.6: Performance comparison of PHAL and the baseline methods with different pair selection strategies on 10 data sets.

Chapter 6

Active Learning with Unknown Class Space and Weak Labeling Knowledge

6.1 Introduction

An active learner builds a powerful predictive model with as few labeled samples as possible, thereby minimizing the manual labeling requirement. However, most existing active learning methods assume that the number of class labels is known and a labeler has expertise to provide the ground truth for each queried instance. Under such circumstance, a number of randomly selected instances are labeled by the labeler as an initial training set. So active learning process can kick-off to gradually expand the training set. We refer to this type of active learning as **Hot-Start** Active Learning.

However, in dynamic data environments, the number of classes in data sets is unknown beforehand. In addition, new classes may emerge so the classes in the data do not remain stable as **Hot-Start** active learning has assumed. For example, micro-blogging sites like *Twitter* have rich source of information about "event ", ranging from entertainment gossip to political news. It is often in a matter of seconds to break important events after they occur. So it is immensely valuable to discover such emerging events timely and make them available. When a new event appears, traditional active learning techniques do not help to discover new topic but misclassify it to one of existing known classes. On the other hand, the labeler may

also have very limited knowledge about the underlying data, so accurately labeling each single instance is impossible. Instead, the labeler can answer pairwise label homogeneity query, which is the same pairwise correlation used in Chapter 5.

For active learning tasks with incomplete class information, some methodologies in terms of rare class discovery have been developed [56, 52, 48]. Rare class discovery in active learning focuses on exploring rare class(es) whose labeled samples are missed at the early stage of labeling process. This is mainly because those samples representing a rare class are a small portion of the total population, so random labeling has little chance to cover rare class samples. The active learning goal, under such circumstances, is to discover rare class samples as early as possible. Several methods exist to discover rare class samples by using likelihood [90], gradient [51] or clustering [114] criteria. However, all these methods assume that the number of classes (including rare classes) is known beforehand. In addition, they mainly focus on finding new classes, and not on improving the classification performance (refining the decision boundaries). Recently, several approaches have tried to address class discovery and classification simultaneously. Stokes et al. [108] developed a non-adaptive query strategy by fixing the proportion of discovery and classification criteria. Hospedales et al. [56] considered two models (generative and discriminative models) and proposed the use of two active learning criteria (new class discovery and boundary refinement) at the same time. Hanies and Xiang [48] formulated an active query strategy using the Dirichlet process to balance new class discovery and misclassification error. For all these methods, the number of classes in the data is known a priori. Moreover, they all assume that labelers can provide ground truth for each queried instance, which is hardly the case in reality because the labeler may not have the knowledge to label new class samples.

When labelers have weak labeling knowledge and cannot provide the ground truth of each single instance, solutions also exist to leverage weak labeling knowledge by using pairwise correlation [41, 6, 116]. Pairwise correlation has been applied to both active learning and active clustering, specifying whether two instances have the same membership or not. For classification, a pairwise correlation indicates whether a pair of instances belong to the same class (positive constraint) or not (negative constraint). [41] applies pairwise correlation to a graph-based classifier

such as the Mincut classifier for bipartition tasks. [125] presents a discriminative learning method by learning the decision boundary with labeled data, as well as additional constraints. [83] also introduces a discriminative learning approach that incorporates pairwise correlation into the traditional margin-based learning framework. In active clustering, pairwise correlation is presented in the form of either a *must-link* constraint where two instances belonging to the same cluster, or a *cannot-link* where two instances belong to different clusters. Active constraint selection for clustering emphasizes reducing the number of pairwise constrains for the best clustering results. In addition, some active spectral clustering algorithms have been applied to both two-cluster cases by examining eigenvectors [121] or calculating entropies between pairwise samples [116], and applied to multi-cluster cases by purifying a kNN graph iteratively [120]. Unfortunately, for classification or clustering, utilization of pairwise correlation has been limited to either classification models or clustering models, respectively. It is important to combine both supervised and unsupervised models for consolidated predictions.

Ensemble learning emerges based on the theory that combining multiple base models can improve a single base model. Many classification ensemble [91, 97] and clustering ensemble [34, 73] methods exist to improve individual classifier (or cluster) results. Because unsupervised models can provide extra knowledge potentially useful for classification, some methods use clustering to improve classification ensemble [62, 43, 76], or combine both classification and clustering ensembles [1]. A notable work is proposed by Gao *et.al.* [43], which consolidates a classification solution by maximizing the consistency between supervised and unsupervised models. This work motivates our consensus ensemble learning, but the difference is significant: our model uses class distributions from the classification models to generate instance groups for label propagation, whereas [43] uses 0/1 loss to generate instance groups. So their instance groups are coarser and instances in each group may vary significantly.

Above solutions are on the basic assumption that the class number of the data is known or given beforehand. Given a labeling task with unknown classes and a labeler with weak knowledge, there is no labeled data available to kick-off the active learning process. We refers to this problem as **Cold-start** active learning. Fig. 6.1

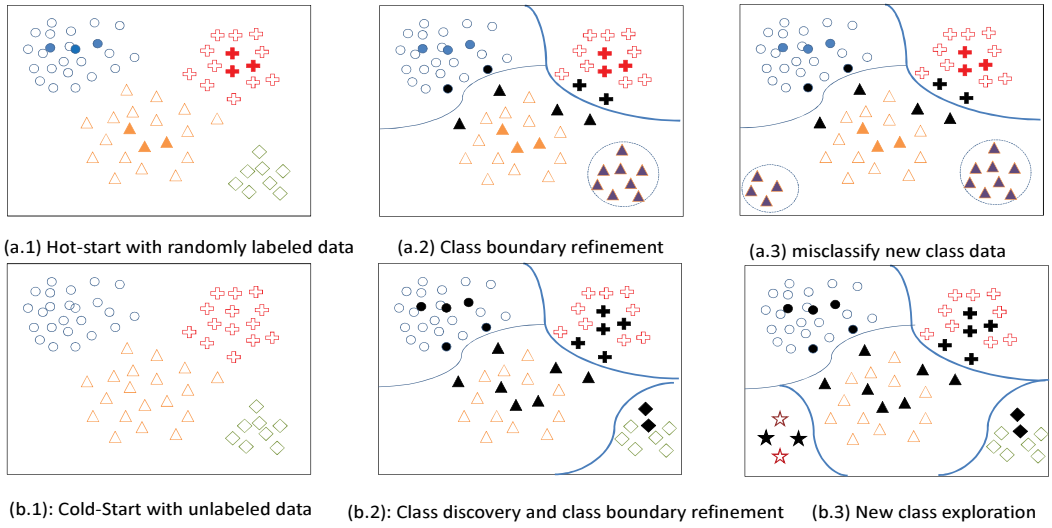


Figure 6.1: A conceptual view demonstrating the difference between Hot-start *vs.* Cold-Start active learning. Circles, pluses, and triangles each denote one type of samples. Solid symbols denote labeled instances and unfilled ones denote unlabeled instances. Diamonds and stars denote unknown and emerging new classes, respectively. Solid lines denote the decision boundaries learnt by the learner. (a) Hot-start active learning kicks-off with randomly labeled samples. It learns decision boundaries to partition samples in known class by querying the most uncertain data, whereas samples from undiscovered new classes will be misclassified (triangles circled in (a.2) and (a.3)); (b) Cold-start active learning starts without any label (b.1). It uses weak labeling knowledge to discover classes (b.2), explores new classes and label most uncertain instances at the same time (b.3).

illustrates the difference between **Hot-Start** active learning and our **Cold-start** scenario.

To address the above challenges, we propose to solve Cold-Start Active Learning (CSAL) by carrying out emerging class exploration and most informative instances simultaneously. More specifically, to discover initial classes (Challenge 1), we use a density-distance combined measure to select a representative subset of instances, and group them to form a forest with a Minimum Spanning Tree (MST) based query strategy. Therefore, our algorithm can discover initial classes with a very little labeling effort. To explore new emerging classes (Challenge 2) and label informative instances (Challenge 3), we propose a consensus ensemble learning framework to combine classifier ensemble and classifier ensemble, and adaptively discover new class instances and most informative data for labeling.

The problem definition is introduced in Section 6.2. The algorithm details are introduced in Section 6.3, followed by experiments in Section 6.4.

6.2 Problem Formulation

We consider a dynamic data environment, where a set of instances collected at a specific time t is denoted by $\mathcal{D}^t = \{x_1, ..., x_n\}$, and each instance $x_i \in \mathbb{R}^{q\times 1}$ is denoted by q features. The class label of x_i is denoted by y_i, which is unknown. The total class space is denoted by $\mathbb{L}$, which is also unknown and needs to be discovered and explored during the active learning process. After an arbitrary time interval Δt, a set of o new unlabeled instances, $D^{\Delta t} = \{x_{n+1}, \ldots, x_{n+o}\}$ are included into the data set. So the whole data set is made up of the previous data and new emerging data with $D^{t+\Delta t} = D^t \cup D^{\Delta t} = \{x_1, \ldots, x_n, x_{n+1}, \ldots, x_{n+o}\}$. Our **aims** are to: (1) identify all classes in D^t with minimum query efforts, (2) discover emerging classes in $D^{\Delta t}$ in real time, and (3) label the most informative instances to train an accurate classifier.

Instead of directly querying the class labels of individual instances, we consider a pairwise corrreltaion query setting, where the labeler only needs to answer whether a pair of instances (x_i, x_j) belong to the same class or not. This type of query is much easier to answer and requires less expertise than directly answering the actual class label of each single instance.

When given a set of n instances with pairwise correlation queries, the genuine class label of each instance can be determined by a maximum $\binom{n}{2}$ number of pairwise queries. In other words, we can use pairwise relationships between all instance pairs to determine each instance's class label and the number of classes in the data set. This approach is practically infeasible because it requires a large number of queries for large n values. To solve this challenge, we propose an active learning approach to select as few pairs as possible for class discovery and instance labeling.

6.3 CSAL: Cold-Start Active Learning

For a given data set without any labeling information, CSAL carries out active learning through two major steps: (1) *Class Discovery* tries to discover the initial (possible incomplete) classes in the data; and (2) *Class Exploration and Instance Labeling* focuses on exploring the missed classes in the initial *Class Discovery* process, and detecting new classes whenever they emerge. Meanwhile, this process also

labels most informative samples to refine the existing model. Fig. 6.2 illustrates the overall framework.

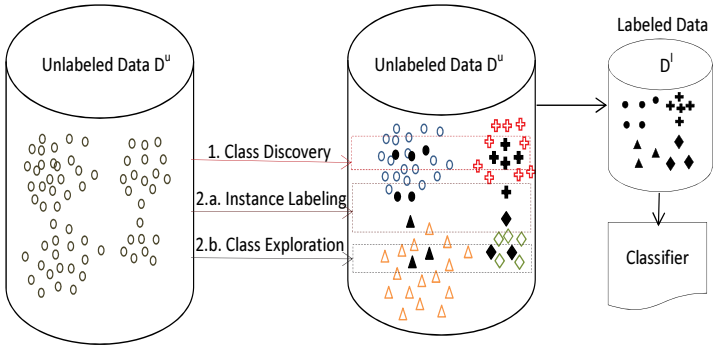


Figure 6.2: The overall framework of the proposed cold-start active learning. The randomly distributed circles in the left figure denote a given data set with no class label information. Circles, pluses , triangles, and diagnoses in the middle figure each denote one type of samples, with solid ones denoting labeled samples.

6.3.1 Class Discovery

Exhaustively querying pairwise instance relationships to determine the number of classes in D^U is expensive and out of question. To reduce the query costs, we propose to query the pairwise relationships on a small representative instance subset Γ. After the classes on Γ are discovered, we can directly determine the number of classes in the original unlabeled data set D^U, so the problem is transferred to a classification problem.

To save query costs, we can reduce the size of Γ to a very small number. This will negatively reduces the representability of Γ to represent the original data set D^U. Alternatively, we can select a reasonably small number for Γ, but avoid querying every pairs of instances in Γ. This is achieved by building minimum spanning trees (MST), and using the MST to query pairwise instance relationships, through which we can discover the initial classes in D^U, as illustrated in Fig. 6.3.

