Chapter 22

Active Learning: A Survey

Charu C. Aggarwal
IBM T. J. Watson Research Center
Yorktown Heights, NY
charu@us.ibm.com

Xiangnan Kong
University of Illinois at Chicago
Chicago, IL
xkong4@uic.edu

Quanquan Gu
University of Illinois at Urbana-Champaign
Urbana, IL
qgu3@illinois.edu

Jiawei Han
University of Illinois at Urbana-Champaign
Urbana, IL
hanj@illinois.edu

Philip S. Yu
University of Illinois at Chicago
Chicago, IL
psyu@uic.edu

22.1 Introduction 572
22.2 Motivation and Comparisons to Other Strategies 574
22.2.1 Comparison with Other Forms of Human Feedback 575
22.2.2 Comparisons with Semi-Supervised and Transfer Learning 576
22.3 Querying Strategies 576
22.3.1 Heterogeneity-Based Models 577
22.3.1.1 Uncertainty Sampling 577
22.3.1.2 Query-by-Committee 578
22.3.1.3 Expected Model Change 578
22.3.2 Performance-Based Models 579
22.3.2.1 Expected Error Reduction 579
22.3.2.2 Expected Variance Reduction 580
22.3.3 Representativeness-Based Models 580
22.3.4 Hybrid Models 580
22.4 Active Learning with Theoretical Guarantees 581
22.4.1 A Simple Example 581
22.4.2 Existing Works 582
22.4.3 Preliminaries 582

22.4.4 Importance Weighted Active Learning 582
22.4.4.1 Algorithm 583
22.4.4.2 Consistency 583
22.4.4.3 Label Complexity 584
22.5 Dependency-Oriented Data Types for Active Learning 585
22.5.1 Active Learning in Sequences 585
22.5.2 Active Learning in Graphs 585
22.5.2.1 Classification of Many Small Graphs 586
22.5.2.2 Node Classification in a Single Large Graph 587
22.6 Advanced Methods 589
22.6.1 Active Learning of Features 589
22.6.2 Active Learning of Kernels 590
22.6.3 Active Learning of Classes 591
22.6.4 Streaming Active Learning 591
22.6.5 Multi-Instance Active Learning 592
22.6.6 Multi-Label Active Learning 593
22.6.7 Multi-Task Active Learning 593
22.6.8 Multi-View Active Learning 594
22.6.9 Multi-Oracle Active Learning 594
22.6.10 Multi-Objective Active Learning 595
22.6.11 Variable Labeling Costs 596
22.6.12 Active Transfer Learning 596
22.6.13 Active Reinforcement Learning 597
22.7 Conclusions 597
Bibliography 597

22.1 Introduction

One of the great challenges in a wide variety of learning problems is the ability to obtain sufficient labeled data for modeling purposes. Labeled data is often expensive to obtain, and frequently requires laborious human effort. In many domains, unlabeled data is copious, though labels can be attached to such data at a specific cost in the labeling process. Some examples of such data are as follows:

- *Document Collections:* Large amounts of document data may be available on the Web, which are usually unlabeled. In such cases, it is desirable to attach labels to documents in order to create a learning model. A common approach is to manually label the documents in order to label the training data, a process that is slow, painstaking, and laborious.

- *Privacy-Constrained Data Sets:* In many scenarios, the labels on records may be sensitive information, which may be acquired at a significant query cost (e.g., obtaining permission from the relevant entity).

- *Social Networks:* In social networks, it may be desirable to identify nodes with specific properties. For example, an advertising company may desire to identify nodes in the social network that are interested in "cosmetics." However, it is rare that labeled nodes will be available in the network that have interests in a specific area. Identification of relevant nodes may only occur through either manual examination of social network posts, or through user surveys. Both processes are time-consuming and costly.

In all these cases, labels can be obtained, but only at a significant cost to the end user. An important observation is that all records are not equally important from the perspective of labeling. For example, some records may be noisy and contain no useful features that are relevant to classification. Similarly, records that cleanly belong to one class or another may be helpful, but less so than records that lie closer to the separation boundaries between the different classes.

An additional advantage of active learning methods is that they can often help in the removal of noisy instances from the data, which can be beneficial from an accuracy perspective. In fact, some studies [104] have shown that a carefully designed active learning method can sometimes provide better accuracy than is available from the base data.

Clearly, given the differential value of different records, an important question that arises in active learning is as follows:

How do we select instances from the underlying data to label, so as to achieve the most effective training for a given level of effort?

Different performance criteria may be used to quantify and fine-tune the tradeoffs between accuracy and cost, but the broader goal of all the criteria is to maximize the "bang for the buck" in spending the minimum effort in selecting examples, that maximize the accuracy as much as possible. An excellent survey on active learning may be found in [105].

Every active learning system has two primary components, one of which is already given:

- *Oracle:* This provides the responses to the underlying query. The oracle may be a human labeler, a cost driven data acquisition system, or any other methodology. It is important to note that the oracle algorithm is part of the input, though the user may play a role in its design. For example, in a multimedia application, the user may look at an image and provide a label, but this comes at an expense of human labor [115]. However, for most of the active learning algorithms, the oracle is really treated as a black box that is used directly.

- *Query System:* The job of the query system is to pose queries to the oracle for labels of specific records. It is here that most of the challenges of active learning systems are found.

Numerous strategies are possible for different active learning scenarios. At the highest level, this corresponds to the broader framework of *how* the queries are posed to the learner.

- *Membership Query Synthesis:* In this case, the learner actively synthesizes instances from the entire space, and does not necessarily sample from some underlying distribution [3]. The key here is that the learner many actually *construct* instances from the underlying space, which may not be a part of any actual pre-existing data. However, this may lead to challenges in the sense that the constructed examples may not be meaningful. For example, a synthesized image from a group of pixels will very rarely be meaningful. On the other hand, arbitrarily chosen spatial coordinates in a sea surface temperature prediction system will almost always be meaningful. Therefore, the usability of the approach clearly depends upon the underlying scenario.

- *Selective or Sequential Sampling:* In this case, the samples are drawn from the underlying data distribution, and the learner decides whether or not they should be labeled [21]. In this case, the query comes from an actual underlying data distribution, and is therefore guaranteed to make sense. In this case, the queries are sampled one by one, and a decision is made whether or not they should be queried. Such an approach is synonymous with the streaming scenario, since the decisions about querying an instance need to be made in real time in such cases. This terminology is however overloaded, since many works such as those in [104] use the term "selective sampling" to refer to another strategy described below.

- *Pool-based Sampling:* As indicated by its name, it suggests the availability of a base "pool" of instances from which to query the records of labels [74]. The task of the learner is to therefore determine instances from this pool (typically one by one), which are as informative as possible for the active learning process. This situation is encountered very commonly in practical scenarios, and also allows for relatively clean models for active learning.

The vast majority of the strategies in the literature use pool-based sampling, and in fact some works such as [104] refer to pool-based sampling as selective sampling. Therefore, this chapter will mostly focus on pool-based strategies, since these form the core of most active learning methods. Beyond these strategies, a number of other basic scenarios are possible. For example, in *batch learning*, an entire set of examples need to be labeled at a given time for the active learning process. An example of this is the *Amazon Mechanical Turk*, in which an entire set of examples is made available for labeling at a given time. This is different from the methods common in the literature, in which samples are labeled one by one, so that the learner has a chance to adjust the model, before selecting the next example. In such cases, it is usually desirable to incorporate *diversity* in the batch of labeled instances, in order to ensure that there is not too much redundancy within a particular batch of labeled instances [16, 55, 57, 119].

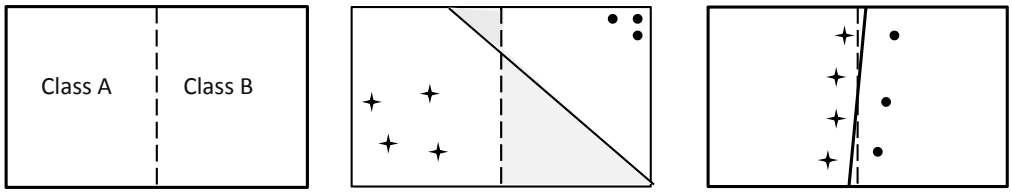
Active learning has numerous challenges, in that it does not always improve the accuracy of classification. While some of these issues may be related to algorithmic aspects such as sample selection bias [12], other cases are inherent to the nature of the underlying data. However, in many special cases, it has been shown [30] the number of labels needed to learn actively can be logarithmic in the usual sample complexity of passive learning.

This chapter will discuss the different methods that are used for active learning. First, we will provide a motivational example of how the selective sampling approach can describe the contours of the different class boundaries with far fewer examples. Then, we will provide a discussion of the different strategies that are commonly used for active learning. We will see that number of different scenarios are possible for the active learning process in terms of how the samples are selected.

This chapter is organized as follows. Section 22.2 provides an example of how active learning provides advantages for the learning process. We will also discuss its relationship to other methods in the literature such as semi-supervised learning. Section 22.3 discusses query strategy frameworks for active learning. Section 22.4 studies models for theoretical active learning. Section 22.5 discusses the methodologies for handling complex data types such as sequences and graphs. Section 22.6 discusses advanced topics for active learning, such as streaming data, feature learning, and class-based querying. Section 22.7 discusses the conclusions and summary.

22.2 Motivation and Comparisons to Other Strategies

The primary motivation of active learning is the paucity of training data available for learning algorithms. While it is possible to query the data randomly for labels, such an approach may not result in the best model, when each query is costly, and therefore, few labels will eventually become available. For example, consider the two class example of Figure 22.1. Here, we have a very simple division of the data into two classes, which is shown by a vertical dotted line, as illustrated in Figure 22.1(a). The two classes here are labeled A and B. Consider the case where it is possible to query only 7 examples for the two different classes. In this case, it is quite possible that the small number of allowed samples may result in a training data that is unrepresentative of the true separation between the two classes. Consider the case when an SVM classifier is used in order to construct a model. In Figure 22.1(b), we have shown a total of 7 samples randomly chosen from the underlying data. Because of the inherent noisiness in the process of picking a small number



(a) Class Separation

(b) Random Sample with SVM Classifier

(c) Active Sample with SVM Classifier

FIGURE 22.1: Motivation of active learning.

of samples, an SVM classifier will be unable to accurately divide the data space. This is shown in Figure 22.1(b), where a portion of the data space is incorrectly classified, because of the error of modeling the SVM classifier. In Figure 22.1(c), we have shown an example of a well chosen set of seven instances along the decision boundary of the two classes. In this case, the SVM classifier is able to accurately model the decision regions between the two classes. This is because of the careful choice of the instances chosen by the active learning process. An important point to note is that it is particularly useful to sample instances that provide a distinct view of how the different classes are separated in the data. As will be discussed later, this general principle is used quite frequently in a variety of active learning methods, where regions of greater uncertainty are often sampled in order to obtain the relevant decision boundaries [75].

Since active learning is an approach that often uses human feedback in order to account for the lack of obtaining training examples, it is related to a number of other methods that either use human feedback or augment it with other kinds of training information. These two classes of methods are as follows:

- *Human Feedback:* Human feedback is often used to improve the accuracy of the learning process. For example, decision trees and other classifiers may be built with the active intervention of the user [4, 113]. In this case, the model itself is constructed with user-intervention rather than the choice of data examples.
- *Semi-supervised and Transfer Learning:* In this case, other *kinds* of data (e.g., unlabeled data or labeled data from other domains) are used in order to overcome the lack of training examples [4, 9, 29, 94, 97, 134].

Since both of these forms of supervision share some common principles with active learning, they will be discussed in some detail below.

22.2.1 Comparison with Other Forms of Human Feedback

Human feedback is a common approach to improving the effectiveness of classification problems. Active learning is one approach for improving the effectiveness of classification models, which is highly label-centered, and focusses *only on acquisition of labels*. Other human-centered methods used human feedback directly in order to improve the model for a given set of labels. For example, a splitting choice in a decision tree can be greatly involved with the use of human intervention. As in the case of active learning, the use of human feedback is greatly helpful in improving the quality of the model constructed with a fewer number of examples. However, the focus of the approach is completely different in terms of using human intervention at the model level rather than

at the level of labels [4, 113]. A detailed discussion on the use of human feedback is provided in the chapter on visual classification in this book.

