Active Learning with Explore and Exploit Equilibriums

by

Ghazal Shams

A Dissertation Presented in Partial Fulfillment
of the Requirement for the Degree
Doctor of Philosophy

Approved June 2020 by the
Graduate Supervisory Committee:

George C. Runger, Chair
Douglas C. Montgomery
Adolfo R. Escobedo
Giulia Pedrielli

ARIZONA STATE UNIVERSITY

August 2020

ABSTRACT

In conventional supervised learning tasks, information retrieval from extensive collections of data happens automatically at low cost, whereas in many real-world problems obtaining labeled data can be hard, time-consuming, and expensive. Consider healthcare systems, for example, where unlabeled medical images are abundant while labeling requires a considerable amount of knowledge from experienced physicians. Active learning addresses this challenge with an iterative process to select instances from the unlabeled data to annotate and improve the supervised learner. At each step, the query of examples to be labeled can be considered as a dilemma between exploitation of the supervised learner's current knowledge and exploration of the unlabeled input features.

Motivated by the need for efficient active learning strategies, this dissertation proposes new algorithms for batch-mode, pool-based active learning. The research considers the following questions: how can unsupervised knowledge of the input features (exploration) improve learning when incorporated with supervised learning (exploitation)? How to characterize exploration in active learning when data is high-dimensional? Finally, how to adaptively make a balance between exploration and exploitation?

The first contribution proposes a new active learning algorithm, Cluster-based Stochastic Query-by-Forest (CSQBF), which provides a batch-mode strategy that accelerates learning with added value from exploration and improved exploitation scores. CSQBF balances exploration and exploitation using a probabilistic scoring criterion based on classification probabilities from a tree-based ensemble model within each data cluster.

The second contribution introduces two more query strategies, Double Margin Active Learning (DMAL) and Cluster Agnostic Active Learning (CAAL), that combine

consistent exploration and exploitation modules into a coherent and unified measure for label query. Instead of assuming a fixed clustering structure, CAAL and DMAL adopt a soft-clustering strategy which provides a new approach to formalize exploration in active learning.

The third contribution addresses the challenge of dynamically making a balance between exploration and exploitation criteria throughout the active learning process. Two adaptive algorithms are proposed based on feedback-driven bandit optimization frameworks that elegantly handle this issue by learning the relationship between exploration-exploitation trade-off and an active learner's performance.

To Maman and Baba, who always encouraged me to go on every advanture, especially this one.

ACKNOWLEDGEMENTS

This dissertation would not exist without the help and guidance of many people, and I am happy to take this opportunity to appreciate those who have influenced me during my graduate career.

Professionally, I am most indebted to my advisor, Dr. George Runger. He not only taught me a great deal on data science and machine learning, but how to look at real problems from an insatiably curious and scientific point of view. Most importantly, he taught me how to be a great mentor by providing the freedom to pursue my own research interests at the same time that he was there to reign me in. Above all, he continuously showed me how to be a great person.

I would also love to thank my Ph.D. committee members for their guidance and feedback provided in this work. Douglas Montgomery, Adolfo Escobedo, and Giulia Pedrielli have introduced me to a variety of ways of thinking about the applications of my work in real-world.

I gratefully acknowledge the financial support I received towards my Ph.D. project from the National Science Foundation (grant 1537898).

Special thanks to Maria Jose Suazo Ocares, who made a great flat-mate during our time at ASU, sharing fabulous teas and many memorable moments. You taught me how to grow an attitude of gratitude towards life, and I'm forever thankful for your unconditional support when I needed it the most.

I must thank all my friends, especially Xiushuang Li, Sangdi Lin, Mona Khoddam, Nooshin Shomal Zadeh, Maziar Kasaei, Hyunsoo Yoon, Nathan Gaw, Kun Wang, Viswanath Potluri, and Shaohao Huang. I found a lot of love and support from you guys, and this journey couldn't be possible without you.

I am most personally indebted to my boyfriend, Benyamin. You taught me how to keep my feet on the ground, how to sacrifice, how to love, and I cannot wait to see

what life has for us next.

It is needless to say how my family has always been supportive of my life choices. Graduate school and my migration have been difficult for them (and of course for me), but it is because of them that I even did it at all. Dr. Runger was my wings, but they were my heart to fly this far...