6.3.1.1 Representative Instance Subset Selection

In order to build a representative instance subset for the given data set D^U, we propose to select instances from D^U by taking the sample distributions into consideration. More specifically, (1) each instance in the representative subset should be

Figure 6.3: The class discovery process

selected from dense instance regions so an representative instance can help preserve the sample distributions in the original data set. In addition, (2) instances in the representative subset should be relatively far from each other, so the whole representative instance subset can maximally span the original instance space. Therefore, we combine *distance* and *density* to determine whether an instance should be selected to form the representative instance subset. The *distance* between a pair of instances x_i and x_j is calculated by using feature values between instances x_i and x_j. The *density* of an instance x_i is the average distance over all other instances in the unlabeled set D^U, which is described as follows:

$$density(x_i) = \frac{1}{|D^U|} \sum_{j=1}^{|D^u|} distance(x_i, x_j) \tag{6.1}$$

By combining instance-density measure, the representative instance subset selection is regarded as selecting a k-instance subset from an n data set, such that the selected subset has the maximum summation of the density and distance. To this end, we use a matrix $\mathbb{R} \in \mathbb{R}^{n \times n}$ to capture both density and distance information. Specifically, we assume that $\mathcal{Z}_{i,i}$ defines the density of instance x_i and $\Omega_{i,j}, i \neq j$ defines the distance between x_i and x_j. (In our experiments, we use Hamming and Euclidean distances for categorial and numerical features, respectively.) The matrix $\mathbb{R}$ is constructed using Eq.(6.2).

$$\mathbb{R}_{i,j} = \begin{cases} \mathcal{Z}_{i,j}, & \text{if } i = j \\ \Omega_{i,j}, & \text{if } i \neq j \end{cases} \tag{6.2}$$

Using the matrix $\mathbb{R}$, the representative instance subset selection can be formulated as a quadratic integer programming problem as follows:

$$\begin{aligned} &\max_{\mathbf{x}} \mathbf{x}^T \mathbb{R} \mathbf{x} \\ s.t. \sum_{i, x_i \in \mathbf{D}} x_i = k \; ; \quad &x_i \in \{0, 1\} \end{aligned} \tag{6.3}$$

where $\mathbf{x}$ is an n-dimensional column vector and n is the size of unlabeled data set D^U. The constraint k defines the size of the representative instance subset. $x_i = 1$ denotes that instance x_i is selected and $x_i = 0$ otherwise.

Assume that the objective function in Eq.(6.3) is properly solved; the representative instance subset Γ will contain k instances with the maximum summation of the density and distances among all alternative subsets with the same size. This problem is a standard 0-1 optimization problem, which is NP-hard in general. Fortunately, this formulation can be transformed to a max-cut with size k problem (MC-k), in which one partitions the vertices of an edge-weighted graph into two sets, one of which contains k vertices, so that the total weight of edges crossing the partition is maximized. The MC-k problem is known to have a very good approximate solution based on semi-definite programming (SDP). The key point of the SDP approximation algorithm is to relax each binary variable into a unit vector. Therefore, we use SDPA [19], which is based on an interior point method, to find solutions for representative subset selection.

6.3.1.2 MST Based Class Discovery

Given a representative subset Γ, we determine the number of classes and the class label for each instance by checking pairwise relationships between all instance pairs. This is still expensive and queries for many pairs are unnecessary, so we use Minimum Spanning Trees (MST) to minimize the number of instance pairs needed for the queries.

Our method begins with a tree consisting of a single vertex picked up randomly, which consists of an initial forest. At each time, we select the unvisited vertex x^* in Γ, which has the minimum distance to the forest, and query its pairwise relationships with a set formed by one vertex from each tree in the current forest. According to the query results, we determine whether to extend the current forest by building a

new tree, or include x^* into one of the existing trees in the forest. We continuously increase the number of visited vertices, one vertex at a time, until all vertices in Γ are visited. Because the vertices connected in the same tree have the same pairwise correlation with x^*, we only query the pairwise relationship between x^* and one vertex from each tree, which effectively reduces the query cost. Moreover, using the MST query strategy for class discovery, multiple minimum spanning trees are built in parallel. Finally, the representative instance subset is represented by a MST forest, where each MST denotes a group of vertices from a particular class. The pseudo-code in Algorithm 5 explains the general process of *Class Discovery*.

Algorithm 5 Class Discovery Process

Input: (1) an unlabeled sample set D^U; (2) the size of an optimal subset k.
Output: discover the classes in the optimal subset Γ.

$\mathbb{R} \leftarrow$ density-distance based matrix Eq.(6.2);
$\Gamma \leftarrow$ use SDP to solve Eq.(6.3);
$x_{init} \leftarrow$ a random instance in Γ;
$\mathfrak{T}_1 \leftarrow x_{init}$;
$\coprod_{\mathfrak{T}} \leftarrow \mathfrak{T}_1$; MST forest with one tree;
while not all vertices in Γ are visited **do**
    $x^* \leftarrow$ instance with minimum distance to $\coprod_{\mathfrak{T}}$;
    $\Psi \leftarrow \{x_1, \ldots, x_{|\coprod_{\mathfrak{T}}|}\}$, where $x_i \in \mathfrak{T}_i, (1 \leq i \leq |\coprod_{\mathfrak{T}}|)$ is one instance randomly selected from each tree;
    query (x^*, x_i), where $x_i \in \Psi, 1 \leq i \leq |\coprod_{\mathfrak{T}}|$;
    if exist $(x^*, x_i) \in$ "**Same Class**" **then**
        $\mathfrak{T}_i \leftarrow x^*$, where $x_i \in \mathfrak{T}_i$;
    else
        $\mathfrak{T}_{|\coprod_{\mathfrak{T}}|+1} \leftarrow x^*, \coprod_{\mathfrak{T}} \leftarrow \mathfrak{T}_{|\coprod_{\mathfrak{T}}|+1}$;
    end if
end while

6.3.2 Class Exploration and Instance Labeling

In the above class discovery process, the number of classes discovered might be incomplete. This is because: (1) the representative instance subset Γ may fail to cover instances from small classes, and (2) Some new classes might emerge when new samples are included into the data set. Therefore, we need to further explore the classes and instance labels by using two alternative steps: (a) build a "Consensus Ensemble Learning "model from both labeled and unlabeled data; and (b) use an adaptive query strategy for both new class sample exploration and informative

sample selection. The relationship between the two steps is illustrated in Fig. 6.4.

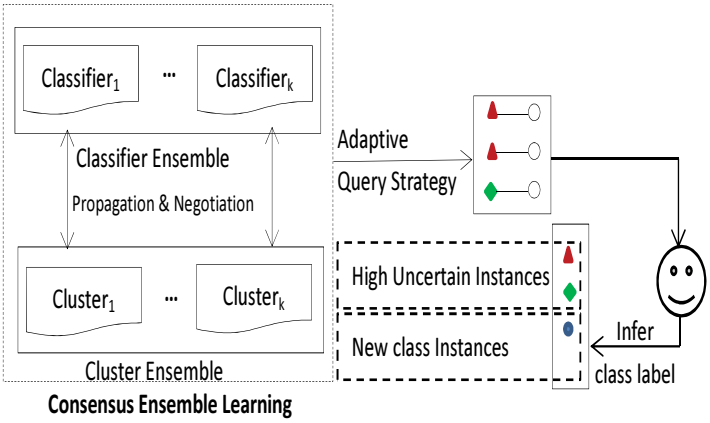


Figure 6.4: The two steps in class exploration and instance labeling.

6.3.2.1 Consensus Ensemble Learning

In dynamic data environments, samples in a new emerging class often have different distributions to the existing classes. Monitoring the changes in the data distributions helps to detect emerging classes. This can be achieved by using some clustering models to regularly check whether some samples are far from the existing data distributions. Some unlabeled samples from the known classes may also exist where the existing classification models are uncertain to classify them to a specific class (*i.e.* high uncertain instances). To leverage the strength of the classification and clustering models, we propose *Consensus Ensemble Learning* to combine classifier ensemble and cluster ensemble to determine whether an instance is possibly from a new class or from an existing known class. Because we have both labeled and unlabeled instances from the initial class discovery process, we can construct a classifier ensemble using β_s classifiers trained from representative instance subset Γ and a cluster ensemble using β_u clusters built from the unlabeled data set D^U.

Given a model set $\mathbb{E} = \{\upsilon_1, \ldots, \upsilon_{\beta_s}, \upsilon_{\beta_{s+1}}, \ldots, \upsilon_\beta\}$, $\beta = \beta_s + \beta_u$, the first β_s models are supervised (classifiers), and the rest are unsupervised (clusters). Consensus ensemble learning **aims** to use models in $\mathbb{E}$ to find a "consolidated" class label y^*

for each instance x in D^U. The objective function is formulated as:

$$\begin{aligned} y^\star &= \underset{y}{argmax}\, P(y|x,\mathbb{E}) \\ &= \underset{y}{argmax} \sum_{a=1}^{g} P(y|x,\hbar_a)P(\hbar_a|x) \\ &= \underset{y}{argmax} \sum_{a=1}^{g} P(y|x,\mathcal{G}_l^a)P(\hbar_a|x) \end{aligned} \tag{6.4}$$

Assume each model $\hbar_a$ divides data set D into several groups, with each instance x_i belonging to exactly one group. We have $P(y|x,\hbar_a) = P(y|x,\mathcal{G}_l^a)$, where $\mathcal{G}_l^a$ denotes the group l to which x is classified by model $\hbar_a$. For each unlabeled instance, we expect that its final predicted label is close to the (unknown) ground truth label. The challenges are therefore threefold, including: 1) how to create groups $\mathcal{G}_l^a$ used in the objective function; 2) how to calculate $P(y|x,\mathcal{G}_l^a)$ when $\hbar_a$ is an unsupervised model, in the case that the category of x cannot be obtained; and 3) how to weight the importance of each model $P(\hbar_a|x)$.

To solve the above challenges, our main idea is to calculate $P(y|x,\mathcal{G}_l^a)$ by propagating the prediction results from supervised to unsupervised models using a belief graph, where each node represents a group generated from each model. Meanwhile, a consistency scheme is used to approximate the model weight $P(\hbar_a|x)$. Different to an existing work [43], which uses a similar belief graph, our method uses: (1) *refined groups*: For each classification model, our method generates groups by using the class distributions from the classifier to merge instances into different groups; (2) *soft groups*: for each group, our method uses the mean class distribution of instances in each cluster as the group's initial label distribution; and (3) *arbitrary group numbers*: our method generates an arbitrary number of groups for each model, so the belief propagation can be fine-grained at different levels.

A. Group Generation: For each *supervised* model $\hbar_a^s$, we denote the prediction distribution of an instance x_j by a vector $\Phi_j = (\phi_a^1, \ldots, \phi_a^{|Y|})$, where $|Y|$ is the number of discovered classes in the total class space $\mathbb{Y}$. ϕ_a^i denotes the probability of x_j belonging to class i given $\hbar_a^s$, that is $\widetilde{P}(y_i|x_j,\hbar_a^s)$. To generate instance groups based on each supervised model's prediction, we use each instance x_j's distribution

vector Φ_j as the input to a clustering algorithm (we use $k-$means in our experiments), and arbitrarily generate κ^a clusters $\mathcal{G}_l^a(1 \leq l \leq \kappa^a), \kappa^a > |Y|$ for each supervised model $\hbar_a^s$. The initial labeling information of each group $\mathcal{G}_l^a$ is indicated by a label distribution vector $\widetilde{P}(\mathcal{G}_l^a) = (\widetilde{P}(y_1|x, \mathcal{G}_l^a), \ldots, \widetilde{P}(y_{|Y|}|x, \mathcal{G}_l^a))$, where $\widetilde{P}(y_i|x, \mathcal{G}_l^a), (1 \leq i \leq |Y|)$ is the mean of $\widetilde{P}(y_i|x_j, \hbar_a^s)$ for each item x_j in $\mathcal{G}_l^a$, which is formulated as follows:

$$\widetilde{P}(y_i|x, \mathcal{G}_l^a) = \frac{1}{|\mathcal{G}_l^a|} \sum_{x_j \in \mathcal{G}_l^a} \widetilde{P}(y_i|x_j, \hbar_a^s) \tag{6.5}$$

For each *unsupervised* model $\hbar_a^u$, its groups are generated by applying a clustering algorithm to the original unlabeled data set D^U, and arbitrarily generating κ^a clusters $\mathcal{G}_l^a(1 \leq l \leq \kappa^a)$, which is regarded as the outcome of $\hbar_a^u$. Because clusters do not have explicit class labels, the corresponding initial label information vector $\widetilde{P}(\mathcal{G}_l^a)$ is set as 0s.