22.2.2 Comparisons with Semi-Supervised and Transfer Learning

These forms of learning are also related to the difficulty in acquisition of labels, though the approach used to improve the learning process is very different. In this case, different kinds of data are used to improve the learning process and *augment* the sparse labeled data that is already available. However, the instances that are used to augment the sparse labeled data are either unlabeled, or they are drawn from another domain. The motivations of using such methods, which are quite distinct from transfer learning, are as follows:

- *Semi-supervised Learning:* In this case, unlabeled data is used in order to learn the base distribution of the data in the underlying space [9, 94]. Once the base distribution of the data has been learned, it is combined with the labeled data in order to learn the class contours more effectively. The idea here is that the data is often either aligned along low dimensional manifolds [9], or is clustered in specific regions [94], and this can be learned effectively from the training data. This additional information helps in learning, even when the amount of labeled data available is small. For example, when the data is clustered [94], each of the dense regions typically belongs to a particular class, and a very small number of labels can map the different dense regions to the different classes.

- *Transfer Learning:* Transfer learning also uses additional labeled data from a different source, and may sometimes even be drawn from a different domain. For example, consider the case where it is desirable to classify Chinese documents with a small training collection. While it may be harder to obtain labeled Chinese documents, it is much easier to obtain labeled English documents. At the same time, data that provides correspondence between Chinese and English documents may be available. These different kinds of data may be combined together in order to provide a more effective training model for Chinese documents. Thus, the core idea here is to "transfer" knowledge from one domain to the other. However, transfer learning does not actively acquire labels in order to augment the sparse training data.

Thus, human feedback methods, transfer learning methods, and semi-supervised learning methods are all designed to handle the problem of paucity of training data. This is their shared characteristic with active learning methods. However, at the detailed level, the strategies are quite different. Both semi-supervised learning and transfer learning have been discussed in detail in different chapters of this book.

22.3 Querying Strategies

The key question in active learning algorithms is to design the precise strategies that are used for querying. At any given point, which sample should be selected so as to maximize the accuracy of the classification process? As is evident from the discussion in the previous section, it is advantageous to use strategies, so that the contours of separation between the different classes are mapped out with the use of a small number of examples. Since the boundary regions are often those in which instances of multiple classes are present, they can be characterized by class label uncertainty or disagreements between different learners. However, this may not always be the case, because instances with greater uncertainty are not representative of the data, and may sometimes lead to the selection

of unrepresentative outliers. This situation is especially likely to occur in data sets that are very noisy. In order to address such issues, some models focus directly on the error itself, or try to find samples that are representative of the underlying data. Therefore, we broadly classify the querying strategies into one of three categories:

- *Heterogeneity-based models:* These models attempt to sample from regions of the space that are either more heterogeneous, or dissimilar to what has already been seen so far. Examples of such models include *uncertainty sampling*, *query-by-committee*, and *expected model change*. All these methods are based on sampling either uncertain regions of the data, or those that are dissimilar to what has been queried so far. These models only look at the heterogeneity behavior of the queried instance, rather than the effect of its addition on the performance of a classifier on the remaining unlabeled instances.
- *Performance-based models:* These models attempt to directly optimize the performance of the classifier in terms of measures such as error or variance reduction. One characteristic of these methods is that they look at the effect of adding the queried instance on the performance of the classifier on the remaining unlabeled instances.
- *Representativeness-based models:* These models attempt to create data that is as representative as possible of the underlying population of training instances. For example, density-based models are an example of such scenarios. In these cases, a product of a heterogeneity criterion and a representativeness criterion is used in order to model the desirability of querying a particular instance. Thus, these methods try to balance the representativeness criteria with the uncertainty properties of the queried instance.

Clearly, there is significant diversity in the strategies that one may use in the active learning process. These different strategies have different tradeoffs and work differently, depending upon the underlying application, analyst goal, and data distribution. This section will provide an overview of these different strategies for active learning. Throughout the following discussion, it will be assumed that there are a total of k classes, though some cases will also be analyzed in the binary scenario when k is set to 2.

22.3.1 Heterogeneity-Based Models

In these models, the idea is to learn the regions that show the greatest heterogeneity, either in terms of uncertainty of classification, dissimilarity with the current model, or disagreement between a committee of classifiers. These different techniques will be studied in this section.

22.3.1.1 Uncertainty Sampling

In uncertainty sampling, the learner attempts to label those instances for which it is least certain how to label. In a binary classification problem, the simplest possible uncertainty labeling scheme would be to use a Bayes classifier on an instance, and query for its label, if the predicted probability of the most probable label is as close to 0.5 as possible [74, 75]. The probabilities predicted by the classifier should be normalized so that they sum to 1. This is important, since many classifiers such as the unnormalized naive Bayes classifier often predict probabilities that do not sum to 1. Therefore, the following entropy-centered objective function $En(\overline{X})$ needs to be minimized for the binary class problem:

$$En(\overline{X}) = \sum_{i=1}^{k} ||p_i - 0.5||.$$

A second criterion is the difference in the predicted probabilities between the two classes. This is, however, equivalent to the first criterion, if the two probabilities have been normalized. A second

pair of criteria that are especially relevant for k-ary classification is the entropy measure or the gini-index. If the predicted probabilities of the k classes are $p_1 \ldots p_k$, respectively, based on the current set of labeled instances, then the entropy measure $En(\overline{X})$ is defined as follows:

$$En(\overline{X}) = -\sum_{i=1}^{k} p_i \cdot \log(p_i).$$

Larger values of the entropy indicate greater uncertainty. Therefore, this objective function needs to be maximized. Note that an equal proportion of labels across the different classes results in the highest possible entropy. A second measure is the gini-index G.

$$G(\overline{X}) = 1 - \sum_{i=1}^{k} p_i^2.$$

As in the case of entropy, higher values of the gini-index indicate greater uncertainty. It should be pointed out that some of these measures may not work in the case of imbalanced data, where the classes are not evenly distributed. In such cases, the classes may often be associated with costs, where the cost of misclassification of i is denoted by w_i. Each probability p_i is replaced by a value proportional to $p_i \cdot w_i$, with the constant of the proportionality being determined by the probability values summing to 1.

Numerous active sampling techniques have been developed in the literature on the basis of these principles, and extensive comparisons have also been performed between these different techniques. The interested reader is referred to [27, 60, 74, 75, 102, 106] for the different techniques, and to [68, 103, 106] for the comparison of these measures. It should also be pointed out that it is not necessary to use a Bayes model that explicitly predicts probabilities. In practice, it is sufficient to use any model that provides a prediction confidence for each class label. This can be converted into a pseudo-probability for each class, and used heuristically for the instance-selection process.

22.3.1.2 Query-by-Committee

This approach [109] uses a committee of different classifiers, which are trained on the current set of labeled instances. These classifiers are then used to predict the class label of each unlabeled instance. The instance for which the classifiers disagree the most is selected as the relevant one in this scenario. At an intuitive level, the query-by-committee method achieves similar heterogeneity goals as the uncertainty sampling method, except that it does so by measuring the differences in the predictions of different classifiers, rather than the uncertainty of labeling a particular instance. Note that an instance that is classified to different classes with almost equal probability (as in uncertainty sampling) is more likely to be predicted in different classes by different classifiers. Thus, there is significant similarity between these methods at an intuitive level, though they are generally treated as very different methods in the literature. Interestingly, the method for measuring the disagreement is also quite similar between the two classes of methods. For example, by replacing the prediction probability p_i of each class i with the fraction of votes received for each class i, it is possible to obtain similar measures for the entropy and the gini-index. In addition, other probabilistic measures such as the KL-divergence have been proposed in [84] for this purpose.

The construction of the committee can be achieved by either varying the model parameters of a particular classifier (through sampling) [28, 84], or by using a bag of different classifiers [1]. It has been shown that the use of a small number of classifiers is generally sufficient [109], and the use of diverse classifiers in the committee is generally beneficial [89].

22.3.1.3 Expected Model Change

A decision theoretic-approach is to select the instance that results in the greatest change from the current model. Specifically, the instance that results in the greatest change in gradient of the

objective function with respect to the model parameters is used. The intuition of such an approach is to use an instance that is most different from the current model that is already known. Thus, this is also a heterogeneity-based approach, as is the case with uncertainty sampling, and query-by-committee. Such an approach is only applicable to models where gradient-based training is used, such as discriminative probabilistic models. Let $\delta g_i(\overline{X})$ be the change in the gradient with respect to the model parameters, if the training label of the candidate instance $\overline{X}$ (with unknown label) is i. Let p_i be the posterior probability of the instance i with respect to the current label set in the training data. Then, the expected model change $C(\overline{X})$ with respect to the instance $\overline{X}$ is defined as follows:

$$C(\overline{X}) = \sum_{i=1}^{k} p_i \cdot \delta g_i(\overline{X}).$$

The instance $\overline{X}$ with the largest value of $C(\overline{X})$ is queried for the label. Numerous techniques for querying, that have been proposed using this approach may be found in [25, 47, 106, 107].

22.3.2 Performance-Based Models

The primary criticism of heterogeneity-based models is that the goal of trying to identify the most unknown regions of the space (based on the current labeling), may sometimes lead to the identification of noisy and unrepresentative regions of the data. The precise impact of using such an approach is of course highly data-dependent. There are two classes of techniques that are based on the performance of a classifier on the *remaining unlabeled instances*.

22.3.2.1 Expected Error Reduction

For the purpose of discussion, the remaining set of instances that have not yet been labeled are denoted by V. This set is used as the validation set on which the expected error reduction is computed. This approach is related to uncertainty sampling in a complementary way. Whereas uncertainty sampling *maximizes* the label uncertainty of the *queried* instance, the expected error reduction *minimizes* the expected label uncertainty of the *remaining* instances V, when the queried instance is added to the data. Thus, in the case of a binary-classification problem, we would like the labels of the instances in V to be as far away from 0.5 as possible. The idea here is that it is good to query for instances, which results in greater certainty of class label for the remaining error rate. Thus, error reduction models can also be considered as *greatest certainty* models, except that the certainty criterion is applied to the instances in V (rather than the query instance itself) after addition of the queried instance to the model. The assumption is that greater certainty of class labels *of the remaining unlabeled instances* corresponds to a lower error rate. Let $p_i(\overline{X})$ denote the posterior probability of the label i for the instance $\overline{X}$, before the queried instance is added. Let $P_j^{(\overline{X},i)}(\overline{Z})$ be the posterior probability of class label j, when the instance-label combination $(\overline{X}, i)$ is added to the model. Then, the error objective function $E(\overline{X}, V)$ for the binary class problem (i.e., $k = 2$) is defined as follows:

$$E(\overline{X}, V) = \sum_{i=1}^{k} p_i(\overline{X}) \cdot \left(\sum_{j=1}^{k} \sum_{\overline{Z} \in V} ||P_j^{(\overline{X},i)}(\overline{Z}) - 0.5|| \right). \tag{22.1}$$

The value of $E(\overline{X}, V)$ is maximized rather than minimized (as in the case of uncertainty-based models). Furthermore, the error objective is a function of both the queried instance and the set of unlabeled instances V. This result can easily be extended to the case of k-way models by using the entropy criterion, as was discussed in the case of uncertainty-based models. In that case, the expression above is modified to replace $||P_j^{(\overline{X},i)}(\overline{Z}) - 0.5||$ with the class-specific entropy term $-P_j^{(\overline{X},i)}(\overline{Z}) \cdot \log(P_j^{(\overline{X},i)}(\overline{Z}))$. Furthermore, this criterion needs to be minimized. In this context, the

minimization of the expression can be viewed as the minimization of the expected loss function. This general framework has been used in a variety of different contexts in [53, 93, 100, 135].

22.3.2.2 Expected Variance Reduction

One observation about the afore-mentioned error reduction method of Equation 22.1 is that it needs to be computed in terms of the entire set of unlabeled instances in V, and a new model needs to be trained incrementally, in order to test the effect of adding a new instance. The model is therefore expensive to compute. It should be pointed out that when the error of an instance set reduces, the corresponding variance also typically reduces. The overall generalization error can be expressed as a sum of the true label noise, model bias, and variance [45]. Of these, only the last term is highly dependent on the choice of instances selected. Therefore, it is possible to reduce the variance instead of the error, and the main advantage of doing so is the reduction in computational requirements.