TABLE OF CONTENTS

CHAPTER Page

CHAPTER Page

LIST OF TABLES

LIST OF FIGURES

Figure Page

Chapter1

INTRODUCTION

1.1 What is Active Learning?

Since the dawn of computing, businesses have struggled with the massive amount of daily data streams. Nowadays, it is a well-understood fact that the ability to analyze data faster and more efficiently brings them a competitive edge. Other than advances in data infrastructures that have facilitated data collection and storage, computers have played a significant role in conducting analysis and providing decision support. Data mining is a generic term that includes all the advanced techniques and processes that are used to discover the underlying credible patterns of data for abundant application domains such as fraud and intrusion detection in financial and network analysis, gene sequencing in bioinformatics, product lifetime prediction in manufacturing, content recommendation in platforms such as Netflix and Amazon. Depending on the problem domain, goals, and constraints, different data mining techniques can be used to learn valuable insights about the data.

However, most of the current methodologies need a considerable amount of annotated (labeled) data to learn the relationship between a set of system characteristics and events. In other words, statistical learning happens under supervision and is evaluated based on the ground truth (label) provided. Note that in this process,

the learning model is only a recipient of the data, and it does not participate in data annotation, i.e., it is passive. In most of the machine learning literature, supervised learning refers to passive supervised learning. Figure 1.1 illustrates the general scheme of this type of learning. In contrast to supervised learning, there is the process of unsupervised learning, where the underlying structure of data is more of a question or no label for the data is available. Understanding the association between shopping basket items or simply grouping products, people, or species based on their characteristics are examples of unsupervised learning.

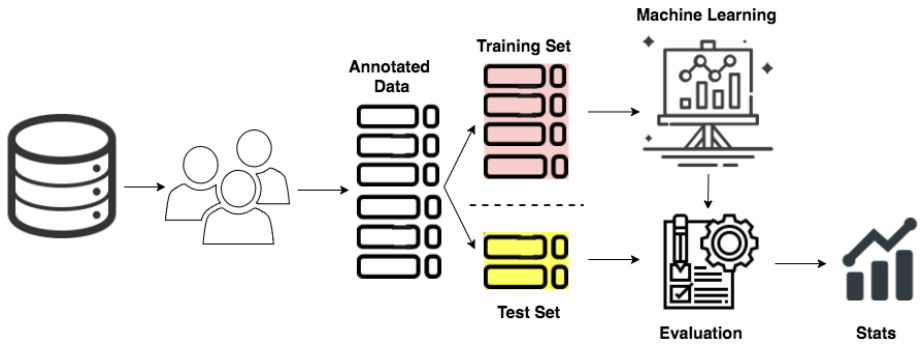


Figure 1.1: General Scheme of Supervised Learning

Learning in supervised models is contingent on the existence of a significant amount of annotated data. Automated systems might be able to provide these annotations at little or no cost. For example, the emotional flags that social media users give to a content is a cheap labeling process for that platform. However, there exist many cases where obtaining labeled data is of hard endeavor, time-consuming, or cost-extensive. For instance, in healthcare, an experienced physician has to go through several health records from medical images to lab tests to be able to make a diagnosis. Another example can be in speech recognition, where labeling speech utterances requires trained linguists and can take a significant amount of time [160].

Active learning (AL) is the study of machine learning algorithms that can query information. They are a form of semi-supervised learning that comes in useful where

labeling data is an expensive and time-intensive task. The key hypothesis in AL is that if allowed to choose the data from which it learns, a statistical learner can perform better with less training. In other words, despite the abundance of data, not all instances have the same value to the learning model. Therefore, if there is any effort or cost associated with data annotation, it has to be optimized so that the desired performance can be achieved with relatively less data. AL aims to provide a framework to selectively choose the potentially most valuable observations to benefit a statistical learner.

AL ultimately aims to learn a mapping function $\Phi : X \mapsto Y$ where $X \in \mathbb{R}^{n \times m}$ represents the m-dimensional features set, and $Y \in \mathbb{R}^{n \times 1}$ is their corresponding output. While input features are abundant or easy to obtain, the active learner needs to interact with an expert (oracle) to query output feature Y. To do so, AL usually proceeds in rounds. Initially the learner is presented with an empty labeled set, $\mathcal{L}^0$, while unlabeled dataset $\mathcal{U}^0$ is readily available. Data labels are provided by an oracle in a sequential manner. At each time step, the learner trains a classifier $\Phi^{(t)}$ using the available labeled data $\mathcal{L}^t$ and selects a batch of instances to query based on their expected benefit to the classifier. Each time the training set is augmented with new labeled instances, the active learner re-trains and evaluates more unlabeled data for the proceeding rounds. This process stops when the learner runs out of the query budget, or it achieves a desired performance. Given the limited query budget T, the goal is to fit the best classifier. Figure 1.2 demonstrates the general routine of AL.