B. Model Probability Propagation and Negotiation: Using groups generated from supervised and unsupervised models, we can build a belief graph $G = (V, \mathfrak{E})$. Each node in V is a group $\mathcal{G}_i$ from a model $\hbar_a$, and each edge in $\mathfrak{E}$ connecting two nodes $\mathcal{G}_i$ and $\mathcal{G}_j$ is weighted by the similarity between these two groups using Jaccard coefficient:

$$J(\mathcal{G}_i, \mathcal{G}_j) = \frac{n_{i,j}}{n_i + n_j - n_{i,j}} \tag{6.6}$$

where n_i, n_j, and $n_{i,j}$ are the numbers of instances in $\mathcal{G}_i$, $\mathcal{G}_j$, and in both $\mathcal{G}_i$ and $\mathcal{G}_j$, respectively. Fig. 6.5 show an example of a constructed graph. Each of the seven groups from supervised models $\hbar_1$ (4 groups) and $\hbar_2$ (3 groups) (the black nodes) has its initial labeling distribution vector; whereas each of the nine groups from unsupervised models $\hbar_3$ (5 groups) and $\hbar_4$ (4 groups)(the white nodes) is unlabeled, with zero label distribution vector. Through this graph, we try to iteratively propagate the labeling distribution information from groups in supervised models to groups in the unsupervised models until convergence, so that the final $\widehat{P}(y|x, \mathcal{G}_i), (1 \leq i \leq 16)$ represents the results of negotiation among all the modes with the prior knowledge from supervised models.

Now we detail the propagation method. We use a matrix $\mathbb{B}_{\Lambda * |Y|}$ to denote the conditional probability we aim for, where $\Lambda = \sum_{a=1}^{f} \kappa^a$ denotes the total groups

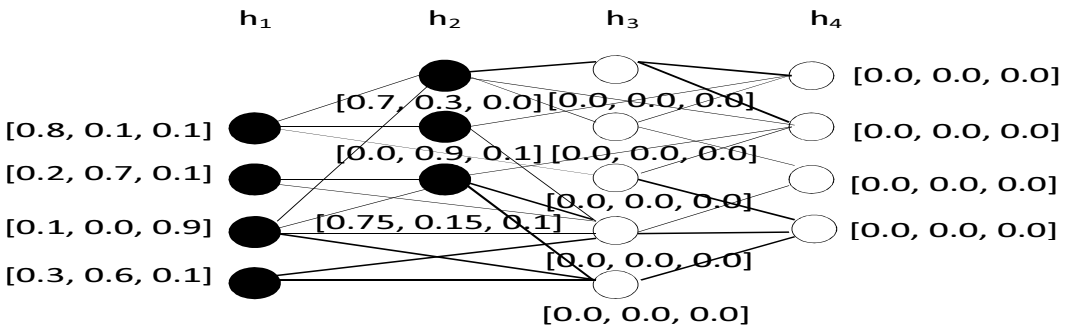


Figure 6.5: Illustration of the brief graph.

generated by the "consensus learning model "$\mathbb{E}$, $|Y|$ is the number of discovered classes so far. The element $\mathbb{B}_{iq} = \widehat{P}(y = q|x, \mathcal{G}_i)$. Initially $\mathbb{B}$ is a zero matrix. Meanwhile, we define another matrix $\mathbb{Q}_{\Lambda*|Y|}$ that records initial labeling information from supervised models, where $\mathbb{Q}_{iq} = \widetilde{P}(y = q|x, \mathcal{G}_i)$ if $\mathcal{G}_i$ is from a supervised model, and 0 otherwise. In other words, all entries corresponding to groups from supervised models have related conditional probability values, whereas all entries corresponding to groups from unsupervised models are 0s. Moreover, a similarity matrix $\mathcal{S}_{\Lambda*\Lambda}$ is constructed to store edge weights between any two groups in the graph, with each element $\mathcal{S}_{ij}, i \neq j$ denoting $J(\mathcal{G}_i, \mathcal{G}_j)$ as defined in Eq.(6.6). We construct a diagonal matrix O with its diagonal element O_{kk} equal to the sum of the kth row of $\mathcal{S}$. Let $\mathbb{H}$ be a normalization form of $\mathcal{S}$, which is computed with Eq.(6.9)

The aim of propagation is formulated as the minimum of the following objective function:

$$\frac{1}{2}\sum_{k,j=1}^{\Lambda}\mathcal{S}_{kj}\sum_{q=1}^{w}(\frac{1}{\sqrt{O_{kk}}}\mathbb{B}_{kq} - \frac{1}{\sqrt{O_{jj}}}\mathbb{B}_{jq})^2 + \mu\sum_{k=1}^{\Lambda}\sum_{q=1}^{w}(\mathbb{B}_{kq} - \mathbb{Q}_{kq})^2 \tag{6.7}$$

In this objective function, the first term evaluates the difference between the labels of two groups $\mathcal{G}_k$ and $\mathcal{G}_j$. The more items are overlapped in both two groups, the higher their similarity $\mathcal{S}_{kj}$ is. The second term penalizes the deviation from the initial label assignments for the groups from supervised models. We construct the normalized graph laplacian as $\varsigma = O^{1/2}(O - \mathcal{S})O^{1/2} = I - \mathbb{H}$. According to the properties of graph laplacians, the above objective function is transformed as:

$$\mathbb{B}^T\varsigma\mathbb{B} + \mu(\mathbb{B} - \mathbb{Q})^T(\mathbb{B} - \mathbb{Q}) \tag{6.8}$$

To derive the optimal solution, we differentiate the objective function in Eq.6.8 with respect to $\mathbb{B}$, we can get: $\mathbb{B}^* - \mathbb{H}\mathbb{B}^* + \mu(\mathbb{B}^* - \mathbb{Q}) = 0$. Let $\alpha = \frac{1}{\mu+1}$,we can get $\mathbb{B}^* = (1-\alpha)(I - \alpha\mathbb{H})^{-1}\mathbb{Q}$. To avoid computing a matrix inverse, we compute $\mathbb{B}$ in an iterative way where $\mathbb{B} = \frac{1}{\mu+1}(\mathbb{H}\mathbb{B} + \mu\mathbb{Q}) = \alpha\mathbb{H}\mathbb{B} + (1-\alpha)\mathbb{Q}$.

$$\mathbb{H} = O^{-1/2}\mathcal{S}O^{-1/2} \tag{6.9}$$

Then we iteratively compute Eq.(6.9) until the algorithm converges, where α is a control parameter weighting the effect of initial labeling from supervised models. Finally, we normalize $\mathbb{Q}$ so that the sum of each row of $\mathbb{Q}$ is equal to 1.

$$\mathbb{B} = \alpha\mathbb{H}\mathbb{B} + (1-\alpha)\mathbb{Q} \tag{6.10}$$

C. Model Weights: After the estimation of the conditional probability of each model $\hbar_a$, we need to determine the importance of each model's prediction in the final solution, with a local model weight $P(\hbar_a|x)$ that reflects the prediction ability of $\hbar_a$ on x. If its prediction on x is close to the genuine class probability $P(y|x)$, the model should have a high weight value. Unfortunately, the genuine class probability $P(y|x)$ is unknown a priori, so we have to use an alternative to approximate the model weight. From the consensus perspective, we assume that the model more consistent with others's predictions on x is more reliable and therefore, has a higher weight. Accordingly, we utilize the prediction consistency between models to approximate the model weight:

$$P(\hbar_a|x) \propto \frac{1}{\beta}\sum_{b=1,b\neq a}^{\beta}\delta(\hbar_a,\hbar_b|x) \tag{6.11}$$

$\mathcal{C}(\hbar_a,\upsilon_b|x)$ denotes the similarity between $\hbar_a$ and $\hbar_b$ on x's label prediction. Similarly, $\frac{1}{\beta}\sum_{b=1,b\neq a}^{\beta}\mathcal{C}(\hbar_a,\upsilon_b|x)$ measures the average model consistency between $\hbar_a$ and all other models' prediction on x. Suppose the sets of instances that are in the same group with x in $\hbar_a$ and $\hbar_b$ are X^a and X^b separately, the pairwise model consistency can be calculated by

$$\delta(\hbar_a,\hbar_b|x) \propto \frac{|X^a \bigcap X^b|}{|X^a \bigcup X^b|} \tag{6.12}$$

In Eq.(6.12), $\hbar_a$ and $\hbar_b$ can be either a supervised or an unsupervised model. We can infer pairwise prediction consistency between $\hbar_a$ and $\hbar_b$ on x by using x's

neighbors according to the grouping results of both models. Meanwhile, because a single prediction tends to have the highest consistency, we add another term to the model weight definition, representing this minority accurate prediction:

$$P(\hbar_a|x) \propto (1-\beta)\frac{1}{\beta}\sum_{b=1,b\neq a}^{\beta}\delta(\hbar_a,\hbar_b|x)+\beta\frac{1}{\beta} \tag{6.13}$$

The above computation about the local model weight includes two components: (1) a consensus model component that measures the local consensus between models, and (2) a random model component that has no preference to any model. β is a parameter to adjust the ratio between two components.

6.3.2.2 Adaptive Query Strategy

Because the initial class discovery process may miss some classes, an adaptive query strategy is involved to: (1) label most uncertain instances for decision boundary refinement, and (2) detect new class samples whenever new classes emerge. In cold start active learning environments, calculating the probability of an instance belonging to an unknown class is challenging. In particular, the new class may appear at any time and there is no clear differentiation between new class samples and uncertain instances belonging to the known classes. To resolve these issues, we propose an adaptive query strategy using the *Dirichlet* process, which can describe the distribution of infinity number of classes as a *Dirichlet* distribution. The typical *Dirichlet* process is denoted by $DP(\theta,\eta)$, where θ is a *concentration parameter* to control the probability of x from a new class. η is its *base measure*, denoting the total label space $\mathbb{Y}$.

For each instance x in the unlabeled data set, we calculate its probability of belonging to an existing class and its probability of belonging to a new class y^{new}. The probability distribution for an instance x is given as

$$P_t(y\in Y\cup y_{new}|x)\propto\begin{cases}\frac{c_y}{\sum_{k\in Y}c_k+\theta}P_{error}(x|y), & if \quad y\in Y\\ \frac{\theta}{\sum_{k\in Y}c_k+\theta}P(x), & if \quad y=y_{new}\end{cases} \tag{6.14}$$

where L is the set of the known classes so far, c_k is the number of instances labeled with the class y_k and θ is the concentration parameter of the *Dirichlet* process. To compute the probability with Eq.(6.14), we need to set the prior $P(x)$ and calculate conditional probability $P_{error}(x|y)$. A prior $P(x)$ should reflect an instance's

usability and avoid the sparse outlier occurring with a high prior probability, so we estimate it with the instance x's density. The posterior probability $P(y|x)$ and $P(y)$ are based on the predictions from "Consensus Ensemble Learning". Therefore, we can calculate $P_{error}(x|y) = \frac{P(y|x)P(x)}{P(y)}$ by using the Bayes rule.

Given the class label distribution for an instance $x, P_t(y \in Y \cup y_{new}|x)$, we calculate the probability of incorrectly classifying an instance with Eq.(6.15)

$$P^m(x) = 1 - P_t(y|x), \qquad y = \underset{y \in Y}{argmax}\, P_{error}(y|x) \tag{6.15}$$

Compared with traditional active learning query strategies, the proposed query strategy will balance the goal of finding new classes *vs.* refining existing classes, rather than simply focusing on the refinement for existing classes. The main reason for this is that $P_t(y|x)$ contains the probability of an instance from a new class $P(y_{new}|x)$. If $P(y_{new}|x)$ is high, $P^m(x)$ is also high. Similarly, if $P(y_{new}|x)$ is low but the classifier is uncertain, then none of the class probabilities in $P_t(y|x)$ is high, so the value of $P(y_{new}|x)$ is still very high. As a result, the misclassification probability $P^m(x)$ is determined by two factors: the probability of an instance belonging to a new class and the uncertainty of the current model's prediction on x.

The general process of class exploration and instance labeling is illustrated in Algorithm 6.

6.3.3 Time Complexity Analysis

The total time complexity of CSAL includes two major parts: 1) class discovery 2) class exploration and instance labeling. In the first part, we assume that the unlabeled data set contains n instances, the calculation of matrix $\mathbb{R}$ representing both instance distance and density information requires O(n^2) time complexity. Thus the SDP process requires O(n^2)complexity to solve the $n \times n$ matrix and selects a k-instance subset. In order to identify the class information on the selected representative subset, a MST based query strategy needs O(k) time complexity (for the best condition) and O(k^2) time complexity (for the worst condition). Because the size of the instance subset k is much smaller than the size of the entire unlabeled data set n, we can regard that class discovery s time complexity is bounded by

Algorithm 6 Class Exploration and Instance Labeling

Input: (1) D^U: an unlabeled data set; (2) $|Y|$: number of classes discovered so far; (3) Γ: Representative instance subset.

Output: a labeled instance subset D^L for training an accurate classifier.