The main advantage of these techniques is the ability to express the variance in *closed form*, and therefore achieve greater computational efficiency. It has been shown [22, 23, 85] that the variance can be expressed in closed form for a wide variety of models such as neural networks, mixture models, or linear regression. In particular, it has been shown [85], that the output variance can be expressed in terms of the gradient with respect to the model parameters, and the Fisher Information Matrix. Interested readers are referred to [22, 23, 85, 103, 129] for details of the different variance-based methods.

22.3.3 Representativeness-Based Models

The main advantage of error-based models over uncertainty-based models is that they intend to improve the error behavior on the *aggregate*, rather than looking at the uncertainty behavior of the *queried* instance, as in heterogeneity-based models. Therefore, unrepresentative or outlier-like queries are avoided. However, representativeness can also be achieved by querying the data in such a way that the acquired instances tend to resemble the overall distribution better. This is achieved by weighting dense regions of the input space to a higher degree during the querying process. Examples of such methods include density-based models [106]. Therefore, these methods *combine* the heterogeneity behavior of the queried instance with a representativeness function from the unlabeled set V in order to decide on the queried instance. Therefore, in general, the objective function $O(\overline{X}, V)$ of such a model is expressed as the product of a heterogeneity component $H(\overline{X})$ and a representativeness component $R(\overline{X}, V)$:

$$O(\overline{X}, V) = H(\overline{X}) \cdot R(\overline{X}, V).$$

The value of $H(\overline{X})$ (assumed to be a maximization function) can be any of the heterogeneity criteria (transformed appropriately for maximization) such as the entropy criterion $En(\overline{X})$ from uncertainty sampling, or the expected model change criterion $C(\overline{X})$. The representativeness criterion $R(\overline{X}, V)$ is simply a measure of the density of $\overline{X}$ with respect to the instances in V. A simple version of this density is the average similarity of $\overline{X}$ to the instances in V [106], though it is possible to use more sophisticated methods such as kernel-density estimation to achieve the same goal. Note that such an approach is likely to ensure that the instance $\overline{X}$ is in a dense region, and is therefore not likely to be an outlier. Numerous variations of this approach have been proposed, such as those in [42, 84, 96, 106].

22.3.4 Hybrid Models

Conventional approaches for active learning usually select either informative or representative unlabeled instances. There are also some works combining multiple criteria for query selection in active learning [31, 58, 59, 120], such that the queried instances will have the following properties:

1) Informative, the queried instance will be close to the decision boundary of the learning model in terms of criteria like uncertainty, or the queried instance should be far away from existing labeled instances in order to bring new knowledge about the feature space. 2) Representative, the queried instance should be less likely to be outlier data and should be representative to a group of other unlabeled data. For example, in the work [59], the query selection is based upon both informativeness and representativeness of the unlabeled instances. A min-max framework of active learning is used to measure scores for both criteria.

22.4 Active Learning with Theoretical Guarantees

Let us recall the typical setting of active learning as follows: given a pool of unlabeled examples, an active learner is allowed to interactively query the label of any particular examples from the pool. The goal of active learning is to learn a classifier that accurately predicts the label of new examples, while requesting as few labels as possible.

It is important to contrast active learning to traditional passive learning, where labeled examples are chosen randomly. In the passive learning literature, there are well-known bounds on the number of training examples that is necessary and sufficient to learn a near-optimal classifier with a high probability. This quantity is called *sample complexity*, which depends largely on the VC dimension [14] of the hypothesis space being learned.

A natural idea is to define a similar quantity for active learning. It is called *label complexity*, i.e., the number of labeling requests that is necessary and sufficient to learn a near-optimal model with a high probability. However, not every active learning algorithm can be analyzed in terms of the label complexity. In previous sections, we introduce many active learning algorithms that are very effective empirically. However, most of these algorithms do not have any theoretical guarantee on the consistency or the label complexity [54]. In this section, we are particularly interested in active learning algorithms whose behaviors can be rigorously analyzed, i.e., that converge to an optimal hypothesis in a given hypothesis class with substantially lower label complexity than passive learning.

22.4.1 A Simple Example

Before going into the details of active learning with provable guarantee, we first present a simple example [6], which demonstrates the potential of active learning in the noise-free case when there is a perfect hypothesis with zero error rate. This is probably the simplest active learning algorithm with provable label complexity.

Consider the active learning algorithm that searches for the optimal threshold on an interval using binary search. We assume that there is a perfect threshold separating the classes, i.e., the realizable case.[1] Binary search needs $O(\log(\frac{1}{\varepsilon}))$ labeled examples to learn a threshold with error less than ε, while passive learning requires $O(\frac{1}{\varepsilon})$ labels. A fundamental drawback of this algorithm is that a small amount of adversarial noise can force the algorithm to behave badly. Thus, it is crucial to develop active learning algorithms that can work in the non-realizable case.[2]

[1] The target concept function is in the hypothesis class we considered.

[2] The target concept function is not in the hypothesis class we considered.

22.4.2 Existing Works

An early landmark result is the selective sampling scheme proposed in [22]. This simple active learning algorithm, designed for the realizable case, has triggered a lot of subsequent works. The seminal work of [44] analyzed an algorithm called query-by-committee, which uses an elegant sampling strategy to decide when to query examples. The core primitive required by the algorithm is the ability to sample randomly from the posterior over the hypothesis space. In some cases, such as the hypothesis class of linear separators in $\mathbb{R}^d$ and where the data is distributed uniformly over the surface of the unit sphere in $\mathbb{R}^d$, this can be achieved efficiently [46]. In particular, the authors showed that the number of labels required to achieve a generalization error ε is just $O(d\log(\frac{1}{\varepsilon}))$, which is exponentially lower than the sample complexity of a typical supervised learning algorithm $O(\frac{d}{\varepsilon})$. Later, [30] showed that a simple variant of Perceptron algorithm also achieves $O(d\log(\frac{1}{\varepsilon}))$ label complexity provided by a spherically uniform unlabeled data distribution. All the works mentioned above address active learning in the realizable case. A natural way to extend active learning to the non-realizable case (i.e., the agnostic case) is to ask the learner to return a hypothesis with error at most $L^* + \varepsilon$, where L^* is the error of the best hypothesis in the specified hypothesis class. The first rigorous analysis of agnostic active learning algorithm was developed by [6], namely A^2. later, [54] derived the label complexity of A^2 algorithm in terms of a parameter, called disagreement coefficient.

Another line of research uses importance weights to avoid the bias due to the active learning. For example, [5] considers linear representations and places some assumptions on the data generation process. However, the analysis is asymptotic rather than being a finite label complexity. To overcome this drawback, [12] proposed a new algorithm, called IWAL, satisfying PAC-style label complexity guarantees.

Due to the space limit, we do not discuss all the above methods in detail. In the remaining part of this section, we choose to introduce the IWAL algorithm [12] in detail.

22.4.3 Preliminaries

Let X be the instance space and $Y = \{\pm 1\}$ be the set of possible labels. Let H be the hypothesis class, i.e., a set of mapping functions from X to Y. We assume that there is a distribution D over all instances in X. For simplicity, we assume that H is finite ($|H| < \infty$), but does not completely agree on any single $x \in X$, i.e., $\forall x \in X, \exists h_1, h_2 \in H, h_1(x) \neq h_2(x)$. Note that $|H|$ can be replaced by VC dimension for an infinite hypothesis space. The algorithm is evaluated with respect to a given loss function $\ell : Y \times Y \to [0, \infty)$. The most common loss is $0-1$ loss, i.e., $\ell(z,y) = 1(y \neq z)$. The other losses include squared loss $\ell(z,y) = (y-z)^2$, hinge loss $\ell(z,y) = (1-yz)_+$ and logistic loss $\ell(z,y) = \log(1+\exp(-yz))$. The loss of a hypothesis $h \in H$ with respect to a distribution P over $X \times Y$ is defined as

$$L(h) = \mathbb{E}_{(x,y)\sim P}\ell(h(x), y). \tag{22.2}$$

Let $h^* = \arg\min_{h\in H} L(h)$ be a hypothesis of the minimum error in H and $L^* = L(h^*)$. We have $L^* = 0$ in the realizable case, and $L^* > 0$ in the non-realizable case. The goal of active learning is to return a hypothesis $h \in H$ with an error $L(h)$ that is not much more than $L(h^*)$, using as few label queries as possible.

22.4.4 Importance Weighted Active Learning

In this section, we will discuss the problem of importance-weighted active learning.

22.4.4.1 Algorithm

In the importance weighted active learning (IWAL) framework [12], an active learner looks at the unlabeled data $x_1, x_2, \ldots, x_t$ one by one. After each new point x_t, the learner determines a probability $p_t \in [0,1]$. Then a coin with the bias p_t is tossed, and the label y_t is queried if and only if the coin comes up heads. The query probability p_t depends on all previous unlabeled examples $x_{1:t-1}$, any previously queries labels, and the current unlabeled example x_t.

The algorithm maintains a set of labeled examples seen so far, where each example is assigned with an importance value p_t. The key of IWAL is a subroutine, which returns the probability p_t of requesting y_t, given x_t and the previous history $x_i, y_i : 1 \leq i \leq t-1$. Specifically, let $Q_t \in \{0,1\}$ be a random variable that determines whether to query y_t or not. That is, $Q_t = 1$ indicates that the label y_t is queried, and otherwise $Q_t = 0$. Then $Q_t \in \{0,1\}$ is conditionally independent of the current label y_t, i.e.,

$$Q_t \perp Y_t | X_{1:t}, Y_{1:t-1}, Q_{1:t-1} \tag{22.3}$$

and with conditional expectation

$$\mathbb{E}[Q_t | X_{1:t}, Y_{1:t-1}] = p_t. \tag{22.4}$$

If y_t is queried, IWAL adds $(x_t, y_t, 1/p_t)$ to the set, where $1/p_t$ is the importance of predicting y_t on x_t. The key of IWAL is how to specify a rejection threshold function to determine the query probability p_t. [12] and [13] discussed different rejection threshold functions.

The importance weighted empirical loss of a hypothesis h is defined as

$$L_T(h) = \frac{1}{T} \sum_{t=1}^{T} \frac{Q_t}{p_t} \ell(h(x_t), y_t) \tag{22.5}$$

A basic property of the above estimator is unbiasedness. It is easy to verify that $\mathbb{E}[L_T(h)] = L(h)$, where the expectation is taken over all the random variables involved.

To summarize, we show the IWAL algorithm in Algorithm 22.1.

Algorithm 22.1 Importance Weighted Active Learning (IWAL)

Input: $S_0 = \emptyset$,
for $t = 1$ to T **do**
  Receive x_t
  Set $p_t = rejection - threshold(x_t, \{x_i, y_i : 1 \leq i \leq t-1\})$
  Toss a coin $Q_t \in 0,1$ with $\mathbb{E}[Q_t] = p_t$
  if $Q_t = 1$ **then**
    request y_t and set $S_t = S_{t-1} \cup \{(x_t, y_t, p_{min}/p_t)\}$
  else
    $S_t = S_{t-1}$
  end if
  Let $h_t = \arg\min_{h \in H} \sum_t \frac{1}{p_t} \ell(h(x_t), y_t)$
end for

22.4.4.2 Consistency

A desirable property of any learning algorithm is the consistency. The following theorem shows that IWAL is consistent, as long as p_t is bounded away from 0. Furthermore, its sample complexity is within a constant factor of supervised learning in the worst case.

Theorem 3 *[12] For all distributions D, for all finite hypothesis classes H, if there is a constant*

$p_{min} > 0$ such that $p_t > p_{min}$ for all $1 \leq t \leq T$, then with a probability that is at least $1-\delta$, we have

$$L_T(h) \leq L(h) + \frac{\sqrt{2}}{p_{min}} \sqrt{\frac{\log|H| + log(\frac{1}{\delta})}{T}}. \tag{22.6}$$

Recall that a typical supervised learning algorithm has the following bound for sample complexity

Theorem 4 *For all distributions D, for all finite hypothesis classes H, with a probability that is at least $1-\delta$, we have*

$$L_T(h) \leq L(h) + \sqrt{\frac{log|H| + \log(\frac{1}{\delta})}{T}}. \tag{22.7}$$

By comparing with the above results, we can see that the sample complexity of IWAL is at most $\frac{2}{p_{min}^2}$ times the sample complexity of a typical supervised learning algorithm.