Depending on the application domain, different label query scenarios can be considered for AL. A prevalent AL scenario is *Pool-based*, where an abundant amount of unlabeled data is available, while the labeled data is limited or not available at all. In this scenario, a learner may start with a few labeled instances and iteratively query labels based on the leveraged knowledge from the previous iteration. Note that

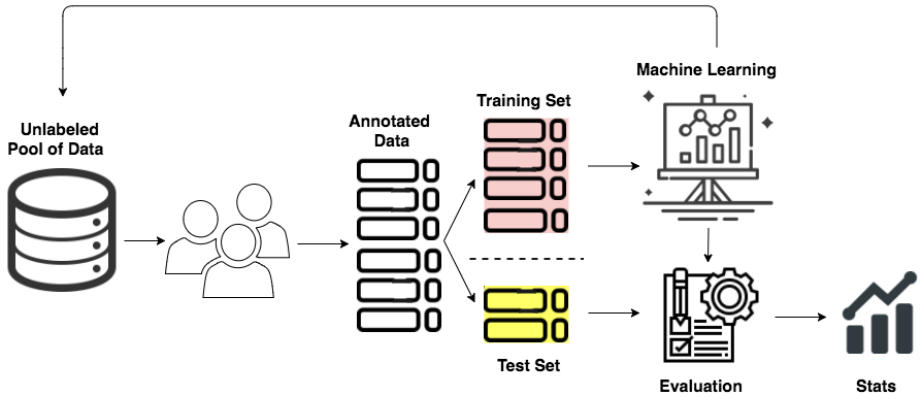


Figure 1.2: General Scheme of Active Learning

in this case, the learner can request labels for any of the observations in the pool. An example of pool-based AL can be the classification of those tweets that have a cyber-bullying content, among millions of tweets that are posted everyday. In this case, human annotators are needed to flag the tweets according to some predefined labels (bully, aggressive, neutral, etc.).

Another common scenario is the *Stream-based* AL (also known as *Sequential* or *Selective* sampling), where at each round, the learner has to decide whether to label an instance or discard it. Unlike the pool-based scenario, in sequential AL instances would no longer be available if discarded [122]. For instance, consider the example of internet advertisements popping up on customers' screens. Whether a customer shows interest in the content of this pop-up by clicking on it, is associated with a cost (we may lose the customer if there are too many pop-ups or they are unrelated). The active learner only has one chance of whether to show an advert when a page is opened. If it decides not to, the opportunity is no longer available. The goal is to achieve maximum click-through rates by learning customer's behavior.

Query Synthesis is another query scheme, where queries can be made for any data point in the feature space, even if those points have not been observed. For example, in [75], possible combinations of proteins were synthesized and labeled based on the

presence of a target feature. Although this scenario may fit some experimental design applications, it is not a well-defined approach for typical classification problems where observations cannot be created de novo. Experimental design is a related field to AL, where credence are given for choosing a set of data instances to fulfill a specific set of assumptions and objectives. One major difference between the majority of AL applications and those of experimental design is the type of response variable which indicates whether the problem is a classification or a regression one. We further discuss the similarities and differences between these two research areas later.

Regardless of the AL scenario, the decision of whether to query an instance or not is formulated in various ways. *Informativeness* is the general term used in the literature referring to mechanisms that evaluate instances for label queries from the supervised model point of view. The main idea behind these mechanisms is that the decision boundary lies in the most ambiguous regions of the feature space in terms of classification. Hence, instances from these regions are more likely to be valuable to the learning engine. Several methodologies to evaluate the informativeness of instances have been proposed and successfully applied to various AL problems, as we discuss later. Nevertheless, queries by these approaches tend to be acutely biased towards those instances that do not represent the true feature space [29]. To enhance the informativeness-based AL approaches, *representativeness* concept was introduced to the query strategies where the active learner tries to exploit the input data distribution [122] and as a result make the labeled set a better representative of the input data.

In AL terminology, *uncertainty* and *density*, as well as *exploitation* and *exploration*, are two groups of terms used interchangeably with informativeness and representativeness, respectively.

AL methodologies are usually evaluated based on how much they speed up the learning process.