$D^L \leftarrow \Gamma$;
while $|D^L| < budget$ **do**
  $\mathbb{E}^s \leftarrow$ train a classifier ensemble from D^L;
  $\mathbb{E}^u \leftarrow$ train a cluster ensemble from D^U;
  $\mathbb{E} \leftarrow \mathbb{E}^s \bigcup \mathbb{E}^u$;
  $E_{\mathcal{G}} \leftarrow \{\mathcal{G}_1, \mathcal{G}_2, \ldots, \mathcal{G}_\beta\}$ groups with initial class distribution Eq.(6.5);
  $G \leftarrow \langle V, E \rangle$, where $V = \mathbb{G}, E_{k,j} \leftarrow$Group Similarity Eq.(6.6).
  $\widehat{P}(y = q|x, \mathcal{G}_k) \leftarrow$Conditional model probability Eq.(6.10);
  $P(\hbar_a|x) \leftarrow$ Model weight Eq.(6.13);
  $y_x^\star \leftarrow$ Most probable class Eq.(6.4);
  $P^w(x) \leftarrow$ Misclassification probability Eq.(6.15);
  $\Theta \leftarrow$ Instances with high $P^w(x)$;
  $\Psi \leftarrow \{(x_1, y_1), \ldots, (x_{|L|}, y_{|L|}\}$, where $(x_i, y_i) \in D^L$ is one instance randomly selected from each class y_i
  Query (x_u, x_v), $x_u \in \Theta, (1 \leq u \leq |\Theta|)$, and $x_v \in \Psi, (1 \leq v \leq |\Psi|)$;
  if exist $(x_u, x_v) \in$ "**Same Class**" **then**
    $y(x_u) \leftarrow y(x_v)$;
  else
    $|Y|$=$|Y|$+1, $y(x) \leftarrow y_{new}$;
  end if
  $D^L \leftarrow D^L \cup \Theta$;
end while

$O(n^2)$.

In the process of class exploration and instance labeling, the whole process requires the repetitive training the consensus ensemble model. Suppose the while loop between steps 2 and 21 in Algorithm 6 needs to repeat ς times; the size of a most utility subset added in each loop is r; and the feature dimension of the data set is d. A β_s-J48 classifier ensemble construction needs $O(\beta_s(k + (\varsigma - 1) \times rd^2)))$time complexity. While building a β_u-Kmeans cluster ensemble requires $O(\beta_u n^{dw+1} logn)$ time complexity, where w is the number of groups generated by each cluster. We assume that the uncovered classes at current iteration is c. We further use Kmeans algorithm to generate several groups based on the prediction results on each classifier, which needs $O(\beta_s n^{cw+1} logn)$. In general, $c \ll d$, we roughly regard the complexity in terms of model construction of consensus ensemble learning is bounded by $O(d^2)$+$O(n^{dw+1} logn)$.

After building a consensus ensemble learning model, our algorithm predicts the class labels of unlabeled instances through two sub-steps: (1) conditional probability propagation (2) model weight estimation. In the first sub-step, a brief graph is built with these groups generated by the classifier ensemble and the cluster ensemble. We need to calculate the conditional probability of each group through propagation from supervised nodes to unsupervised nodes. The total number of groups is e, and each group can be represented using a binary vector, so the time to construct and normalize the similarity matrix $\mathcal{S}$ is simply $O(\Lambda^2)$. Suppose we have r iterations, then the propagation time is $O(rc\Lambda^2)$ where c is the number of classes. The normalization on the prediction results takes O(e). The time of the second step is mainly attributed to the computation of pairwise local consistency. We have $\frac{\Lambda(\Lambda-1)}{2}$ pairs of groups and we only need to calculate the pairwise local consistency for these pairs of groups with the time complexity of $O(\Lambda^2)$. Due to we work at the level of groups instead of instance, and usually the number of groups are quite smaller than the number of instances in the data set. Therefore, the computation is independent of the number of examples , with a time complexity of $O(\Lambda^2)$.

To adaptively select new class samples and most utility data, we need to compute the misclassification probability for each unlabeled instance with Eq.(6.15). Accordingly, the calculation of misclassification probability requires O(n) time complexity.

To this end, the total time complexity of our algorithm CSAL is given as follows:

$$O(CSAL) = O(n^2) + O(d^2) + O(n^{dw+1}logn) + O(\Lambda^2) + O(n) \tag{6.16}$$

The aforementioned complexity analysis indicates that the bottleneck time complexity of CSAL is asymptotically bounded by the process of building consensus ensemble learning model.

6.4 Experiments

We implement CSAL and several baseline approaches using Java and WEKA data mining tools, and comparatively study their performances on ten benchmark data sets, as shown in Table 6.1. A simple description of the benchmark data sets is summarized in Table 6.1. For fair comparisons, all experiments are reported based on 10 times 10-fold cross validation and all methods are compared based on the same training and test data sets (the initial randomly labeled samples are also the same for all methods). To build a "consensus ensemble learning "model, we train four decision tree classifiers (using WEKA J48 implementation) from bootstrap sample sets generated from the labeled set, and build four $k-$means clustering solutions on the whole unlabeled data set. At each iteration, the number of groups for each model in the "consensus ensemble learning "is a random number ranging from the number of classes discovered so far to twice of this number.

Table 6.1: A simple description of the benchmark data

ID	Dataset	Instances	Features	Classes
1	segement-challenge (sc)	1500	20	7
2	poker	25010	11	10
3	covertype	5000	10	7
4	letter recognition(lr)	20000	16	26
5	glass	214	10	6
6	vowel	990	10	11
7	MNIST	60000	50	10
8	fbis.wc	2463	2001	17
9	ERA	1000	5	19
10	yeast	1484	8	10

6.4.1 Baseline Methods.

To demonstrate the effectiveness of the proposed approach, we compare our CSAL with the following baseline methods.

- **Supervised Ensemble learning with Adaptive Query strategy**(SEAQ) is a classifier ensemble, formed by using four decision trees trained from bootstrap samples of the labeled data. SEAQ uses our adaptive query strategy to query a subset of the most informative instances and includes them in the labeled set. The classifier ensemble is updated with the new labeled data iteratively until it reaches the user's requirements.

- **Consensus Ensemble Learning with Random Sampling strategy**(ECRS) has the same framework as *CSAL*, except that: (1) ECRS uses a random strategy to generate an initial representative instance subset (whereas CSAL uses a density-distance selection criterion), and (2) ECRS randomly selects unlabeled instances for labeling, rather than using a more useful utility metric like *CSAL* does.

- **Supervised Ensemble learning with Random Sampling strategy**(SERS) has the same framework as *SEAQ*, except that it uses a random sampling method.

- **Consensus Ensemble Learning with Entropy Measure**(ECEM) has the same framework as *CSAL*, except that this method uses *entropy*, a commonly used query strategy in traditional active learning, as a utility metric for instance selection.

- **Active Learning with Generative and Discriminative Models**(ALGD)[56] is a combined classifier model that uses generative and discriminative models to discover rare classes during the active learning process. By adapting two query strategies online, it aims to choose new class samples and most uncertain data for classification refinement.

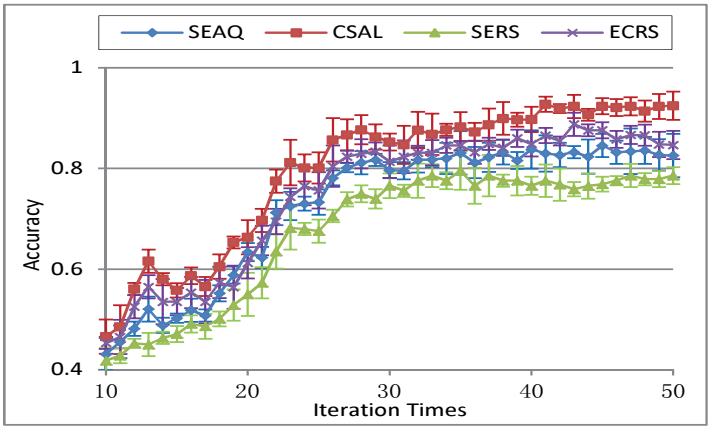
In the following sections, we study algorithm performance with respect to accuracy gains, the number of queries required to discover new classes, and new emerging class discovery results.

6.4.2 Accuracy Gains with Different Ensemble Methods

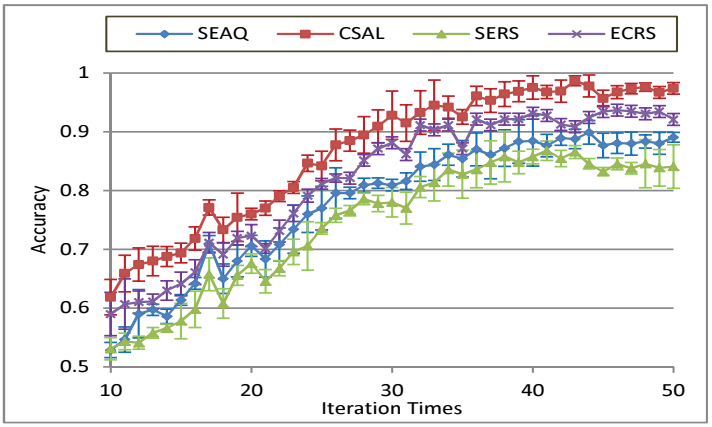
In Fig. 6.6, we compare the algorithm performance with respect to different ensemble methods. Two consensus ensemble learning methods, *CSAL* and *ECRS*, are built on the same framework but differ in adaptive query strategy (CSAL) and random query strategy (ECRS), as do the other two supervised ensemble methods *SEAQ* and *SERS*. At each iteration, we query a subset that includes 1% of the most utility data from the unlabeled pool. As the iteration time increases, the performances of all methods continuously improve. This observation suggests that the pairwise query information does help improve model effectiveness, regardless of the query strategy used in the active learning process. It is clear that the proposed *CSAL* performs best among all baseline methods on the three data sets. Although *ECRS* is a consensus learning model, its performance is interior to *CSAL*. We assert that pairs selected by using the adaptive strategy is more critical than the pairs selected at random. For the supervised ensemble, *SEAQ* with an adaptive query strategy also outperforms *SERS*, which uses a random query strategy, because our adaptive strategy balances the goal of discovering the new class and finding uncertain data to refine decision boundaries, which helps improve the model performance. Furthermore, using the same query strategy, ensemble methods (*CSAL*,*ECRS*) outperform supervised ensemble methods (*SEAQ*,*SERS*). This observation suggests that the combination of supervised and unsupervised models can indeed boost the model performance.

6.4.3 Class Discovery using Different Query Strategies

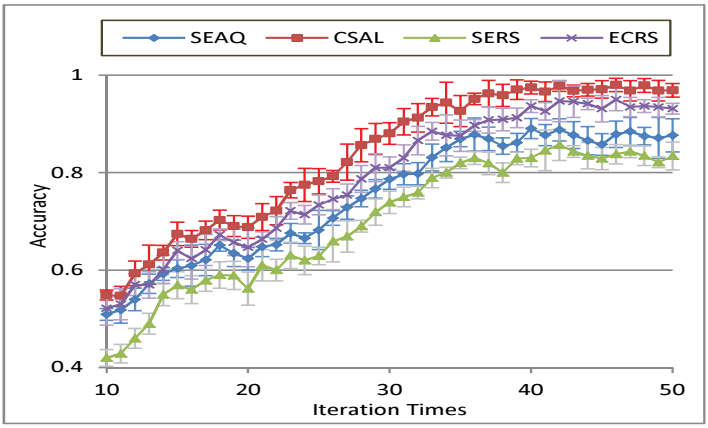
In Fig. 6.7, we report the query numbers with respect to different query strategies for class discovery. These methods are all built based on pairwise relationship queries, including a random selection in *RanPQC*, a maximum distance selection in *MaxSTC*, and a minimum distance selection in *MinSTC*. We apply these methods to help discover classes based on the representative instance subset built using the density-distance metric. In our implementation, *MaxSTC* has a similar framework with our *MinSTC*, which starts with a single vertex, and continuously increases the size of a tree, one edge at a time, until it spans all vertices. In contrast to *MinSTC*,



(a) segment-challenge1



(b) vowel



(c) ERA

Figure 6.6: Accuracy comparisons with different ensemble methods

the vertex having the longest distance to the existing tree vertices is visited each time. It also queries the pairwise corrreltaion between the added vertex and the existing vertices in the trees to determine whether to build a new tree or include the instance into one of the existing trees. *RanPQC* randomly selects a pair of instances to query its label relationship at each time until the number of classes in the representative subset is identified.