In fact, a fairly strong and large deviation bound can be given for each h_t output by IWAL as follows:

Theorem 5 *[12] For all distributions D, for all finite hypothesis classes H, with a probability that is at least $1-\delta$, the hypothesis output by IWAL satisfies*

$$L(h_T) \leq L(h) + 2\sqrt{\frac{8}{t} \log \frac{t(t+1)|H_t|^2}{\delta}}. \tag{22.8}$$

22.4.4.3 Label Complexity

Suppose we have a stream of T examples, some of whose labels are queried. The consistency analysis tells us that at the end of the IWAL algorithm, the classifier output by IWAL is comparable to the classifier that would have been chosen by a passive supervised learning algorithm that saw all T labels. The remaining problem is how many labels of those T examples are queried by IWAL. In other words, what is the label complexity of IWAL? This subsection is devoted to proving that IWAL can yield substantial improvement on label complexity over passive learning.

The bound for this algorithm depends critically on a particular quantity, called the disagreement coefficient [54], which depends upon the hypothesis space and the example distribution. Note that the original disagreement coefficient is defined for the $0-1$ loss function. Here we need a general disagreement coefficient [12] for general loss functions $\ell(z,y)$. The definition of disagreement coefficient needs a few new concepts.

Definition 22.4.1 *[12] The pseudo-metric on H induced by D is defined as*

$$\rho(h_1,h_2) = \mathbb{E}_{x\sim D} \max_y |\ell(h_1(x),y) - \ell(h_2(x),y)| \tag{22.9}$$

for $h_1, h_2 \in H$. Let $B(h,r) = \{h' \in H : \rho(h,h') \leq r\}$ be the ball centered around h of radius r.

Definition 22.4.2 *[12](disagreement coefficient) The disagreement coefficient is the infimum value of θ such that for all r*

$$\sup_{h \in B(h^*,r)} \rho(h^*,h) \leq \theta r. \tag{22.10}$$

Definition 22.4.3 *[12] The slope asymmetry of a loss function ℓ is*

$$C_l = \sup_{z,z'} |\frac{\max_y \ell(z,y) - \ell(z',y)}{\min_y \ell(z,y) - \ell(z',y)}|. \tag{22.11}$$

Theorem 6 *[12] For all distributions D, for all finite hypothesis classes H, if the loss function has a slope asymmetry C_l, and the learning problem has a disagreement coefficient θ, the expected number of labels queried by IWAL after seeing T examples is at most*

$$4\theta C_l \left(L^* T + O\left(\sqrt{T \log \frac{|H|T}{\delta}} \right) \right). \tag{22.12}$$

22.5 Dependency-Oriented Data Types for Active Learning

Active learning problems can arise in the context of different kinds of complex data types such as sequences and graphs. Since these data types often have implicit or explicit dependencies between instances, this can impact the active learning process directly. This is because the dependencies between the instance translates to correlations in the node labels among different instances.

The two primary dependency-oriented data types are sequential data and graph-based data. Graph-based learning poses different kinds of challenges because of the wide variety of scenarios in which it can arise, such as the classification of many small graphs, or the classification of nodes in a single large graph. This section will discuss sequences and graphs, the two data types that are frequently explored in the context of dependency-oriented active learning.

22.5.1 Active Learning in Sequences

Sequences are defined as a succession of discrete values (tokens) $X_1 \ldots X_N$, each of which is possibly associated with a label. The problem of sequence labeling forms the core of all information extraction tasks in natural language processing, which attempt to annotate the individual words (tokens) with labels. For example, consider the following sentence:

"The bomb blast in Cairo $<$ *place* $>$ left Mubarak $<$ *person* $>$ wounded."

In this case, the token "Cairo" is associated with the label *place*, whereas the token "Mubarak" is associated with the label *person*. Features may also be associated with tokens of the instance, which provide useful information for classification purposes. The labels associated with the individual tokens in a sequence are not independent of one another, since the adjacent tokens (and labels) in a given sequence directly impact the label of any given token. A common approach for sequence classification is to use Hidden Markov Models (HMM), because they are naturally designed to address the dependencies between the different data instances. Another class of methods that are frequently used for active learning in sequences are Conditional Random Fields (CRF). A detailed discussion of these methods is beyond the scope of this chapter. The reader is referred to the work in [8, 28, 60, 102, 106], which provide a number of different probabilistic algorithms for active learning from sequences. The work in [106] is particularly notable, because it presents an experimental comparison of a large number of algorithms.

22.5.2 Active Learning in Graphs

Graphs represent an important scenario for active learning, since many real world networks such as the Web and social networks can be represented as graphs. In many applications involving such networks, labeled data is not readily available, and therefore active learning becomes an important approach to improve the classification process. Graph-based learning is studied in different scenar-

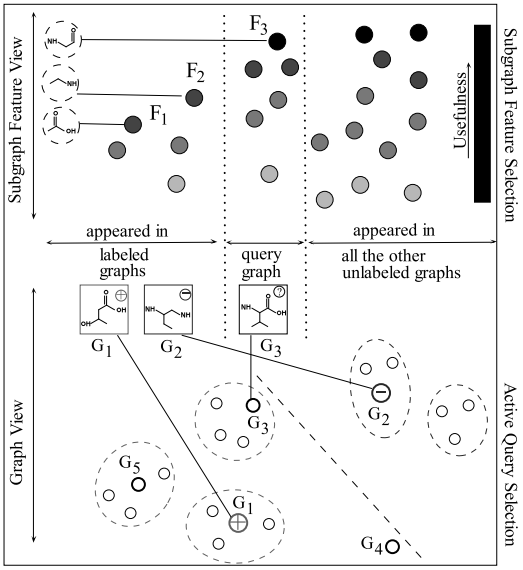


FIGURE 22.2 (See color insert.): Motivation of active learning on small graphs [70].

ios, such as the classification of many small graphs (e.g., chemical compounds) [70], or the classification of nodes in a single large graph (e.g., a social or information network) [10, 18, 49–52, 61]. Most of the specialized active learning methods in the literature are designed for the latter scenario, because the former scenario can often be addressed with straightforward adaptations of standard active learning methods for multidimensional data. However, in some cases, specific aspects of feature selection in graphs can be combined with active learning methods. Such methods will be discussed below.

22.5.2.1 Classification of Many Small Graphs

The first scenario addresses the problem of classifying graph objects within a graph database. Graphs are ubiquitous and have become increasingly important in modeling diverse kinds of objects. For many real-world classification problems, the data samples (i.e., instances) are not directly represented as feature vectors, but graph objects with complex structures. Each graph object in these applications can be associated with a label based upon certain properties. For example, in drug discovery, researchers want to discover new drugs for certain diseases by screening the chemical database. Each drug candidate, i.e., a chemical compound, is naturally represented as a graph, where the nodes correspond to the atoms and the links correspond to the chemical bonds. The chemical compound can be associated with a label corresponding to the drug activities, such as anticancer efficacy, toxicology properties, and Kinase inhibition. Label acquisition can often be expensive and time-consuming because the properties of compounds may often need to be derived through a variety of chemical processing techniques. For example, in molecular drug discovery, it requires time, efforts, and excessive resources to test drugs' anti-cancer efficacies by pre-clinical studies and clinical trials, while there are often copious amounts of unlabeled drugs or molecules available from various sources. In software engineering, human experts have to examine a program flow in order to find software bugs. All these processes are rather expensive. This is how active learning comes into play. Active learning methods could be used to select the most important drug candidates for the human experts to test the label in the lab, and could potentially save a significant amount of money and time in the drug discovery process.

Conventional active learning methods usually assume that the features of the instances are given beforehand. One issue with graph data is that the features are not readily available. Many classification approaches for small graphs require a feature extraction step to extract a set of subgraphs that are discriminative for the classification task [63,69,114,121,123]. The number of possible subgraph features that can be extracted is rather large, and the features extracted are highly dependent on the labeled instances. In the active learning settings, we can only afford to query a small number of graphs and obtain their labels. The performance of the feature extraction process depends heavily on the quality of the queried graphs in the active learning process. Meanwhile, in the active learning, we need to evaluate the importance of each instance. The performance of active learning also depends heavily on the quality of the feature extraction process. Thus the active learning and subgraph feature extraction steps are mutually beneficial.

For example, in Figure 22.2, we are given two labeled graphs (G_1 and G_2) within which we have only a small number of the useful subgraph features (F_1 and F_2). If we query the graph G_3 for the label, we are not only improving the classification performances due to the fact that G_3 is both representative and informative among the unlabeled graphs, but we are also likely to improve the performances of feature extraction, because G_3 contains new features like F_3.

The process of query selection can assist the process of feature selection in finding useful subgraph features. In other words, the better the graph object we select, the more effectively we can discover the useful subgraph features. Therefore, the work in [70] couples the active sample problem and subgraph feature selection problem, where the two processes are considered simultaneously. The idea is that the two processes influence each other since sample selection should affect feature selection and vice-versa. A method called gActive was proposed to maximize the dependency between subgraph features and graph labels using an active learning framework. A branch-and-bound algorithm is used to search for the optimal query graph and optimal features simultaneously.

22.5.2.2 Node Classification in a Single Large Graph

A second scenario addresses the problem of node classification in a single large graph. This is a very common problem in social networks, where it may be desirable to know specific properties of nodes for the purpose of influence analysis. For example, a manufacturer of a particular sporting item may wish to know nodes that correspond to interest in that sport. However, this is often difficult to know a-priori, since the interests of nodes may often need to be acquired through some querying or manual examination mechanism, which is expensive. The work in [10, 18, 49–52, 61] actively queries the labels of the nodes in order to maximize the effectiveness of classification. A particular class of active learning methods, which are popular in the graph scenario are *non-adaptive active learning methods*, in that the previous labels are not used to select the examples, and therefore the selection can be performed in batch in an unsupervised way. The downside is that the learning process is not quite as effective, because it does not use labels for the selection process. Many active learning methods for graphs [49, 51, 52, 61] belong to this category, whereas a few methods [10, 11, 18, 50] also belong to the adaptive learning category. The non-adaptive methods are generally unsupervised and attempt to use representativeness-based models such as clustering in order to select the nodes. Many of the non-adaptive methods are motivated by experimental design approaches [61, 126].

An important aspect of non-adaptive active learning of node labels in large graphs is that the structural properties of the graph play an important role in the selection of nodes for the active learning process. This is because labels are not available, and therefore only the structure of the graph may be used for the selection process. For example, the work in [15] showed that the prediction error is small when the graph cut size is large. This suggests that it is better to label nodes, such that the size of the graph cut on the labeled graph is large. The work in [51, 52] proposes a heuristic label selection method that maximizes the graph cut. Furthermore, a generalized error bound was proposed by replacing the graph cut with an arbitrary symmetric modular function. This was

then leveraged to provide an improved algorithm for the use of submodular function maximization techniques. The active learning method first clusters the graph and then randomly chooses a node in each cluster. The work in [61] proposes a variance minimization approach to active learning in graphs. Specifically, the work in [61] proposes to select the most informative nodes, by proposing to minimize the prediction variance of the Gaussian field and harmonic function, which was used in [133] for semi-supervised learning in graphs. The work in [49] approaches the problem by considering the generalization error of a specific classifier on graphs. The Learning with Local and Global Consistency (LLGC) method proposed in [135] was chosen because of its greater effectiveness. A data-dependent generalization error bound for LLGC was proposed in [49] using the tool of transductive Rademacher Complexity [35]. This tool measures the richness of a class of real-valued functions with respect to a probability distribution. It was shown in [49] that the empirical transductive Rademacher complexity is a good surrogate for active learning on graphs. The work therefore selects the nodes by minimizing the empirical transductive Rademacher complexity of LLGC on a graph. The resulting active learning method is a combinatorial optimization problem, which is optimized using a sequential optimization algorithm.

Another line of research [10, 11, 18, 50] has considered adaptive active learning, where the labels for the nodes of a graph are queried and predicted in an iterative way with the use of a trained classifier from the previous labels. The work in [11] made the observation that label propagation in graphs is prone to "accumulating" errors owing to correlations among the node labels. In other words, once an error is made in the classification, this error propagates throughout the network to the other nodes. Therefore, an acquisition method is proposed, which learns where a given collective classification method makes mistakes, and then suggests acquisitions in order to correct those mistakes. It should be pointed out that such a strategy is unique to active learning in graphs, because of the edge-based relationships between nodes, which result in homophily-based correlations among node labels. The work in [11] also proposes two other methods, one of which greedily improves the objective function value, and the other adapts a viral marketing model to the label acquisition process. It was shown in [11] that the corrective model to label acquisition generally provides the best results.