1.2 Thesis Statement and Organization

This thesis proposes new methodologies for pool-based AL. We address the challenge of sampling bias introduced by query strategies merely based on instance informativeness where the queries might not reflect the input data distribution. The research questions that this thesis addresses are: how can AL improve learning with unsupervised knowledge of the data? What if this unsupervised knowledge itself is subject to uncertainty, i.e. how to characterize our understanding from data structure into the query process? Finally, how to make a balance between informativeness and representativeness adaptively throughout the course of AL?

Our introduced algorithms in the following chapters are inspired by the winner of the 2010 AISTATS active learning challenge, the Stochastic Query-by-Forest (SQBF) algorithm [8]. To advance the AL field, the Causality Active Learning Challenge was conducted in 2010 where participants tried to solve six binary classification problems from different domains such as chemo-informatics, handwriting recognition, text processing, etc. where labeling can be a challenging task [53]. SQBF introduced a sampling procedure for which is robust to the type as well as the size of data, and is efficient for batch-mode AL. Throughout the following chapters, we show how the proposed query strategies significantly improve upon SQBF by offering several new features. The key aspects of the proposed algorithms in this thesis other than an enhanced performance are i) extendability to include various AL modules other than uncertainty and density (such as diversity) ii) applicability to various domains due to robustness of the utilized supervised and unsupervised models iii) scalability to large datasets of different sizes and types (as we show through our experiments). Furthermore, the proposed algorithms demonstrate how unsupervised knowledge of the data and its reliability influences the learning process when aggregated (statically

and dynamically) with model supervised uncertainty. The rest of this dissertation is organized as follows:

Chapter 2 provides a detailed background on AL query strategies as well as their advantages and disadvantages, the informativeness and representativeness methodologies that are utilized in this work, and the intersection of AL with other similar fields such as design of experiments and Bayesian optimization.

Chapter 3 introduces the first proposed algorithm designed to utilize a probabilistic instance scoring criterion based on instance informativeness within each data cluster. The developed methodology, which we call Cluster-based Stochastic Query by Forest (CSQBF), provides a batch-mode strategy that accelerates learning from the early stages by focusing on informative areas of the feature space that are also better representatives of the data. CSQBF benefits from a stochastic sampling process that not only is computationally efficient, but also it diversifies the instances in each query batch.

Chapter 4 proposes two more new algorithms that bring informativeness and representativeness criteria into a coherent and unified query strategy for AL. These algorithms, namely the Double Margin Active Learning (DMAL) and the Cluster Agnostic Active Learning (CAAL), consider the ambiguity associated with our understanding of the high-dimensional feature space and incorporate that into the query process. While CSQBF constructs distinct clusters with the assumption that instances within a cluster are more likely to share labels, CAAL and DMAL allow for a soft clustering structure. As a result, instead of searching data clusters for more informative instances, representativeness accepts a new form which is easily combined with informativeness to create a consistent utility measure for label querying.

Chapter 5 addresses the challenge of dynamically making a balance between informativeness and representativeness throughout the AL process. The algorithms

proposed in Chapters 3–4 define new strategies to formulate informativeness and representativeness. However, they do not combine the two criteria strategically based on the state of the learning process. Inspired by the bandit optimization problems, where decisions are made over time under uncertainty, two strategies are proposed that learn how to make a trade-off between informativeness and representativeness in any AL framework (including those introduced previously). Moreover, they can easily extend to cases when there are more factors other than informativeness and representativeness involved in the query process.

In order to show the robustness of our proposed methodologies, in each chapter, we present multiple sets of experiments and statistical tests using several real-world datasets with different characteristics from various domains.

Chapter2

RELATED WORK

2.1 Theories Behind Queries

The main purpose of AL is to reduce the cost of data annotation (labeling) by choosing instances that benefit the classifier the most. However, how these instances are selected creates several research branches. Work by [122] reviews different AL scenarios, query strategy frameworks, and theoretical as well as empirical analysis of AL problems. The three main AL scenarios, as described in this review, are:

- ***Membership Query Synthesis*** in which the statistical learner can request label for any data point that lies in the feature space. It has been argued that these approaches are more suitable for problems with non-human annotators.

- ***Stream-based Sampling*** approaches are used when a stream of unlabeled instances are presented to the learner, and the learner has to decide whether to query label for each instance or not.

- ***Pool-based Sampling*** is the most common scenario where a learner has to choose instances from an existing pool of unlabeled data and query labels for them.

Although there have been several studies on query strategies for each of these sce-

narios, we dedicate our discussion to the methods developed on pool-based AL for applicability in this dissertation. Also, [123] provides further resources on different AL scenarios.