For all benchmark data sets, *MaxSTC* and *MinSTC*, which share the same framework, are always superior to *RanPQC*. This observation suggests that using a heuristic strategy in pair query effectively reduces the query cost. Moreover, *MinSTC* performs better than *MaxSTC*. This is because the two instances near in the topology are more similar, with a high probability of belonging to the same class. When selecting the instance by using *MinSTC* strategy, a selected instance x is more likely from one of the existing groups, so only a very few queries are needed to validate whether x is from an existing class, or whether it is from a new class. On the other hand, an instance x selected by *MaxSTC* is dissimilar to the existing trees. So one has to query the pairwise relationships between x and all of the existing trees, and then starts to build a new tree (if x does not belong to any of the existing class).

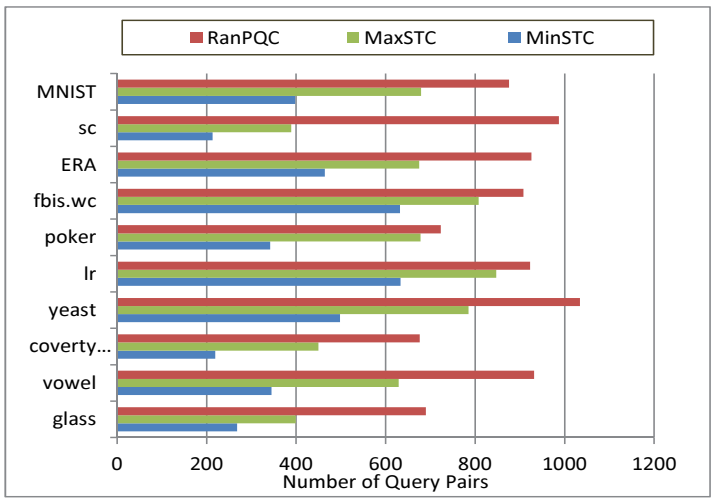


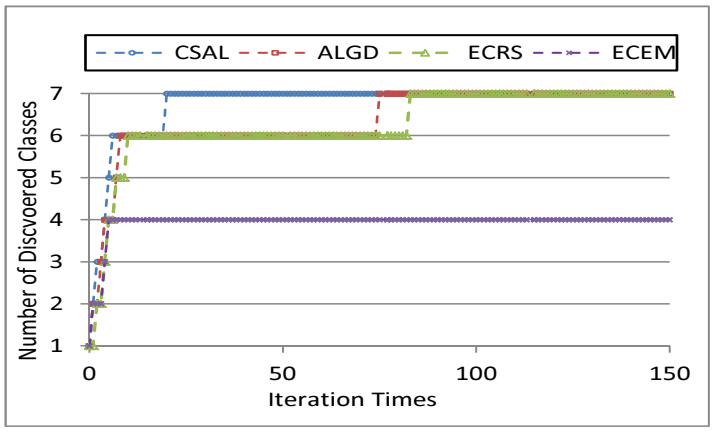
Figure 6.7: query number comparison with different query strategies .

6.4.4 Class Exploration Comparisons

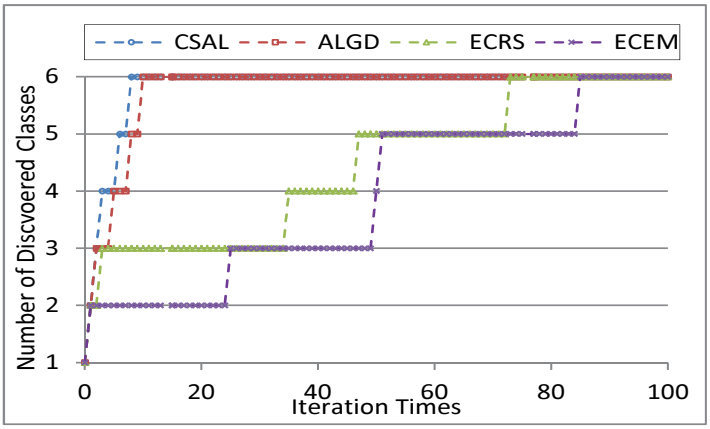
In this subsection, we compare the number of classes discovered by different class discovery methods, including CSAL, ALGD, ECRS and ECEM. In our experiments, the initial labeling set contains only one known class. We compare the number of discovered classes with respect to the iteration times and report the results in Fig. 6.8. The results show that CSAL has the least number of iterations to find all the classes in *yeast*, *covertype*, and *glass* data sets, which confirms that the adaptive query strategy used in CSAL effectively reduces the number of iterations needed to discover all classes in each data set. In the *yeast* data set, CSAL performs better than ALGD because, although ALGD uses two respective query criteria for class discovery and class boundary refinement, it only selects one query strategy with the highest weight at each iteration. For CSAL, it simultaneously considers the probability of the queried data belonging to one of the existing classes (for boundary refinement) and the probability of belonging to a new class. We also note that ECEM, which solely uses an uncertainty metric, is always inferior for class discovery. ECEM only finds four and eight classes in the *covertype* and *yeast* data set, respectively. This suggests that the uncertainty measure falls short in detecting new class samples that often have different distributions from existing known classes. Meanwhile, ECRS employs a random query strategy and is always interior to the two algorithms CSAL and ALGD, which use a heuristic query measure. However, the ECRS performance has a big margin over the uncertainty measure based method ECEM on the *covertype* and *glass* data sets. We assert that the uncertain measure, which prefers data close to the decision boundaries of existing models, is not helpful for new class discovery.

6.4.5 Experiment Results with Dynamic Data

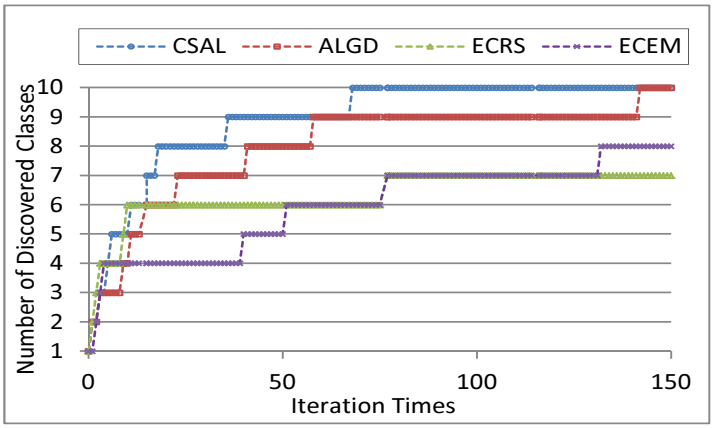
Iteration times Comparisons for Class Exploration To simulate dynamic data settings, we add a discrete time attribute to the following three data sets - *covertype*, *glass*, and *MINST*. We sort instances in each data set according to their class label, so that the indexes of instances from the same category are next to each other. Given a p-class data set, we assign the initial time value t_i to the data



(a) covertype



(b) glass



(c) yeast

Figure 6.8: Iteration comparison with regard to Class Discovery

from the first two classes y_1 and y_2. We then increase the time value with the same step (15 min) for each of the following classes. As a result, the time values for the data from y_3 to y_p range from $t_i + 15$ to $t_i + (p-2) * 15$. To demonstrate different methods' performance in detecting new classes, we report the results by using the number of iterations for new class detection (only one instance pair is queried in each iteration). Meanwhile, all experiments follow the same setting as follows: the initial labeling sets are constructed from one class with the same size and the total number of classes in the initial unlabeled data set is two. After every 10 queries, we add a new class to the unlabeled data pool according to the time value. We compare the performance of CSAL with ALGD , ECRS and ECEM and report their results in Fig. 6.9, which records the iteration times with respects to the new class discovered in a dynamic environment. Table 6.2 reports the finial accuracy after adding all new classes in the data set.

In Fig. 6.9, the solid lines denote the genuine number of classes in the data set at different iterations. The dashed lines with different maker points represent the performances of different methods. For the *MNIST* data set, ALGD is a winner for class exploration, with the proposed CSAL following in second position. For the *glass* data set, our method takes the first place. While for the *covertype* data set, the performances of our method and ALGD are similar. Note that both ALGD and CSAL explore the new class in the same stair step when a new class emerges, because these two heuristic methods both combine the goals of new class discovery and boundary refinement in the query process. However, finding a class quickly does not mean its classification performance is good. Although ALGD performs slightly better than CSAL, the proposed method CSAL always has the highest accuracy in the three data sets as shown in Table 6.2. When considering performances in terms of concurrent timeliness and accuracy, our method CSAL is the most competitive amongst all the baseline methods. Moreover, the accuracies of both ALGD and C-SAL show a significant improvement over the random metric based ECRS and the uncertainty metric based ECEM. Therefore, we assert that focusing on both class exploration and boundary refinement significantly boosts the classification performance than only focusing on boundary refinement. Meanwhile, due to its inherent random query strategy, ECRS has an unstable performance. ECEM employs an

uncertainty metric but it has the lowest class discovery rate and accuracy, which demonstrates that uncertainty is not very useful for class exploration.

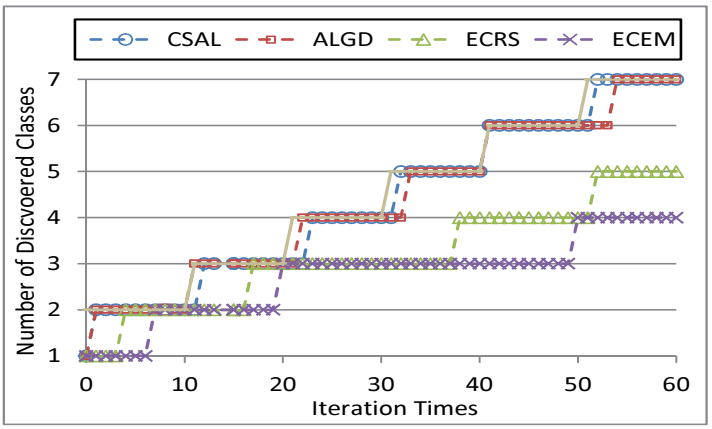
Table 6.2: accuracy comparisons with different methods

Dataset	CSAL	ALGD	ECRS	ECEM
covertype	49.56%	44.87%	36.87%	30.56%
glass	69.87%	66.39%	61.56%	50.12%
MNIST	56.8%	54.6%	50.5%	48.6%

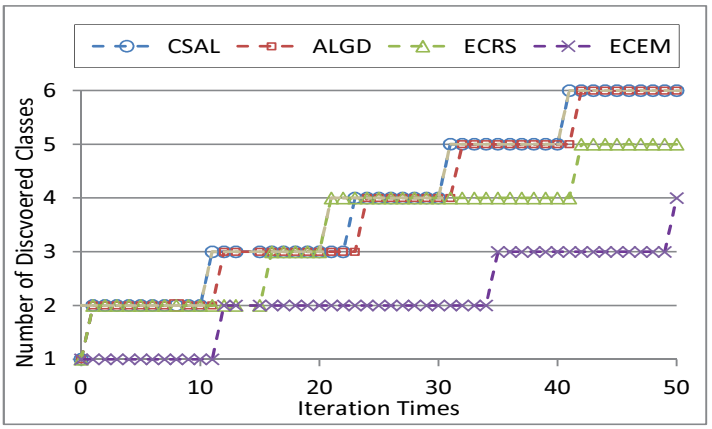
Visualization of the Results: To visually demonstrate the class detection results, we apply CSAL to a newsgroup data set with 20 classes. This data set has 400 examples (each has 1437 feature) representing news articles emerging in an hour (from 9:00AM to 10:00AM). To simplify the time concept, we transform the continual time value to four discrete time stamps with news having one unique time stamp in {9:00am, 9:20am, 9:40am ,10:00am}. To demonstrate our adaptive query process, Fig. 6.10 reports some illustrative decisions made by our model during the first eight iterations of a typical run. The rounded rectangles with different colors at the right side of the dashed line denote the classes existing in a specific time interval. In the first time interval between 9:00AM and 9:20AM, the new class example in class "alt.athesim" is selected at the first iteration. Additionally, in the remaining time slots, the new class samples are captured in the same time slot as they appear. This observation validates the feature of new class discovery in our adaptive query strategy. Meanwhile, at the second iteration, our adaptive measure selects the most dissimilar news to the initial training example from the same category "comp.graphics"; this also occur for the fifth and the eighth iterations, because the classifier learnt from labeled data is uncertain about the instances. This observation asserts that our adaptive measure helps to find uncertain instances for labeling.