The work in [10] works in scenarios where both content and structure are available with the nodes. It is assumed that the labels are acquired sequentially in batch sizes of k. The algorithm uses two learners called *CO* (content-only) and *CC* (collective classifier), and combines their predictions in order to make decisions about labeling nodes. In particular, nodes for which these two classifiers differ in their prediction are considered good candidates for labeling, because the labels of such nodes provide the most informative labels for knowing the relative importance of the different aspects (content and structure) over different parts of the data. The proposed algorithm *ALFNET* proceeds by first clustering the nodes using the network structure of the data. Then, it selects the k clusters that satisfy the following two properties:

- The two learners *CC* and *CO* differ the most in their predictions.
- The predictions of the classifiers have a distribution that does not match the distribution of the observed labels in the cluster.

An overall score is computed on the basis of these two criteria. For a batch size of k, the top-k clusters are selected. One node is selected from each of these clusters for labeling purposes.

The work in [18] proposes an active learning method that is based on the results of [51, 52] in order to design optimal query placement methods. The work in [18] proposes a method for active learning in the special case of trees, and shows that the optimal number of mistakes on the non-queries nodes is made by a simple mincut classifier. A simple modification of this algorithm also achieves optimality (within a constant factor) on the trade-off between the number of mistakes, and the number of non-queried nodes. By using spanning trees, the method can also be generalized to arbitrary graphs, though the problem of finding an optimal solution in this general case remains open.

The work in [50] proposes selective sampling for graphs, which combines the ideas of online learning and active learning. In this case, the selective sampling algorithm observes the examples in a sequential manner, and predicts its label after each observation. The algorithm can, however, choose to decide whether to receive feedback indicating whether the label is correct or not. This is because, if the label can already be predicted with high confidence, a significant amount of effort can be saved by not receiving feedback for such examples. The work in [50] uses the LLGC framework [135] for the prediction process.

22.6 Advanced Methods

Active learning can arise in the context of different scenarios, such as feature-based learning, streaming data, or other kinds of query variants. In feature-based learning, it is desirable to learn techniques for acquiring *features* rather than instance labels for classification. These different scenarios will be discussed in detail in this section.

22.6.1 Active Learning of Features

Most of the discussion in the previous sections focussed on cases where new instances are acquired in order to maximize the effectiveness of a given learner. However, there are also numerous scenarios where the labels of the instances are known, but many feature values are unknown, and require a cost of acquisition. In such scenarios, when we query the features for some instances, the "oracle" can provide their corresponding feature values. It is assumed that the values for some features of an instance are rather expensive to obtain. In some very complex scenarios, knowledge of the feature values *and* the labels may be incomplete, and either may be acquired at a given cost. Furthermore, in feature-based active learning, the acquisition of more features may be done either for the training data or for a test instance, or for both, depending upon the underlying application. In some applications, the training data may be incompletely specified in a non-homogeneous way over different instances, and the problem may be to determine the subset of features to query for a given test instance in order to maximize accuracy. In other application, the same feature may be collected for both the training data and the test data at a particular cost. An example of this case would be the installation of a sensor at a particular location, which is helpful for augmenting both the training data and the test data. Some examples of feature-based active learning scenarios are as follows:

- *Medical Applications:* Consider a medical application, where it is desirable to collect information from different medical tests on the patient. The results of each medical test may be considered a feature. Clearly, the acquisition of more features in the training data helps in better modeling, and the acquisition of more features in the test data helps in better diagnosis. However, acquisition of such features is typically expensive in most applications.
- *Sensor Applications:* In sensor applications, the collection and transmission of data is often associated with costs. In many cases, it is desirable to keep certain subsets of sensors passive, if their data is redundant with respect to other sensors and do not contribute significantly to the inference process.
- *Commercial Data Acquisition:* In many commercial applications (*e.g.*, marketing, and credit card companies), different features about customers (or customer behavior) can be acquired at a cost. The acquisition of this information can greatly improve the learning process. Therefore, feature-based active learning can help evaluate the trade-offs between acquiring more features, and performing effective classification.

Feature-based active learning is similar in spirit to instance-based active learning, but the methodologies used for querying are quite different.

One observation about missing feature values is that many of them can often be partially imputed from the correlations among the features in the existing data. Therefore, it is not useful to determine those features that can be imputed. Based on this principle, a technique was proposed in [132] in which missing feature values are first imputed. The ones among them that have the greatest uncertainty are then queried. A second approach proposed in this work uses a classifier to determine those instances that are misclassified. The feature values of the misclassified instances are then queried for labels. In incremental feature acquisition, a small set of misclassified instances are used in order to acquire their labels [88], or by using a utility function that needs to be maximized [87, 101].

Active learning for feature values is also relevant in the context of scenarios where the feature values are acquired for test instances rather than training instances. Many of the methodologies used for training instances cannot be used in this scenario, since class labels are not available in order to supervise the training process. This particular scenario was proposed in [48]. Typically standard classifiers such as naive Bayes and decision trees are modified in order to improve the classification [19, 37, 79, 110]. Other more sophisticated methods model the approach as a sequence of decisions. For such sequential decision making, a natural approach is to use Hidden Markov Models [62].

Another scenario for active learning of features is that we can actively select instances for feature selection tasks. Conventional active learning methods are mainly designed for classification tasks. The works in [80–82] studied the problem of how to selectively sample instances for feature selection tasks. The idea is that by partitioning the instances in the feature space using a KD-tree, it is more efficient to selectively sample the instances with very different features.

22.6.2 Active Learning of Kernels

Active kernel learning is loosely related to the active learning of features in the previous subsection. Most of the discussion in the previous sections focuses on designing selective sampling methods that can maximize the effectiveness of a given supervised learning method, such as SVM. Kernel-based methods are very important and effective in supervised learning, where the kernel function/matrix, instead of the feature values, plays an important role in the learning process. Since the choice of kernel functions or matrices is often critical to the performance of kernel-based methods, it becomes a more and more important research problem to automatically learn a kernel function/matrix for a given dataset. There are many research works on kernel learning, i.e., the problem of learning a kernel function or matrix from *side information* (e.g., either labeled instances or pairwise constraints). In previous kernel learning methods, such *side information* is assumed to be given beforehand. However, in some scenarios, the side information can be very expensive to acquire, and it is possible to actively select some side information to query the oracle for the ground-truth. This is how active kernel learning comes into play.

Specifically, active kernel learning corresponds to the problem of exploiting active learning to learn an optimal kernel function/matrix for a dataset with the least labeling cost. The work in [56] studied the problem of selectively sampling instance-pairs, instead of sampling instances, for a given dataset, in order to learn a proper kernel function/matrix for kernel-based methods. Given a pair of instances (x_i, x_j), the corresponding kernel value is $K_{i,j} \in \mathbb{R}$, indicating the similarity between x_i and x_j. If the two instances are in the same class, $K_{i,j}$ is usually a large positive number, while if the two instances are in different classes, $K_{i,j}$ is more likely to be a large negative number. One simple solution for active kernel learning is to select the kernel similarities that are most close to zero, i.e., $(i^*, j^*) = \arg\min_{(i,j)} |K_{i,j}|$. This solution is simply extended from the uncertainty-based active learning methods. The underlining assumption is that such instance pairs are the most informative in the kernel learning process. However, in real-world kernel-based classification methods, the instance

pairs with the smallest absolute kernel values $|K_{i,j}|$ are more likely to be *must-link pairs*, i.e., both instances are in the same class. It is also crucial to query instances pairs that are more likely to be *cannot-link pairs*, where the two instances are in different classes.

The work in [56] proposed an approach based upon *min-max* principle, i.e., the assumption is that the most informative instance pairs are those resulting in a large classification margin regardless of their assigned class labels.

22.6.3 Active Learning of Classes

In most active learning scenarios, it is assumed that the instances are queried for labels. In such querying processes, we can control which instances we want to query for the labels, but cannot control which classes we will get for the queried instances. It is explicitly or implicitly assumed that: 1) The cost of getting the class label for an instance is quite expensive. 2) The cost of obtaining the instances is inexpensive or free. However, in some scenarios, such assumptions may not hold, and it may be possible to query with a *class label*, so that the oracle will feedback with some instances in the class. Thus, this is a kind of reverse query, which is relevant to certain kinds of applications [83]. For example, the work in [83] designs method for training an artificial nose which can distinguish between different kinds of vapors (the class labels). However, the key distinction here is that the instances here are *generated* for training a particular application. For example, in the "artificial nose" task, the chemical (the instances) are synthesized. Different kinds of instances are synthetically generated representing the different kinds of vapors. In traditional active learning, the instances are already *existing*. A key observation in [83] is that methods that optimize the class stability were more effective than either random sampling or methods that optimized the class accuracy.

22.6.4 Streaming Active Learning

The data stream scenario is particularly suited to active learning, since large volumes of data are often available, and the real-time stream generation process usually does not create labels of interest for specific applications. Unlike static data sets, which have often been extensively processed in order to generate labels, data stream instances are often "newly generated," and therefore there has never been an opportunity to label them. For example, in a network system, the incoming records may correspond to real time intrusions, but this cannot be known without spending (possibly manual) effort in labeling the records in order to determine whether or not they correspond to intrusions. Numerous simple methods are possible for adapting data streams to the active learning scenario. In local uncertainty sampling, active learning methods can be applied to a current segment of the data stream, with the use of existing algorithms, without considering any other segments. The weakness of this is that it fails to use any of the relevant training data from the remaining data stream. One way to reduce this is to build historical classifiers on different segments, and combine the results from the current segment with the results from the historical classifier. The work in [136] goes one step further, and combines an ensemble technique with active learning in order to perform the classification. The minimal variance principle is used for selecting the relevant sample.

The work in [26] proposes unbiased methods for active sampling in data streams. The approach designs optimal instrumental distributions to minimize the variance in the sampling process. Bayesian linear classifiers with weighted maximum likelihood are optimized online on order to estimate parameters. The approach was applied on a data stream of user-generated comments on a commercial news portal. Offline evaluation was used in order to compare various sampling strategies, and it was shown that the unbiased approach is more effective than the other methods.

The work in [41] introduces the concept of demand-driven active mining of data streams. In this model, it is assumed that a periodic refresh of the labels of the stream is performed in batch. The key question here is the choice of the timing of the model for refreshing. If the model is refreshed too slowly, it may result in a more erroneous model, which is generally undesirable. If the model is

refreshed too frequently, then it will result in unnecessary computational effort without significant benefit. Therefore, the approach uses a mathematical model in order to determine when the stream patterns have changed significantly enough for an update to be reasonably considered. This is done by estimating the error of the model on the newly incoming data stream, without knowing the true labels. Thus, this approach is not, strictly speaking, a traditional active learning method that queries for the labels of individual instances in the stream.

The traditional problem of active learning from data streams, where a decision is made on each instance to be queried, is also referred to as *selective sampling*, which was discussed in the introduction section of this chapter. The term "selective sampling" is generally used by the machine learning community, whereas the term "data stream active learning" is often used by data mining researchers outside the core machine learning community. Most of the standard techniques for sample selection that are dependent on the properties of the unlabeled instance (e.g., query-by-committee), can be easily extended to the streaming scenario [43,44]. Methods that are dependent on the aggregate performance of the classifier are somewhat inefficient in this context. This problem has been studied in several different contexts such as part-of-speech tagging [28], sensor scheduling [71], information retrieval [127], and word-sense disambiguation [42].

22.6.5 Multi-Instance Active Learning

The idea in multiple instance active learning [33] is that the instances are naturally organized into "bags," each of which is assigned a label. When a label is assigned to a bag, it is implied that all instances in that bag have that label. This problem was first formalized in [107]. This problem arises quite naturally in some data domains such as protein family modeling, drug activity prediction, stock market prediction, and image retrieval. In these data domains, scenarios arise in which it is inexpensive to obtain labels for bags, but rather expensive to obtain labels at the individual instance level. The real issue here is often one of granularity. For example, in a text document, image, or a biological sequence, consider the case where individual instances correspond to portions of these objects. In such cases, it may often be easier to label the whole object, rather than portions of it, and all segments of the object inherit this label. In many cases, the labeling may simply be *available* at the higher level of granularity. It should be pointed out that multiple instance labeling naturally leads to errors in labeling of the fine-grained instances. Therefore, a major challenge in multiple-instance learning is that the learner must determine which instances of a particular label in a bag do actually belong to that label. On the flip side, these coarse labelings are relatively inexpensive, and may be obtained at a low cost.

Several scenarios have been discussed in [107] for the multi-class problem. For example, the learner may either query for bag labels, for specific instance labels, or for all instance labels in a specific bag. The appropriateness of a particular scenario of course depends upon the application setting. The work in [107] focuses on the second scenario, where specific instance labels are queried. In addition, a binary classification problem is assumed, and it is assumed that only positively labeled bags are of interest. Therefore, only instances in positive bags are queried. The approach uses multiple-instance logistic regression for classification purposes. In addition, uncertainty sampling is used, where the instances for which the labels are the least certain are sampled. It should be pointed out that the bag labels directly impact both the logistic regression modeling process, and the computation of the uncertainty values from the labels. Details of the approach are discussed in [107].

A second scenario is the case where each query may potentially correspond to multiple instances. For example, in the case of *Amazon Mechanical Turk*, the user provides a batch of instances to the learner, which are then labeled by experts at a cost. In many cases, this batch is provided in the context of a *cold start*, where no previous labeled instances are available. The cold start scenario is closer to *unsupervised* or *non-adaptive* active learning, where no labeled data is available in order to evaluate the performance of the methods. Some examples of this scenario have already been dis-

cussed in the context of active learning for node classification in networks, where such methods are very popular. Even in the case where labeled data is available, it is generally not advisable to pick the k best instances. This is because the k best instances that are picked may all be quite similar, and may not provide information that is proportional to the underlying batch size. Therefore, much of the focus of the work in the literature [16,55,57,119] is to incorporate diversity among the instances in the selection process.

22.6.6 Multi-Label Active Learning

Conventional active learning approaches mainly focus on single-label classification problems, i.e., binary classification and multi-class classification. It is usually assumed that each instance can only be assigned to *one* class among all candidate categories. Thus the different classes are mutually exclusive in the instance level. However, in some applications, such assumptions may not hold. Each instance can be associated with multiple labels simultaneously, where the classification task is to assign *a set* of labels for each instance. For example, in text classification, each text document may be assigned with multiple categories, e.g., news, politics, economy. In such classification tasks, when we query the "oracle" with an instance, the "oracle" will annotate all the labels for the queried instance. In the active learning process, we need to estimate the importance or uncertainty of an instance corresponding to all the labels.

One solution is to measure the informativeness of each unlabeled instance across all the candidate labels [17, 38, 78, 112]. For example, Yang et. al. [124] proposed an approach that selects the unlabeled data based upon the largest reduction of the expected model loss for multi-label data by summing up losses on all labels. In multi-label learning, it is also important to consider the relationship among different labels instead of treating each label independently. A second type of approaches considers the label structure of a multi-label learning problem, such as the correlations among the labels [98], the cardinality of label sets [77], and hierarchy of labels [77], to measure the informativeness of the unlabeled instances. The work in [76] proposed two methods based upon max-margin prediction uncertainty and a label cardinality inconsistency, respectively, and integrated them into an adaptive method of multi-label active learning.

22.6.7 Multi-Task Active Learning

Multi-task learning is closely related to multi-label learning scenarios. Conventional active learning mainly focuses on the scenarios of single task with single learner. However, in some applications, one instance can be involved in multiple tasks and be used by multiple learners [99]. For example, in NLP (Natural Language Processing) tasks, each instance can be involved in multiple levels of linguistic processing tasks, such as POS tagging, named entity recognition, statistical parsing. It is usually easier for the "oracle" to be queried with one instance, and feedback with labels for all the tasks, instead of being queried multiple times for different tasks. In this scenario, the key problem is how to assess the informativeness or representativeness of an unlabeled instance w.r.t. all the learning tasks. Some examples of this scenario have already been discussed in the context of active learning for NLP applications [99], which usually involve multiple tasks and the labeling costs are very high.

In the work [99], two approaches have been proposed. In the first approach, i.e., alternating selection, the parser task and named entity recognition task take turns to select important instances to query for the label iteratively. While the second approach, i.e., rank combination, combines the ranking of importance from both tasks and select important instances that are highly ranked in both tasks.

Another scenario for multi-task active learning is that the outputs of multiple tasks are coupled together and should satisfy certain constraints. For example, in relational learning, the entities classification and link prediction are two prediction tasks, where their outputs can often be related.

If an entity is predicted as the mayor of a city, the class label of the entity should be "politician" or "person", instead of other labels like "water." In such scenarios, the outputs of multiple tasks are coupled with certain constraints. These constraints could provide additional knowledge for the active learning process. The work in [128] studied the problem of active learning for multi-task learning problem with output constraints. The idea is that we should exploit not only the uncertainty of prediction in each task but also the inconsistency among the outputs on multiple tasks. If the class labels on two different tasks are mutually exclusive while the model predicts positive labels in both tasks, it indicates that the model makes an error on one of the two tasks. Such information is provided by the output constraints and can potentially help active learners to evaluate the importance of each unlabeled instance more accurately.

22.6.8 Multi-View Active Learning

Most of the discussion in the previous sections focuses on single view settings, where each instance only has one view of a set of features. However, there are also numerous scenarios where each instance is described with several different disjoint sets of features. Each set of features corresponds to one *view*, which is sufficient for learning the label concept. For example, in the Web page classification problems, we can have two different views: 1) One set of features corresponds to the text appearing on a Web page. 2) Another set of features corresponds to the anchor text in the hyper-links pointing to the Web page. Multi-view active learning is closely related to semi-supervised learning and active learning [90, 116]. The multi-view active learning problem was first studied in [90–92], where a model called co-testing was proposed. The idea is related to Query-by-Committee, which uses multiple learners and selects the set of unlabeled instances where the predictions on different learners disagree with each other. However, Query-by-Committee approaches build multiple learners on one single view, while the co-testing method focuses on multiple views and trains one learner on each view. The motivation of such model design is as follows: Suppose learners in different views predict different labels on an unlabeled instance. These kinds of unlabeled instances are called *contention points*. At least one learner could make a mistake on that instance prediction. If we query the oracle with that instance (i.e., a contention point), it is guaranteed that at least one of the learners will benefit from the label information. In this sense, the active sampling is based upon learning from mistakes (at least in one view). So the co-testing method can potentially converge to the target concept more quickly than alternative active learning methods.

The work in [116] provided a theoretical analysis on the sample complexity of multi-view active learning based upon the α-expansion assumption [7]. The sample complexity of multi-view active learning can be exponentially improved to $\widetilde{O}(\log \frac{1}{\epsilon})$, approximately. The work in [117] extended the theoretical analysis to *Non-Realizable* cases, where the data are no longer assumed to be perfectly separable by any hypothesis in the hypothesis class because of the noise.

22.6.9 Multi-Oracle Active Learning

In conventional active learning settings, it is usually assumed that there is only one "oracle" for the system to query the labels, and the "oracle" knows ground-truth about classification tasks and can annotate all the queried samples without any errors. These assumptions may not hold in some real-world applications, where it is very hard or impossible to find such an "oracle" that knows everything about the ground-truth. It is more likely that each "oracle" (i.e., annotator) only knows or is familiar with a small part of the problem, while we could have access to multiple "oracles" with varying expertise. For example, in *Crowdsourcing* systems, such as Amazon Mechanical Turk,[3] each user or annotator may only know a small part of the task to be queried while we could query multiple users, each with different queries. In many other applications, many "oracles" may disagree

[3] `http://www.mturk.com`

with each other on the same instance. The answer from an individual "oracle" may not be accurate enough. Combining answers from multiple "oracles" would be necessary. For example, in medical imaging, the diagnosis of different doctors may be different on one patient's image, while it is often impossible to perform a biopsy on the patient in order to figure out the ground-truth.

In multi-oracle active learning, we could not only select which instance to query for the label, but also select which oracle(s) or annotator(s) to be queried with the instance. In other words, multi-oracle active learning is an instance-oracle-pair selection problem, instead of an instance-selection problem. This paradigm brings up new challenges to the active learning scenario. In order to benefit the learning model the most, we need to estimate both the importance of each unlabeled instance and the expertise of each annotator.

The work in [122] designed an approach for multi-oracle active learning. The model first estimated the importance of each instance, which is similar to conventional active learning models. The model then estimated which "oracle" is most confident about its label annotation on the instance. These two steps are performed iteratively. The high level idea of the approach is that we first decide which instance we need to query the most, then decide who has the most confident answer for the query.

The work in [111] studied both the uncertainty in the model and noises in the oracle. The idea is that the costs of labelers with low qualities are usually low in crowdsourcing applications, while we can repeatedly query different labelers for one instance to improve th labeling quality with additional costs. Thus the work [111] studied the problem of selective repeated labeling, which can potentially reduce the cost in labeling while improving the overall labeling quality, although this work focused on the cases that all oracles are equally and consistently noisy, which may not fit in many real-world cases. The work [32] extended the problem setting by allowing labelers to have different noise levels, though the noise levels are still consistent over time. It shows that the qualities of individual oracles can be properly estimated, then we can perform active learning more effectively by querying only the most reliable oracles in selective labeling process. Another work extended the setting of multi-oracle active learning in that the skill sets of multiple oracles are not assumed to be static, but are changeable by teaching each other actively. The work in [39] explored this problem setting, and proposed a self-taught active learning method from crowdsourcing. In addition to selecting the most important instance-oracle-pair, we also need to figure out which pair of oracles could be the best pair for self-teaching, i.e., one oracle has the expertise in a skill where the other oracle needs to improve the most. In order to find the most effective oracle pair, we not only need to estimate how well one oracle is on a type of skill, but also which oracle needs the most help on improving the type of skill. In this way, the active learning method can help improve the overall skill sets of the oracles and serve the labeling task with better performance.

22.6.10 Multi-Objective Active Learning

Most previous works on active learning usually involved supervised learning models with single objective optimization. The selective sampling process is mainly performed in the level of data instances, i.e., it is assumed that the labeling of the instances is very expensive. However, there are some scenarios where the learning problem involve multiple objectives in the optimization process. Each model will have multiple scores with reference to multiple objective functions. For example, in classification problems, the performance of the model can be evaluated from multiple objective/loss functions, such as empirical loss, model complexity, precision, recall, and AUC (for ROC curve). In hardware design, each model design can have different scores on multiple criteria, such as energy consumption, throughput, and chip area. In all these scenarios, the models are not evaluated according to one single objective function, but a set of multiple objective functions.

Usually there is no single model that is the best model in all objectives. Different "optimal" solutions/models can have different trade-offs on the multiple objective functions. For example, model M_1 may have better performance on "energy consumption" than another model M_2, while

M_2 may have better performance on "throughput" than model M_1. In this case, we call a model a non-dominated solution or "optimal" solution, if there is no other solution that can achieve better performance in all objectives than the model. Thus the task of multi-objective optimization is to obtain a set of multiple "optimal" models/solutions, such that all the models in the set are "optimal" and cannot be dominated by any other solution or dominate each other. Such a model set is called a "Pareto-optimal set," where there is a potentially infinite number of "optimal" solutions in the model space. It is usually infeasible to get a set of Pareto-optimal models by exhaustive search.

Moreover, in some real-world applications, the evaluation of the performance of a model/solution with reference to multiple criteria is not free, but can be rather expensive. One example is the hardware design problem. If we would like to test the performance of a model design in hardware, synthesis of only one design can take hours or even days. The major cost of the learning process is on the model evaluation step, instead of the label acquiring step. This is how active learning comes into play. We could perform active learning in the model/design space, instead of the data instance space. The work in [138] studied the problem of selectively sampling the model/design space to predict the Pareto-optimal set, and proposed a method called Pareto Active Learning (PAL). The performance on multiple objectives is modeled as a Gaussian process, and the designs are estimated to iteratively select the models that can maximize the model searching process.

22.6.11 Variable Labeling Costs

In the previous discussion, we considered cases where the costs are fixed for each training instance. This may, however, often not be the case, where some instances may be harder to label than others. For example, consider the case where the labels correspond to information extracted from text documents. In practice, the documents may correspond to varying lengths and complexity, and this may reflect the variable labeling cost associated with these documents. This interesting scenario has been discussed in [66, 67, 86, 108]. The difference in these many scenarios is in how the labeling cost is defined, and whether it is known prior to the learning process. The work in [108] provides a detailed experimental study of the annotation times over different instances in different domains, and shows that these times can be significantly different, and can also depend on the underlying data domain.