6.4.6 Detailed Comparisons for All Methods

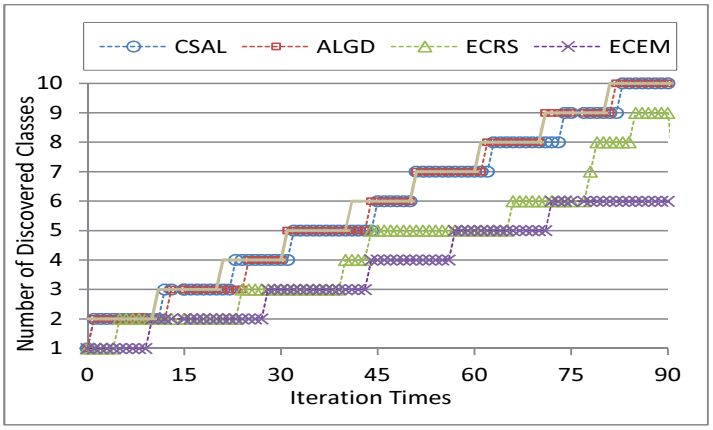
In Table 6.3, we report detailed comparisons of all methods on 10 benchmark data sets. In all the data sets, the initial labeling set contains only one known class, the iteration number is fixed to 150, with one most informative instance selected at each iteration. Among all methods, CSAL achieves the best performance gain, ALGD are



(a) covertype



(b) glass



(c) MNIST

Figure 6.9: Iteration comparisons with regard to class exploration in a dynamic environment

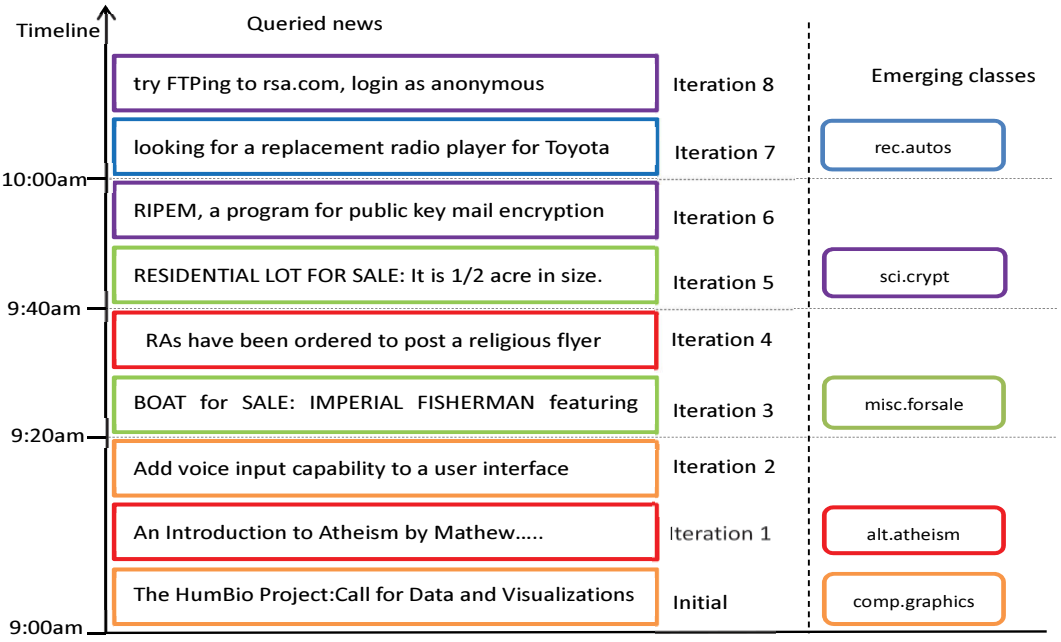


Figure 6.10: Illustration of the brief graph.

Table 6.3: Detailed algorithm performance comparisons (iteration number is 150)

Dataset	SERS	SEAQ	ECRS	ECEM	ALGD	CSAL
sc	$55.85_{\pm 1.60}$	$57.12_{\pm 1.79}$	$58.98_{\pm 1.70}$	$53.02_{\pm 1.78}$	$62.95_{\pm 1.76}$	$\mathbf{67.17}_{\pm 1.69}$
poker	$58.73_{\pm 2.16}$	$59.75_{\pm 2.96}$	$60.97_{\pm 1.12}$	$56.52_{\pm 1.87}$	$65.04_{\pm 2.62}$	$\mathbf{68.21}_{\pm 1.13}$
covertype	$31.13_{\pm 1.19}$	$32.56_{\pm 2.9}$	$36.87_{\pm 1.28}$	$30.56_{\pm 2.94}$	$44.87_{\pm 1.23}$	$\mathbf{49.56}_{\pm 1.23}$
lr	$17.55_{\pm 1.48}$	$21.11_{\pm 1.88}$	$21.24_{\pm 1.68}$	$18.96_{\pm 2.34}$	$28.89_{\pm 1.96}$	$\mathbf{33.75}_{\pm 1.04}$
glass	$53.98_{\pm 1.12}$	$62.03_{\pm 1.93}$	$61.56_{\pm 1.14}$	$50.12_{\pm 1.78}$	$66.39_{\pm 1.43}$	$\mathbf{69.87}_{\pm 1.45}$
vowel	$68.54_{\pm 2.43}$	$70.21_{\pm 1.73}$	$72.76_{\pm 1.38}$	$67.64_{\pm 1.61}$	$\mathbf{77.09}_{\pm 1.56}$	$76.89_{\pm 2.05}$
MNIST	$47.23_{\pm 1.16}$	$51.90_{\pm 1.58}$	$50.5_{\pm 2.06}$	$48.6_{\pm 3.76}$	$54.6_{\pm 2.19}$	$\mathbf{56.8}_{\pm 1.82}$
fbis.wc	$54.35_{\pm 2.63}$	$58.76_{\pm 1.75}$	$57.68_{\pm 2.25}$	$51.07_{\pm 1.73}$	$63.58_{\pm 2.87}$	$\mathbf{75.45}_{\pm 1.25}$
ERA	$22.29_{\pm 3.08}$	$26.75_{\pm 2.12}$	$28.84_{\pm 2.87}$	$19.02_{\pm 1.25}$	$30.98_{\pm 1.25}$	$\mathbf{33.78}_{\pm 1.69}$
yeast	$33.78_{\pm 1.76}$	$35.85_{\pm 1.2}$	$36.67_{\pm 1.34}$	$31.56_{\pm 1.76}$	$42.21_{\pm 3.21}$	$\mathbf{44.23}_{\pm 2.35}$
Average	$44.34_{\pm 1.86}$	$47.6_{\pm 1.98}$	$48.6_{\pm 1.68}$	$42.7_{\pm 2.08}$	$53.7_{\pm 2.01}$	$\mathbf{57.67}_{\pm 1.57}$

the second tier. This observation asserts that considering data distribution and class label information simultaneously can boost model performance. Timely supervising the data distribution change helps explore new classes. Meanwhile, supervised class label information effectively detect the most uncertain data for the current model. In addtion, SEAQ performs always better than SERS, which again assert that the classification performance depends on a good query strategy which is capable to find both new class samples and most uncertain data at the same time. Another interesting observation is that ECRS is superior to SERS. This fact demonstrates that the use of clustering indeed improves classifier accuracy effectively. However,

although the mean performance of ECRS is a little better than SEAQ, we can not conclude ECRS is better. Either of them win in 5 data sets. It is fact that both consensus ensemble learning and adaptive query strategy have capability of boosting model performance, but which factor is more effective is unknown. As we expect, ECEM performs poorly on each data set. This is because when a data set with incomplete class information, its query strategy can not detect new class samples but the most uncertain instances.

Chapter 7

Conclusions

7.1 Summary of This Thesis

In many domains of interest, instances are connected in some ways. Instance correlation in a data set thus forms a network, in which neighboring instances frequently have close relationships (e.g. same class labels). For example, in document classification, documents that cite each other often have similar topics, and in social networks, people that are friends often have similar characteristics. Active learning focuses on exploiting such instance correlation achieves better predictive accuracy than AL treating them as independent samples. There has been numerous research integrating different kinds of instance correlation into AL scenario. Generally speaking, these works have explored instance correlation from three points of view: (1) Content based instance correlation uncovers the hidden topological structure in the feature spaces, explores relationship among the class labels, or discovers dependency between features and class labels. (2) Link based instance correlation explores the link information for networked data, and (3) Content and Link based Instance Correlation combines the link and the node-specific content information in a unified framework for classifying networked data.

In this thesis, we focus on exploring various content-based instance correlations in different active learning scenarios. We aim to explore these instance correlations from different views and utilize them to boost system performance in three different tasks, that are (1) reduce redundancy for optimal subset selection, (2) reduce labeling cost with a non-expert labeler and (3) discover class spaces for dynamic data sets.

Firstly, we incorporate instance correlation into labeling subset selection of AL, with a purpose of avoiding information redundancy in the subset. Unlike instance-based utility measures assess individual instances and label the ones with the maximum values, our method ALOSS simultaneously considers the following: 1) the importance of individual instances and 2) the disparity between instances, to build an instance-correlation matrix. We employ two types of distance measures, prediction distance and feature distance, for disparity assessment. The former intends to compare prediction dissimilarity of a same set of classifiers on two instances, and the latter intends to capture the disparity of a pair of instances in the feature space. Employment of instance-correlation matrix balances the requirements of data importance and diversity, effectively avoiding a redundancy issue in optimal subset selection. Experimental comparisons demonstrate that ALOSS outperforms the state-of-the-art active learners.

Secondly, a pairwise homogeneity query approach (PHAL) is introduced to further reduce labeling cost in active learning. Although active learning traditionally queries class labels of a subset, the labeling cost is still very expensive, even if the size of subset is not large. In this approach, a non-expert labeler is only asked "whether a pair of instances belongs to the same class ". This intuition is motivated by noise tolerate and label cost reduction. Even if the non-expert tells a wrong answer for a hard pair. 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.

Finally, an active class discovery method (CSAL) is proposed for dynamic data, where also a non-expert who can only answer pairwise label homogeneity questions is employed. With the help of pairwise label homogeneity information, this framework is capable to discover a number of candidate classes from an unlabeled data set and then further explore uncovered classes and most uncertain data by using a consensus ensemble learning model and an adaptive query strategy. Experiments and comparisons demonstrate that our new active learning framework achieves superior performance compared to various baseline methods on both static and dynamic data sets.

7.2 Future Work

In this thesis, we have studied active learning from the view of instance correlation. This studies are commonly applied to traditional small data sets. However, applying active learning algorithms to very large scale problems still poses challenges: (1) how to find an effective representation for a high-dimensional large-scale data, so as to fit in memory, (2) what kind of base learner is used for reducing the time in terms of model training and model testing, and (3) how to choose most uncertain data to update the model. To the best of our knowledge, there is no work combining large scale learning and active learning. In our future work, we focus on designing active learning algorithms and approaches that are faster, data efficient and less demanding in computational resources to achieve scalable algorithms for large scale problems.

Bibliography

[1] A. Acharya, E. R. Hruschka, J. Ghosh, and S. Acharyya. A framework for combining ensembles of classifier and clusters. In *Proceedings of MCS*, pages 269–278, 2011.

[2] A.Culotta and A. McCallum. Reducing labeling effort for stuctured prediction tasks. In *Proceeding of the 20th National Conference on Artificial Intelligenc(AAAI-2005)*, pages 746–751, 2005.

[3] M. Aminian. Active learning with scarcely labeled instances via bias variance reduction. In *Proceedings of International Conference on Artificial Intelligence and Machine Learning (ICAIML-2005)*, pages 41–45, 2005.

[4] D. Angluin. Queries and concept learning. *Machine Learning*, pages 319–342, 1988.

[5] M. F. Balcan, A. Beygelzimer, and J. Langford. Agnostic active learning. *Journal of Computer and System Sciences*, 75(1):78–89, 2009.

[6] S. Basu, A. Banerjee, and R.J. Mooney. Active semi-supervision for pairwise constrained clustering. In *Proceedings of the 20th International Conference on Machine Learning*, 2003.

[7] M. Becker, B. Hachey, B. Alex, and C. Grover. Optimising selective sampling for boostrapping named entity recognition. In *Proceedings of Workshop on Learning with Multiple View the 22nd International Conference on Machine Learning (ICML-2005)*, pages 5–11, Bonn , Germany, 2005.

[8] A. Beygelzimer, S. Dasgupa, and J. Langford. Important weighted active learning. In *Proceedings of the 26th International Conference on Machine Learning*, pages 49–56, 2009.

[9] A. Bifet, G. Holmes, B. Pfahringer, R. Kirkby, and R. Gavalda. New ensemble methods for evolving data streams. In *Proceedings of the 15th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (SIGKDD-2009)*, pages 139–148, Paris, France, 2009.

[10] M. Bilgic and L. Getoor. Active inference for collective classification. In *Proceedings of the 24th AAAI Conference on Artificial Intelligence (AAAI-2010)*, pages 1652–1655, Georgia, USA, 2010.