22.6.12 Active Transfer Learning

Active learning and transfer learning are two different ways to reduce labeling costs in classification problems. Most of the previous works discussed these two problem settings separately. However, in some scenarios, it can be more effective to combine the two settings via active transfer learning. Transfer learning focuses on exploiting the labeled data from a related domain (called source domain/dataset) to facilitate the learning process in a target domain/dataset. The basic idea is that if the source dataset is very similar to the target dataset, we can save a lot of labeling costs in the target domain by transferring the existing knowledge in the source domain to the target domain. On the other hand, active learning focuses on selective sampling of unlabeled instances in the target domain.

The work in [118] first studied the problem of combining active learning with transfer learning. The idea is that both transfer learning and active learning have their own strengths and weakneses. It is possible to combine them to form a potentially better framework. In active transfer learning, the transfer learner can have two different choices during the domain knowledge transferring. One choice is to directly transfer an instance from the source domain to the target domain following conventional transfer learning methods. A second choice is to query the oracle with an instance for the label instead of directly transfering the labels.

The work in [137] studied a second scenario where the unlabeled data in the target domain is not available. Only the source domain has unlabeled instances. The task is to actively select some

unlabeled instances from the source domain and query the oracle with the instance. The queried instances should be able to benefit the target domain in the transfer learning process. The work in [20] integrated the active learning and transfer learning into one unique framework using a single convex optimization. The work in [125] provided a theoretical analysis on active transfer learning.

The work in [40] extended the active transfer learning problem with the multi-oracle settings, where there can be multiple oracles in the system. Each oracle can have different expertise, which can be hard to estimate because the quantity of instances labeled by all the participating oracles is rather small. The work in [131] extended the active transfer learning framework into the recommendation problems, where the knowledge can be actively transfered from one recommendation system to another recommendation system.

22.6.13 Active Reinforcement Learning

Active learning has also been widely used beyond supervised learning. In reinforcement learning, there are many scenarios where active learning could help improve the performance while reducing human efforts/costs during the learning process. The work in [36] studied the problem of active reinforcement learning, where transition probabilities in a Markov Decision Process (MDP) can be difficult/expensive for experts to specify. The idea is to combine offline planning and online exploration in one framework that allows oracles to specify possibly inaccurate models of the world offline.

The work in [65] studied the active imitation learning that can be reduced to I.I.D active learning. Imitation learning usually learns a target policy by observing full execution trajectories, which can be hard/expensive to generate in practice. The idea is that we could actively query the expert about the desired action at individual states, based upon the oracle's answers to previous queries.

22.7 Conclusions

This chapter discusses the methodology of active learning, which is relevant in the context of cases where it is expensive to acquire labels. While the approach has a similar motivation to many other methods, such as semi-supervised learning and transfer learning, the approach used for dealing with the paucity of labels is quite different. In active learning, instances from the training data are modeled with the use of a careful sampling approach, which maximizes the accuracy of classification, while reducing the cost of label acquisition. Active learning has also been generalized to structured data with dependencies such as sequences and graphs. The problem is studied in the context of a wide variety of different scenarios such as feature-based acquisition and data streams. With the increasing amounts of data available in the big-data world, and the tools now available to us through crowd sourcing methodologies such as Amazon Mechanical Turk, batch processing of such data sets has become an imperative. While some methodologies still exist for these scenarios, a significant scope exists for the further development of technologies for large scale active learning.

Bibliography

[1] N. Abe and H. Mamitsuka. Query learning strategies using boosting and bagging. *ICML Conference*, pages 1–9, 1998.

[2] C. Aggarwal. *Outlier Analysis*, Springer, 2013.

[3] D. Angluin. Queries and concept learning. *Machine Learning*, 2:319–342, 1988.

[4] M. Ankerst, M. Ester, and H.-P. Kriegel. Towards an effective cooperation of the user and the computer for classification, *ACM KDD Conference*, 2000.

[5] F. R. Bach. Active learning for misspecified generalized linear models. In *NIPS*, pages 65–72, 2006.

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

[7] M. F. Balcan, A. Blum, and K. Yang. Co-training and expansion: Towards bridging theory and practice. *NIPS*, pages 89–96, 2005.

[8] J. Baldridge and M. Osborne. Active learning and the total cost of annotation. *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 9–16, 2004.

[9] M. Belkin and P. Niyogi. Semi-supervised Learning on Riemannian Manifolds. *Machine Learning*, 56:209–239, 2004.

[10] M. Bilgic, L. Mihalkova, and L. Getoor. Active learning for networked data. *ICML Conference*, pp. 79–86, 2010.

[11] M. Bilgic and L.Getoor. Effective Label Acquisition for Collective Classification. *ACM KDD Conference*, pages 43–51, 2008.

[12] A. Beygelzimer, S. Dasgupta, and J. Langford. Importance weighted active learning. *ICML Conference*, pages 49–56, 2009.

[13] A. Beygelzimer, D. Hsu, J. Langford, and T. Zhang. Agnostic active learning without constraints. In *NIPS*, pp. 199–207, 2010.

[14] O. Bousquet, S. Boucheron, and G. Lugosi. Introduction to statistical learning theory. In *Advanced Lectures on Machine Learning*, pages 169–207, 2003.

[15] A. Blum and S. Chawla. Learning from labeled and unlabeled data using graph mincuts. *ICML Conference*, pages 19–26, 2001.

[16] K. Brinker. Incorporating diversity in active learning with support vector machines. *ICML Conference*, pages 59–66, 2003.

[17] K. Brinker. On active learning in multi-label classification. *Studies in Classication, Data Analysis, and Knowledge Organization*, Springer, 2006.

[18] N. Cesa-Bianchi, C. Gentile, F. Vitale, and G. Zappella. Active learning on trees and graphs. *COLT*, pages 320–332, 2010.

[19] X. Chai, L. Deng, Q. Yang, and C.X. Ling. Test-cost sentistive naive Bayes classification. *IEEE Conference on Data Mining (ICDM)*, pages 51–60, 2004.

[20] R. Chattopadhyay, W. Fan, I. Davidson, S. Panchanathan, and J. Ye. Joint transfer and batch-mode active learning. *ICML Conference*, pages 253–261, 2013

[21] D. Cohn, L. Atlas, R. Ladner, M. El-Sharkawi, R. Marks II, M. Aggoune, and D. Park. Training connectionist networks with queries and selective sampling. *Advances in Neural Information Processing Systems (NIPS)*, pages 566–573, 1990.

[22] D. Cohn, L. Atlas, and R. Ladner. Improving generalization with active learning. *Machine Learning*, 5(2):201–221, 1994.

[23] D. Cohn, Z. Ghahramani, and M. Jordan. Active learning with statistical models. *Journal of Artificial Intelligence Research*, 4:129–145, 1996.

[24] O. Chapelle, B. Scholkopf, and A. Zien. Semi-Supervised Learning. Vol. 2, *MIT Press*, Cambridge, MA, 2006.

[25] S.F. Chen and R. Rosenfeld. A survey of smoothing techniques for ME models. *IEEE Transactions on Speech and Audio Processing*, 8(1):37–50, 2000.

[26] W. Chu, M. Zinkevich, L. Li, A. Thomas, and B. Tseng. Unbiased online active learning in data streams. *ACM KDD Conference*, pages 195–203, 2011.

[27] A. Culotta and A. McCallum. Reducing labeling effort for structured prediction tasks, *AAAI Conference*, pages 746–751, 2005.

[28] I. Dagan and S. Engelson. Committee-based sampling for training probabilistic classifiers. *ICML Conference*, pages 150–157, 1995.

[29] W. Dai, Y. Chen, G.-R. Xue, Q. Yang, and Y. Yu. Translated learning: Transfer learning across different feature spaces. *Proceedings of Advances in Neural Information Processing Systems*, pages 353–360, 2008.

[30] S. Dasgupta, A. Kalai, and C. Monteleoni. Analysis of perceptron-based active learning. *Learning Theory*, pages 249–263, 2005.

[31] P. Donmez, J. G. Carbonell, and P. N. Bennett. Dual strategy active learning. *ECML conference*, pages 116–127, 2007.

[32] P. Donmez, J. Carbonell, and J. Schneider. Efficiently learning the accuracy of labeling sources for selective sampling. *KDD Conference*, pages 259–268, 2009.

[33] T. Dietterich, R. Lathrop, and T. Lozano-Perez. Solving the multiple-instance problem with axis-parallel rectangles. *Artificial Intelligence*, 89:31–71, 1997.

[34] G. Druck, B. Settles, and A. McCallum. Active learning by labeling features. *EMNLP*, pages 81–90, 2009.

[35] R. El-Yaniv and D. Pechyony. Transductive rademacher complexity and its applications. *Journal of Artificial Intelligence Research*, 35:193–234, 2009.

[36] A. Epshteyn, A. Vogel, G. DeJong. Active reinforcement learning. *ICML Conference*, pages 296–303, 2008.

[37] S. Esmeir and S. Markovitch. Anytime induction of cost-sensitive trees. *Advances in Neural Information Processing Systems (NIPS)*, pages 425–432, 2008.

[38] A. Esuli and F. Sebastiani. Active learning strategies for multi-label text classification. *ECIR Conference*, pages 102–113, 2009.

[39] M. Fang, X. Zhu, B. Li, W. Ding, and X. Wu, Self-taught active learning from crowds. *ICDM Conference*, pages 273–288, 2012.

[40] M. Fang, J. Yin, and X. Zhu. Knowledge transfer for multi-labeler active learning *ECMLP-KDD Conference*, 2013

[41] W. Fan, Y. A. Huang, H. Wang, and P. S. Yu. Active mining of data streams. *SIAM Conference on Data Mining*, 2004.

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

[43] Y. Freund and R.E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1):119–139, 1997.

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

[45] S. Geman, E. Bienenstock, and R. Doursat. Neural networks and the bias/variance dilemma. *Neural Computation*, 4:1–58, 1992.

[46] R. Gilad-Bachrach, A. Navot, and N. Tishby. Query by committee made real. In *NIPS*, 2005.

[47] J. Goodman. Exponential priors for maximum entropy models. *Human Language Technology and the North American Association for Computational Linguistics*, pages 305–312, 2004.

[48] R. Greiner, A. Grove, and D. Roth. Learning cost-sensitive active classifiers. *Artificial Intelligence*, 139, pages 137–174, 2002.

[49] Q. Gu and J. Han. Towards active learning on graphs: An error bound minimization approach. *IEEE International Conference on Data Mining*, pages 882–887, 2012.

[50] Q. Gu, C. Aggarwal, J. Liu, and J. Han. Selective sampling on graphs for classification. *ACM KDD Conference*, pages 131–139, 2013.

[51] A. Guillory and J. Bilmes. Active semi-supervised learning using submodular functions. *UAI*, 274–282, 2011.

[52] A. Guillory and J. A. Bilmes. Label selection on graphs. *NIPS*, pages 691–699, 2009.

[53] Y. Guo and R. Greiner. Optimistic active learning using mutual information. *International Joint Conference on Artificial Intelligence*, pages 823–829, 2007.

[54] S. Hanneke. A bound on the label complexity of agnostic active learning. In *ICML*, pages 353–360, 2007.

[55] S. C. H. Hoi, R. Jin, and M.R. Lyu. Large-scale text categorization by batch mode active learning. *World Wide Web Conference*, pages 633–642, 2006.

[56] S. C. H. Hoi and R. Jin. Active kernel learning. *ICML*, 2008.

[57] S. C. H. Hoi, R. Jin, J. Zhu, and M.R. Lyu. Batch mode active learning and its application to medical image classification. *International Conference on Machine Learning (ICML)*, pages 417–424, 2006.

[58] S. C. H. Hoi, R. Jin, J. Zhu, and M.R. Lyu. Semi-supervised SVM batch mode active learning for image retrieval. *IEEE Conference on CVPR*, pages 1–3, 2008.

[59] S. Huang, R. Jin, and Z. Zhou. Active learning by querying informative and representative examples. *NIPS Conference*, pages 892–200, 2011.

[60] R. Hwa. Sample selection for statistical parsing. *Computational Linguistics*, 30:73–77, 2004.

[61] M. Ji and J. Han. A variance minimization criterion to active learning on graphs. *AISTATS*, pages 556–554, 2012.

[62] S. Ji and L. Carin. Cost-sensitive feature acquisition and classification. *Pattern Recognition*, 40:1474–1485, 2007.

[63] N. Jin, C. Young, and W. Wang. GAIA: graph classification using evolutionary computation. In *SIGMOD*, pages 879–890, 2010.

[64] A. Joshi, F. Porikli, and N. Papanikolopoulos. Multi-class active learning for image classification In *CVPR Conference*, 2009.

[65] K. Judah, A. Fern, and T. Dietterich. Active imitation learning via reduction to I.I.D. active learning. In *UAI Conference*, 2012.

[66] A. Kapoor, E. Horvitz, and S. Basu. Selective supervision: Guiding supervised learning with decision theoretic active learning. *Proceedings of International Joint Conference on Artificial Intelligence (IJCAI)*, pages 877–882, 2007.

[67] R.D. King, K.E. Whelan, F.M. Jones, P.G. Reiser, C.H. Bryant, S.H. Muggleton, D.B. Kell, and S.G. Oliver. Functional genomic hypothesis generation and experimentation by a robot scientist. *Nature*, 427 (6971):247–252, 2004.

[68] C. Korner and S. Wrobel. Multi-class ensemble-based active learning. *European Conference on Machine Learning*, pp. 687–694, pages 687–694, 2006.

[69] X. Kong and P. S. Yu. Semi-supervised feature selection for graph classification. In *KDD*, pages 793–802, 2010.

[70] X. Kong, W. Fan, and P. Yu. Dual Active feature and sample selection for graph classification. *ACM KDD Conference*, pages 654–662, 2011.

[71] V. Krishnamurthy. Algorithms for optimal scheduling and management of hidden Markov model sensors. *IEEE Transactions on Signal Processing*, 50(6):1382–1397, 2002.

[72] B. Krishnapuram, D. Williams, Y. Xue, L. Carin, M. Figueiredo, and A. Hartemink. Active learning of features and labels. *ICML Conference*, 2005.

[73] A. Kuwadekar and J. Neville, Relational active learning for joint collective classification models. *ICML Conference*, pages 385–392, 2011.

[74] D. Lewis and W. Gale. A sequential algorithm for training text classifiers. *ACM SIGIR Conference*, pages 3–12, 1994.

[75] D. Lewis and J. Catlett. Heterogeneous uncertainty sampling for supervised learning. *ICML Conference*, pages 148–156, 1994.

[76] X. Li and Y. Guo. Active learning with multi-label SVM classification. *IJCAI Conference*, pages 14–25, 2013.

[77] X. Li, D. Kuang, and C.X. Ling. Active learning for hierarchical text classication. *PAKDD Conference*, pages 14–25, 2012.

[78] X. Li, D. Kuang and C.X. Ling. Multilabel SVM active learning for image classification. *ICIP Conference*, pages 2207–2210, 2004.

[79] C. X. Ling, Q. Yang, J. Wang, and S. Zhang. Decision trees with minimal costs. *ICML Conference*, pages 483–486, 2004.

[80] H. Liu, H. Motoda, and L. Yu. A selective sampling approach to active feature selection. *Artificial Intelligence*, 159(1-2):49–74, 2004

[81] H. Liu, H. Motoda, and L. Yu. Feature selection with selective sampling. *ICML Conference*, pages 395–402, 2002

[82] H. Liu, L. Yu, M. Dash, and H. Motoda. Active feature selection using classes. *PAKDD conference*, page 474–485, 2003

[83] R. Lomasky, C. Brodley, M. Aernecke, D. Walt, and M. Friedl. Active class selection. *Proceedings of the European Conference on Machine Learning*, pages 640–647, 2007.

[84] A. McCallum and K. Nigam. Employing EM in pool-based active learning for text classification. *ICML Conference*, pages 359–367, 1998.

[85] D. MacKay. Information-based objective functions for active data selection. *Neural Computation*, 4(4):590–604, 1992.

[86] D. Margineantu. Active cost-sensitive learning. *International Joint Conference on Artificial Intelligence (IJCAI)*, pages 1622–1623, 2005.

[87] P. Melville, M. Saar-Tsechansky, F. Provost, and R. Mooney. An expected utility approach to active feature-value acquisition. *IEEE ICDM Conference*, pages 2005.

[88] P. Melville, M. Saar-Tsechansky, F. Provost, and R. Mooney. Active feature-value acquisition for classifier induction. *IEEE International Conference on Data Mining (ICDM)*, pages 483–486, 2004.

[89] P. Melville and R. Mooney. Diverse ensembles for active learning. *ICML Conference*, pages 584–591, 2004.

[90] I. Muslea, S. Minton, and C. Knoblock. Active + semi-supervised learning = robust multi-view learning. *ICML Conference*, pages 435-442, 2002.

[91] I. Muslea, S. Minton, and C. Knoblock. Active learning with multiple views. *Journal of Artificial Intelligence Research*, 27(1):203–233, 2006.

[92] I. Muslea, S. Minton, and C. Knoblock. Selective sampling with redundant views. *AAAI Conference*, pages 621–626, 2000.

[93] R. Moskovitch, N. Nissim, D. Stopel, C. Feher, R. Englert, and Y. Elovici. Improving the detection of unknown computer worms activity using active learning. *Proceedings of the German Conference on AI*, pages 489–493, 2007.

[94] K. Nigam, A. McCallum, S. Thrun, and T. Mitchell. Text classification from labeled and unlabeled documents using EM. *Machine Learning*, 39(2–3):103–134, 2000.

[95] A. McCallum and K. Nigam. Employing EM pool-based active learning for text classification. *ICML Conference*, pages 350–358, 1998.

[96] H. T. Nguyen and A. Smeulders. Active learning using pre-clustering. *ICML Conference*, pages 79–86, 2004.

[97] S. J. Pan, Q. Yang. A survey on transfer learning. *IEEE Transactons on Knowledge and Data Engineering*, 22(10):1345–1359, 2010.

[98] G. Qi, X. Hua, Y. Rui, J. Tang, and H. Zhang. Two-dimensional active learning for image classification. *CVPR Conference*, 2008.

[99] R. Reichart, K. Tomanek, U. Hahn, and A. Rappoport. Multi-task active learning for linguistic annotations. *ACL Conference*, 2008.

[100] N. Roy and A. McCallum. Toward optimal active learning through sampling estimation of error reduction. *ICML Conference*, pages 441–448, 2001.

[101] M. Saar-Tsechansky, P. Melville, and F. Provost. Active feature-value acquisition. *Management Science*, 55(4):664–684, 2009.

[102] T. Scheffer, C. Decomain, and S. Wrobel. Active hidden Markov models for information extraction. *International Conference on Advances in Intelligent Data Analysis (CAIDA)*, pages 309–318, 2001.

[103] A. I. Schein and L.H. Ungar. Active learning for logistic regression: An evaluation. *Machine Learning*, 68(3):253–265, 2007.

[104] G. Schohn and D. Cohn. Less is more: Active learning with support vector machines. *ICML Conference*, pages 839–846, 2000.

[105] B. Settles. Active learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 6(1):1–114, 2012.

[106] B. Settles and M. Craven. An analysis of active learning strategies for sequence labeling tasks. *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1069–1078, 2008.

[107] B. Settles, M. Craven, and S. Ray. Multiple-instance active learning. *Advances in Neural Information Processing Systems (NIPS)*, pages 1289–1296, 2008.

[108] B. Settles, M. Craven, and L. Friedland. Active learning with real annotation costs. *NIPS Workshop on Cost-Sensitive Learning*, 2008.

[109] H. S. Seung, M. Opper, and H. Sompolinsky. Query by committee. *ACM Workshop on Computational Learning Theory*, pages 287–294, 1992.

[110] V. S. Sheng and C. X. Ling. Feature value acquisition in testing: A sequential batch test algorithm. *ICML Conference*, pages 809–816, 2006.

[111] V. S. Sheng, F. Provost, and P.G. Ipeirotis. Get another label? Improving data quality and data mining using multiple, noisy labelers. *ACM KDD Conference*, pages 614–622, 2008

[112] M. Singh, E. Curran, and P. Cunningham. Active learning for multi-label image annotation. Technical report, University College Dublin, 2009

[113] T. Soukop and I. Davidson. *Visual Data Mining: Techniques and Tools for Data Visualization*, Wiley, 2002.

[114] M. Thoma, H. Cheng, A. Gretton, J. Han, H. Kriegel, A. Smola, L. Song, P. Yu, X. Yan, and K. Borgwardt. Near-optimal supervised feature selection among frequent subgraphs. In *SDM*, pages 1075–1086, 2009.

[115] M. Wang and X. S. Hua. Active learning in multimedia annotation and retrieval: A survey. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 2(2):10, 2011.

[116] W. Wang and Z. Zhou. On multi-view active learning and the combination with semi-supervised learning. *ICML Conference*, pages 1152–1159, 2008.

[117] W. Wang and Z. Zhou. Multi-view active learning in the non-realizable case. *NIPS Conference*, pages 2388–2396, 2010.

[118] X. Shi, W. Fan, and J. Ren. Actively transfer domain knowledge. *ECML Conference*, pages 342–357, 2008

[119] Z. Xu, R. Akella, and Y. Zhang. Incorporating diversity and density in active learning for relevance feedback. *European Conference on IR Research (ECIR)*, pages 246–257, 2007.

[120] Z. Xu, K. Yu, V. Tresp, X. Xu, and J. Wang. Representative sampling for text classification using support vector machines. *ECIR Conference*, pages 393–407, 2003.

[121] X. Yan, H. Cheng, J. Han, and P. Yu. Mining significant graph patterns by leap search. In *SIGMOD*, pages 433–444, 2008.

[122] Y. Yan, R. Rosales, G. Fung, and J. Dy. Active learning from crowds. In *ICML*, 2011.

[123] X. Yan, P. S. Yu, and J. Han. Graph indexing based on discriminative frequent struture analysis. *ACM Transactions on Database Systems*, 30(4):960–993, 2005.

[124] B. Yang, J. Sun, T. Wang, and Z. Chen. Effective multi-label active learning for text classication. *ACM KDD Conference*, pages 917–926, 2009.

[125] L. Yang, S. Hanneke, and J. Carbonell A theory of transfer learning with applications to active learning. *Machine Learning*, 90(2):161–189, 2013.

[126] K. Yu, J. Bi, and V. Tresp. Active learning via transductive experimental design. *ICML Conference*, pages 1081–1088, 2006.

[127] H. Yu. SVM selective sampling for ranking with application to data retrieval. *ACM KDD Conference*, pages 354–363, 2005.

[128] Y. Zhang. Multi-task active learning with output constraints. *AAAI*, 2010

[129] T. Zhang and F.J. Oles. A probability analysis on the value of unlabeled data for classification problems. *ICML Conference*, pages 1191–1198, 2000.

[130] Q. Zhang and S. Sun. Multiple-view multiple-learner active learning. *Pattern Recognition*, 43(9):3113–3119, 2010.

[131] L. Zhao, S. Pan, E. Xiang, E. Zhong, Z. Lu, and Q. Yang Active transfer learning for cross-system recommendation. *AAAI Conference*, 2013

[132] Z. Zheng and B. Padmanabhan. On active learning for data acquisition. *IEEE International Conference on Data Mining*, pages 562–569, 2002.

[133] D. Zhou, O. Bousquet, T. N. Lal, J. Weston, and B. Scholkopf. Learning with local and global consistency. *NIPS Conference*, pages 321–328, 2003.

[134] Y. Zhu, S. J. Pan, Y. Chen, G.-R. Xue, Q. Yang, and Y. Yu. Heterogeneous transfer learning for image classification. *Special Track on AI and the Web, associated with The Twenty-Fourth AAAI Conference on Artificial Intelligence*, 2010.

[135] X. Zhu, J. Lafferty, and Z. Ghahramani. Combining active learning and semisupervised learning using Gaussian fields and harmonic functions. *Proceedings of the ICML Workshop on the Continuum from Labeled to Unlabeled Data*, pages 58–65, 2003.

[136] X. Zhu, P. Zhang, X. Lin, and Y. Shi. Active learning from data streams. *ICDM Conference*, pages 757–762, 2007.

[137] Z. Zhu, X. Zhu, Y. Ye, Y. Guo, and X. Xue. Transfer active learning. *CIKM Conference*, pages 2169–2172, 2011.

[138] M. Zuluaga, G. Sergent, A. Krause, M. Puschel, and Y. Shi. Active Learning for Multi-Objective Optimization. *JMLR*, 28(1):462–470, 2013.