[11] M. Bilgic, L. Mihalkova, and L. Getoor. Active learning for networked data. In *Proceedings of ICML*, pages 79–86, 2010.

[12] Y. Bishan, J. Sun, T. Wang, and Z. Chen. Effective multi-label active learning for text classification. In *Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining (SIGKDD-2009)*, pages 917–925, Paris, France, 2009.

[13] A. Blum and S. Chawla. Learning form labled and unlabeled data using graph mincuts. In *Proceedings of ICML*, pages 19–26, 2001.

[14] A. Blum, J. Lafferty, M. Rwebangira, and R. Reddy. Seim-supervise learinng using randomized mincuts. In *Proceedings of ICML*, 2004.

[15] L. Bottou. *One approche theorique del apprentissage connexionniste: Applications. ala reconnaissance de la parole.* PhD thesis, Universite de Paris XI, 1991.

[16] K. Brinker. Incorporating diversity in active learning with support vector machines. In *Proceeding of International Conference on Machine Learning(ICML-2003)*, pages 59–66, 2003.

[17] MC. Burl and E. Wang. Active learning for directed exploration of complex systems. In *Proceedings of the 26th International Conference on Machine Learning (ICML 2009)*, pages 89–96, Montreal, Canada, 2009.

[18] C. Campbell, N. Cristianini, and A. Smola. Query learning with large margin classifiers. In *Proceedings of the 17th International Conference of Machine Learning (ICML 2000)*, pages 111–118, CA, USA, 2000.

[19] A. Cangelosi and T. Riga. Sdpa (semidefinite programming algorithm).

[20] A. Carlson, J. Berreridge, R. Wang, ER. Hruschka, and TM. Mitchell. Coupling semi-supervised learning of information extraction. In *Proceedings of the ACM International Conference on Web Search and Data Mining (ICWSDM-2010)*, pages 101–110, Washington DC, USA, 2010.

[21] N. Cesa-Bianchi, C. Gentile, F. Vitale, and G. Zappella. Active learning on trees and graphs. In *Proceedings of the 23rd International Conference on Learning Theory*, pages 320–332, Haifa, Israel, 2009.

[22] Y. Chan and H. Ng. Domain adaptation with active learning for word sense disambiguation. *Computational Linguistics*, 45:49–56, 2007.

[23] MW. Chang, L. Ratinov, N. Rizzolo, and D. Roth. Learning and inference with constraints. In *Proceedings of the 23rd national conference on Artificial intelligence*, pages 1513–1518, 2008.

[24] MW. Chang, L. Ratinov, and D. Roth. Guiding semi-supervision with constraint-driven learning. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics (ACL-2007)*, pages 280–287, Prague, Czech Republic, 2007.

[25] Y. Chen and M. Subramani. Study of active learning in the challenge. In *Proceedings of the International Joint Conference on Neural Network (IJCNN-2010)*, pages 1–7, 2010.

[26] H. Cheng, R. Zhang, Y. Peng, J. Mao, and P. Tan. Maximum margin active learning for sequence labeling with different length. In *Proceedings of the 8th Industrial Conference on Advances in Data Mining: Medical Applications E-Commerce Marketing and Theoretical Aspects (ICADM-2008)*, pages 345–359, Leipzig, Germany, 2008.

[27] Y. Cheng, K. Zhang, Y. Xie, A. Agrawal, W. Liao, and A. Choudhary. Learning to groupweb text incorporating prior information. In *Proceedings of ICDM Workshop*, 2011.

[28] W. Chu, M. Zinkevich, and L. Li. Unbiased online active learning in data streams. In *Proceedings of the 17th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (SIGKDD-2011)*, CA, USA, 2011.

[29] L. Copa, T. Devis, V. Michele, and K. Mikhail. Unbiased query-by-bagging active learning for vhr image classification. In *Proceedings of Conference on Image and Signal Processing for Remote Sensing XVI (ISPRS-2010)*, volume 7830, pages 78300K–78300K–8, Toulouse, France, 2010.

[30] TM. Cover and PE. Hart. Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, pages 21–27, 1967.

[31] A.T. Dasgupta, S.and Kalai and C. Monteleoni. Analysis of perceptron-based active learning. *J. Mach. Learn. Res.*, 10:281–299, June 2009.

[32] S. Dasgupta. Analysis of a greedy active learning strategy. In *Advances in Neural Information Processing Systems*, pages 337–344. MIT Press, 2004.

[33] S. Dasgupta, D. Hsu, and C. Monteleoni. A general agnostic active learning algorithm. Technical report, University of California, San Diego, 2007.

[34] C. Domeniconi and M. A. Razgan. Weighted cluster ensembles: Methods and analysis. *ACM Trans. Knowl. Discov. Data*, 2(4):1–40, 2009.

[35] P. Donmez and J. Carbonell. Optimizing estimated loss reduction for active sampling in rank learning. In *Proceeding of International Conference on Machine Learning(ICML-2008)*, 2008.

[36] P. Donmez and J. Carbonell. Paired sampling in density-sensitive active learning. In *Proceeding of International Symposium on Artificial Intelligence and Mathematics*, 2008.

[37] P. Donmez and J. Carbonell. Proactive learning: Cost-sensitive active learning with multiple imperfect oracles. In *Proceedings of the ACM Conference on Information and Knowledge Management(CIKM-2008)*, pages 619–628, 2008.

[38] J. Edmonds and R.M. Karp. Theoretical improvements in algorithmic efficiency for network flow problems. *Journal of the ACM*, pages 248–264, 1972.

[39] N. Escudeiro and A. Jorge. D-confidence: an active learning strategy which efficiently identifies small classes. In *Proceedings of workshop on Active Learning for Natural Language Processing (ALNLP-2010)*, pages 18–26, Los Angels, California, 2010.

[40] W. Fan, Y. Huang, H. Wang, and P. Yu. Active mining of data streams. In *Proceedings of SIAM international conference on data mining(SDM-2004)*, Florida, USA, 2004.

[41] Y. Fu, B. Li, X. Zhu, and C. Zhang. Do they belong to the same class: active learning by querying pairwise label homogeneity. In *Proceedings of CIKM*, pages 2161–2164, 2011.

[42] A. Fuji, T. Tokunaga, K. Inui, and H. Tanaka. Selective sampling for example based word sense disambiguation. *Computational Linguistics*, 24(4):573–597, 1998.

[43] J. Gao, W. Fan, Y. Sun, and J. Han. Heterogenous source consensus learning via decision propagation and negotiation. In *Proceedings of KDD*, 2009.

[44] R. Gilad-Bachrach and A. Navor. Kernel query by committee algorithm. *Technology Report no. 2003-88*, 2003.

[45] M. Godec, S. Sternig, P.M. Roth, and H. Bischof. Context-driven clustering by multi-class classification in an active learning framework. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, pages 19–24, 2010.

[46] M. Goemans and D. Williamson. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite prgramming. *Journal of ACM*, 42:1115–1145, 1995.

[47] A. Guillory and J. Bilmes. Labeled selection on graphs. In *Proceedings of NIPS*, pages 320–332, 2009.

[48] T. Haines and T. Xiang. Active learning using dirichlet processes for rare class discovery and classification. In *Proceedings of the British Machine Vision Conference*, 2011.

[49] S. Hanneke. A bound on the label complexity of agnostic active learning. In *Proceedings of the 24th international conference on Machine learning*, ICML '07, pages 353–360, New York, NY, USA, 2007. ACM.

[50] H. Hassanzadeh and M. Keyvanpour. A variance based active learning approach for named entity recognition. In Ran Chen, editor, *Intelligent Computing and Information Science*, volume 135 of *Communications in Computer and Information Science*, pages 347–352. Springer Berlin Heidelberg, 2011.

[51] J. He and J. Carbonell. Nearest-neighbor-based active learning for rare category detection. *Neural Information Processing system*, 2007.

[52] J. He and J. Carbonell. Rare class discovery based on active learning. In *Proceedings of ISAIM-08*, Florida, USA, 2008.

[53] SCH. Hoi, R. Jin, and M.R. Lyu. Large-scale text categorization by batch model active learning. In *Proceedings of the International Conference on the World Wide Web (WWW-2006)*, pages 633–642. ACM Press, 2006.

[54] S.C.H. Hoi, R. Jin, J. Zhu, and M.R. Lyu. Batch mode active learning and its application to medical image classification. In *Proceeding of International Conference on Machine Learning(ICML-2006)*, 2006.

[55] A. Holub and P. Perona. Entropy-based active learning for object recognition. In *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshop (CVPR-2008)*, pages 1–8, 2008.

[56] T.M. Hospedales, S. Gong, and T.Xiang. Find rare classes:active learning with generative and discriminative models. *IEEE Transactions on Knowledge and Data Mining*, 11, 2011.

[57] A. Huang, D. Milne, E. Frank, and I. Witten. Clustering documents with active learning using wikipedia. In *Proceedings of the 8th IEEE International Conference on Data Mining (ICDM-2008)*, pages 839–844, Pisa, 2008.

[58] T. Ishihara, K.I. Abe, and H. Takeda. Extensions of innovations dual control. *International Journal of Systems Sciences*, 19:653–667, 1988.

[59] D. Jensen, J. Neville, and B. Andgallagher. Why collective inference improves relational classification. In *Proceedings of ACM Special Interest Group on Knowledge Discovery and Data Mining (SIGKDD)*, pages 593–598, 2004.

[60] T. Joachims. Transductive learning via spectral graph partitioning. In *Proceedings of ICML*, pages 290–297, 2003.

[61] R. Jones, R. Ghani, T. Mitchell, and E. Rilo. Active learning for information extraction with multiple view feature sets. In *Proceedings of ECML Workshop on Adaptive Text Extraction and Mining (ATEM-2003)*, 2003.

[62] A. Jurek, Y. Bi, S. Wu, and C. Nugent. Classification by cluster analysis: A new meta-learning based approach. In *Proceedings of MCS*, pages 259–268, 2011.

[63] S. Kim, Y. Song, K. Kim, J. Cha, and G. Lee. Mmr-based active machine learning for bio named entity recognition. In *Proceedings of the Human Language Technology Conference of the NAACL, Companion Volume: Short Papers*, NAACL-Short '06, pages 69–72, Stroudsburg, PA, USA, 2006. Association for Computational Linguistics.

[64] J. Kunegis, A. Lommatzsch, and C. Bauckhage. Alternative similarity functions for graph kernels. In *Proceedings of International Conference on Pattern Recognition (ICPR-2008)*, pages 1–4, Florida, USA, 2003.

[65] J. Lafferty. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 282–289, 2001.

[66] D. Lewis and J. Catlett. Heterogeneous uncertainty sampling for supervised learning. In *Proceedings of the 11th International Conference on Machine Learning(ICML-1994)*, pages 148–156, 1994.

[67] D. Lewis and W. Gale. A sequential algorithm for training text classifiers. In *Proceedings of the ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR-1994)*, pages 3–12, Dublin, Ireland, 1994.

[68] B. Li, S. Yu, and Q. Lu. A sequential algorithm for training text classifiers. In *Proceedings of the ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR-1994)*, pages 3–12, Dublin, Ireland, 1994.

[69] D. Li, F. Qian, and P. Fu. Variance minimization approach for a class of dual control problems. In *Proceedings of the 2002 American Control Conference (ACC-2002)*, pages 3759–3764, Alaska, USA, 2002.

[70] M. Li and KS. Ishwar. Confidence-based active learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(8):1251–1261, 2006.

[71] C. X. Ling and J. Du. Active learning with direct query construction. In *Proceeding of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining(KDD-2008)*, pages 480–487, 2008.

[72] W. Liu and T. Wang. Online active multi-field learning for efficent email spam filtering. *Knowledge and Information Systems (KAIS)*, pages 1–20, 2011.

[73] Y. Liu, R. Jin, and A. K. Jain. Boostcluster: Boosting clustering by pairwise constraints. In *Proceedings of KDD*, 2007.

[74] B. Long, O. Chapelle, Y. Zhang, Y. Chang, Z. Zheng, and B. Tseng. Active learning for ranking through expected loss optimization. In *Proceedings of the 33rd international ACM SIGIR conference on Research and development in information retrieval (SIGIR-2010)*, pages 267–274, Geneva, Switzerland, 2010.

[75] 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*, pages 589–613, 2005.

[76] X. Ma, P. Luo, F. Zhang, Q. He, Z. Shi, and Z. Shen. Combining supervised and unsupervised models via unconstrained probablistic embedding. In *Proceedings of IJCAI*, 2011.

[77] S. A. Macskassy. Using graph-based metrics with empirical risk minimization to speed up active learning on networked data. In *Proceedings of the ACM Special Interest Group on Knowledge Discovery and Data Mining (SIGKDD)*, pages 597–606, 2009.

[78] G. Mann and A. McCallum. Efficent computation of entropy gradient for semi-supervised conditional random fields. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-2007)*, pages 109–112, PA, USA, 2007.

[79] A. McCallum and K. Nigam. Employing em in pool-based active learning for text classification. In *Proceeding of International Conference on Machine Learning(ICML-1998)*, pages 359–367, 1998.

[80] R. Milito, C. Padilla, R. Padilla, and D. Cadorin. An innovations approach to dual control. *IEEE Trans on Automatic Control*, 27(1):132–137, 1982.

[81] I. Muslea. *Active learning with multiple views*. PhD thesis, University of South California, 2002.

[82] D. Newman, S. Hettich, C. Blake, and C. Merz. Uci repository of machine learning. 1998.

[83] H. Nguyen and R. Caruana. Improving classification with pairwise constraints: A margin-based approach. In *Proceedings of EMCL KDD*, pages 847–856, 2008.

[84] H. Nguyen and B. Li. Cosine similarity metric learning for face verification. In *Proceedings of Asian Conference on Computer Vision (ACCV-2010)*, pages 709–720, QueensTown, New Zealand, 2010.

[85] H. Nguyen and A. Smeulders. Active learning using pre-clustering. In *Proceedings of the 21st International Conference on Machine Learning (ICML-2004)*, pages 1022–1027, Miami,Florida, 2004.

[86] U. Nodelman, C. Shelton, and D. Andkoller. Learning continuous time bayesian networks. In *Proceedings of the Annual Conference on Uncertainty in Artificial Intelligence (UAI-2003)*, pages 451–458, 2003.

[87] N. OTSU. A threshold selection method from gray-level histogram. *IEEE Transaction on Systems, Man, and Cybernetics*, 9(1), 1979.

[88] S. Pan, Y. Zhang, and X. Li. Dynamic classifier ensemble for positive unlabeled text stream classification. *Knowledge and Information Systems (KAIS)*, pages 1–21, 2011.

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

[90] D. Pelleg and A. Moore. Active learning for anomaly and rare category detection. *Neural Information Processing system*, 11, 2004.

[91] R. Polikar. Ensemble based systems in decision making. *IEEE Circuits and Systems Manazine*, 6(3):21–45, 2006.

[92] G. Qi, X. Hua, Y. Rui, J. Tang, and H. Zhang. Two-dimensional active learning for image classification. In *Proceedings of the 23rd IEEE Conference on Computer Vision and Pattern Recognition (CVPR-2008)*, pages 1–8, Alaska, USA, 2008.

[93] 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. In *Proc. of ICML*, 2009.

[94] N. Roy and A. McCallum. Toward optimal active learning through sampling estimation of error reduction. In *Proceeding of International Conference on Machine Learning(ICML-2001)*, pages 441–448, 2001.

[95] M. Saar-Tsechansky and F. Provost. Variance-based active learning. *The CeDER Working Paper, No.IS-00-05*, 2000.

[96] A. Saha, P. Rai, H. Daume, S. Venkatasubramanian, and S. DuVall. Active supervised domain adaptation. In *Proceedings of European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML/PKDD-2011)*, Athens, Greece, 2011.

[97] G. Seni and J. Elder. Ensemble methods in data mining: Improving accuracy through combining predictions. *Morgan and Claypool*, 2010.

[98] B. Settles. Active learning literature survey. *Technical Report 1648*, 2009.

[99] B. Settles and M. Craven. An analysis of active learning strategies for sequence labeling tasks. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP-2008)*, pages 1070–1079, 2008.

[100] B. Settles, M.Craven, and S. Ray. Multiple-instance active learning. *Advances in Neural Information Processing Systems*, 20:1289–1296, 2008.

[101] H.S. Seung, M. Opper, and H. Sompolinsky. Query by committee. In *Proceedings of the 5th Annual Workshop on Computational Learning Theory (COLT-1992)*, pages 287–294, Pittsburgh PA, USA, 1992.

[102] D. Shen, J. Zhang, J. Su, G. Zhou, and C. Tan. Multi-criteria-based active learning for named entity recognition. In *Proceedings of the 42nd annual Meeting of Association for Computational Linguistics (ACL-2004)*, pages 589–596, Barcelona, Spain, 2004.

[103] V.S. Sheng, F. Provost, and P. Ipeirotis. Get another label? improving data quality and data mining using multiple, noisy labelers. In *Proceedings of 16th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD-2008)*, pages 615–622, Washingto DC, USA, 2008.

[104] X. Shi, W. Fan, and J. Ren. Actively transfer domain knowledge. In *Proceedings of European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML/PKDD-2011)*, Antwerp, Belgium, 2008.

[105] S. Shum, N. Dehak, R. Dehak, and J. Glass. Unsupervised speaker adaptation based on the consine similarity for text-independent speaker verification. In *Proceedings of the IEEE Odyssey Workshop*, Brno, Czech Republic, 2010.

[106] S. Sohn, D. Comeau, W. Kim, and J. Wilbur. Term-centric active learning for naive bayes document classification. *The Open Information Systems Journal*, 3:54–67, 2009.

[107] S. Stolfo, W. Fan, W. Lee, and A. Prodromidis. Credit card fraud detection using meta-learning: Issues and initial results. In *Proceeding of AAAI Workshop on Fraud Detection and Risk Management*, pages 83–90, 1997.

[108] J. W. Strokes, J. C. Platt, J. Kravis, and M. Shilman. Aladin: Active learning of anomalies to detect intursions. *MSR, Tech.Rep.2008-24*, 2008.

[109] S. Sun. Active learning with extremely sparse labeled examples. In *Proceedings of the 10th Brazilian Symposium on Neural Networks*, pages 2980–2984, 2010.

[110] C. Tan, J. Tang, J. Sun, Q. Lin, and F. Andwang. Social action tracking via noise tolerant timevarying factor graphs. In *Proceedings of the ACM Special Interest Group on Knowledge Discovery and Data Mining (SIGKDD)*, pages 1049–1058, 2010.

[111] S. Tong and D. Koller. Support vector machine active learning with applications to text classification. In *Proceeding of International Conference on Machine Learning(ICML-2000)*, pages 999–1006, 2000.

[112] S. Tong and D. Koller. Support vector machine active learning with applications to text classification. *Journal of Machine Learning Research*, 2:999–1006, 2000.

[113] V.N. Vapnik and A. Chervonenkis. On the uniform convergence of relative frequencies of events to their probabilities. *Theory of Probability and Its Applications*, 16:264–280, 1971.

[114] P. Vatturi and W. K. Wong. Category detection using hierarchical mean shift. In *Proceedings of ACM SIGKDD*, pages 847–856, 2009.

[115] S. Vijayanarasimhan, P. Jain, and K. Grauman. Far-sighted active learning on a budget for image and video recognition. In *Proceedings of CVPR*, pages 3035–3042, 2010.

[116] X. Wang and I. Davidson. Active spectral clustering. In *Proceedings of ICDM 2010*, pages 561–568, 2010.

[117] JS. Weber and ME. Pollack. Entropy-driven online active learning for interactive calendar management. In *Proceedings of the 12th International Conference on Intelligent User Interfaces (ICIUI-2007)*, pages 141–150, Hawaii, USA, 2007.

[118] I. Witten and E. Frank. *Data mining: practical machine learning tools and techniques*. Morgan Kaufmann, 2005.

[119] B. Wittenmark. An active suboptimal dual controller for systems with stochastic parameters. *Automat Control Theory Application*, 3:13–19, 1975.

[120] C. Xiong, D. M. Johnson, and J. J. Corso. Sepctral active clustering via purification of the k-nearest neighbor graph. In *Proceedings of ECDM*, 2012.

[121] Q. Xu, M. Desjardins, and K. Wagstaff. Active constrained clustering by examining spectral eigenvectors. *Discovery Science*, pages 294–307, 2005.

[122] Z. Xu, R.Akella, and Y. Zhang. Incorporating diversity and density in active learning for relevance feedback. In *Proceeding of the annual European Conference on Information Retrieval*, pages 246–257, 2007.

[123] E. Shamir Y. Freund, H.S. Seung and N. Tishb. Selective samping using the query by committee algorithm. *Machine Learning*, 28:133–168, 1997.

[124] M. Yamashita, K. Fujisawa, and M. Kojima. Implementation and evaluation of sdpa 6.0. *Optim. Methods Softw.*, 18(4):491–505, 2003.

[125] R. Yan, J. Zhang, J. Yang, and AG. Hauptmann. A discriminative learning framework with pairwise constraints for video object classification. *IEEE Trans Pattern Anal Mach Intell*, pages 578–593, Apr 2006.

[126] S. Yan. Semi-automatic video semantic annotation based on active learning. *Visual Communications and Image Processing*, 5960:251–258, 2005.

[127] P. Zhang, X. Zhu, J. Tan, and L. Guo. Classifier and cluster ensembles formining concept drifting data streams. In *Proceedings of the 10th IEEE International Conference on Data Mining (ICDM-2010)*, pages 1175–1180, Sydney, Australia, 2010.

[128] Y. Zhang. Multi-task active learning with output constraints. In *Proceedings of the 24th AAAI Conference on Artificial Intelligence (AAAI-2010)*, pages 667–672, 2010.

[129] Y. Zhang, S. Burer, and W. Street. Ensemble pruning via semi-definite programming. *Journal of Machine Learning Research*, 57:1315–1338, 2006.

[130] L. Zhao, G. Sukthankar, and R. Sukthankar. Incremental relabeling for active learning with noisy crowdsourced annotations. In *Proceedings of Social Computing*, 2011.

[131] Y. Zhao, Y. Cao, and X. Pan. A telecom clients credit risk rating model based on active learning. In *Proceedings of IEEE International Conference on Automation and Logistics (ICAL-2008)*, pages 5–11, Qingdao, China, 2008.

[132] Y. Zhao, C. Xu, and Y. Cao. Research on query-by-committee method of active learning and application. *Lecture Notes on Artificial Intelligence (LNAI-2006)*, 4093:985–991, 2006.

[133] Z. Zhou, Y. Sun, and Y. Li. Multi-instance learning by treating instances as non-iid samples. In *Proceedings of the26th International Conference on Machine Learning (ICML-2009)*, pages 1249–1256, Montreal, Canada, 2009.

[134] J. Zhu, H. Wang, B. Tsou, and M. Ma. Active learning with sampling by uncertainty and density for instances annotations. *IEEE Transactions on Audio Speech and Language Processing*, 18(6):1323–1331, 2010.

[135] X. Zhu. *Semi-Supervised learning with graphs*. PhD thesis, Carnegie Mellon University, 2005.

[136] X. Zhu. Semi-supervised learning literature survey. *Computer Sciences TR 1530*, 2008.

[137] X. Zhu. Cross-domain semi-supervised learning using feature formulation. *IEEE Transactions on Systems, Man, and Cybernetics - Part B*, 41(6):1627–1638, 2011.

[138] X. Zhu, W. Ding, P. Yu, and C. Zhang. One-class learning and concept summarization for data streams. *Knowledge and Information Systems (KAIS)*, 28(3):523–553, 2011.

[139] X. Zhu, Z. Ghahramani, and L. John. Semi-supervised learning using gaussian fields and harmonic functions. In *Proceedings of the 20th International Conference on Machine Learning (ICML-2003)*, pages 912–919, Washington DC , USA, 2003.

[140] X. Zhu, J. Lafferty, and Z. Andghahramani. Combining active learning and semi-supervised learning using gaussian fields and harmonic functions. In *Proceedings of the International Conference on Machine Learning Workshop (ICML Workshop)*, pages 58–65, 2003.

[141] X. Zhu and X. Wu. Scalable representative instance selection and ranking. In *Proceedings of the 18th International Conference on Pattern Recognition (ICPR-2006)*, pages 352–355, 2006.

[142] X. Zhu, P. Zhang, X. Lin, and Y. Shi. Active learning from data streams. In *Proceedings of the 7th IEEE International Conference on Data Mining (ICDM-2007)*, pages 757–762, Nebraska, USA, 2007.

[143] X. Zhu, P. Zhang, Y. Shi, and X. Lin. Active learning from stream data using optimal weight classifier ensemble. *IEEE Trans. on Systems, Man, and Cybernetics, Part B*, pages 1607–1621, 2010.

[144] Z. Zhu, X. Zhu, Y. Ye, Y. Guo, and X. Xue. Transfer active learning. In *Proceedings of the 20th ACM International Conference on Information and Knowledge Management (CIKM-2011)*, Glasgow, UK, 2011.