We described that AL intends to learn a mapping function $\Phi : X \mapsto Y$ between input features X and the response feature (label) Y. Features X are readily available at little or no cost, but the active learner has to query labels Y from an oracle. **The main challenge for the learner is how to score instances in X to find the most useful ones and query their labels from the oracle?**

There is a large body of work on AL query strategies that define the usefulness of an instance from a prediction perspective. These strategies mainly offer two perspectives to seek highly informative instances. The first one is concerned with reducing classifier's uncertainty in assigning labels. The central idea is that if the classifier is not certain how to label a data point, that point is a good candidate to be labeled by the oracle. A vast amount of theoretical works on AL has focused on formulating this notion [59, 122] and we elaborate on it shortly.

The second perspective deals with the distribution of the data which is less explored as far as theory goes. Ideally, data is clustered into several groups that are homogeneous in the label, and one can sample a few instances from each group to train the best classifier. Although this might seem optimistic in general, some AL strategies aim to harness knowledge from clusters (loosely) aligned with class labels [27].

From a decision-theoretic perspective, approaching AL from either classification uncertainty reduction or incorporation of the data distribution point of views, is similar to the exploration-exploitation trade-off in reinforcement learning [121]. The reinforcement learner either has to follow actions that have worked well in the past, or it can try out new actions hoping for a better outcome. [73, 139]. Similarly an

active either learner queries labels for instances that the current classifier is uncertain how to label, or it samples from different areas of the feature space [121]. Throughout this dissertation we aim to make a balance between exploration and exploitation in context of AL which translates to querying labels for instances that reduce model uncertainty in classification while making the queries less biased toward only some parts of the data distribution. We also elaborate more on exploration-exploitation later in Chapter 5 in the context dynamic learning.

2.1.1 Uncertainty Sampling

One of the most prevalent AL query strategies is uncertainty sampling introduced by [81]. During AL, revealing an expected label through a query does not have much value for the current model, as the model has foreseen the label with high certainty. In uncertainty-based sampling, queried instances are those that cannot be easily classified by the current model.

One way to formulate uncertainty is through probabilistic models where class membership probabilities obtained from a model can be interpreted as model confidence in prediction [81, 122]. Variants of this strategy have been proposed which basically differ in the probability that they consider to query the labels and can be as simple as selecting instance $x^*_{LC} \in X$ such that

$$x^*_{LC} = \underset{x \in X}{argmax}\ 1 - P(\hat{y}|x, \Phi) \tag{2.1}$$

where Φ is the probability model of choice, $P(\hat{y}|x, \Phi)$ is the probability of the most probable output class for x under model Φ. However, this strategy only looks at the maximum class membership probability and does not take into account the whole distribution and can be misleading for the active learner.

Margin sampling improves upon the most uncertain measure by incorporating the

probability of the second most likely class as

$$x^*_M = \underset{x}{argmin}[\ \underset{y}{Max}\{P(\hat{y}|x,\Phi)\} - \underset{y}{SecMax}\{P(\hat{y}|x,\Phi)\}\ ] \tag{2.2}$$

Intuitively, if the probability of the most likely class is not much greater than the probability of the second most likely class, then the classifier is uncertain about the label, making that instance a relatively better candidate for label query. An illustrative example of this case is presented in Chapter 4.

Entropy [127] is the third member of the uncertainty-based strategies that is widely used in AL literature and is appropriate when the objective is to minimize the log-loss.

$$x^*_H = \underset{x}{argmax}\ -\sum_i P(y_i|x,\Phi)\log P(y_i|x,\Phi) \tag{2.3}$$

where index i indicates the class category.

Work by [112] conducted a comprehensive study on uncertainty measures and concluded that margin sampling performs best on average. Therefore, throughout this dissertation we formulate classification uncertainty using margin sampling.

Another approach to uncertainty sampling is to use the following concept. The classifier boundary should lie somewhere in the high-dimensional feature space X that is far from the easy-to-classify data points. In other words, closer points to this boundary are more ambiguous for classification. Therefore, labeling them can better clarify the position of the decision boundary. This is the main idea behind many AL methodologies that use Support Vector Machines (SVM) as their classifier as it creates a hyperplane boundary to classify data points which makes it easier to define distance [25, 25, 154]. Work by [140] formalized this idea and inferred that the closest data instance to the decision boundary can decrease model uncertainty the fastest. Figure 2.1 is an illustrative example of this concept.

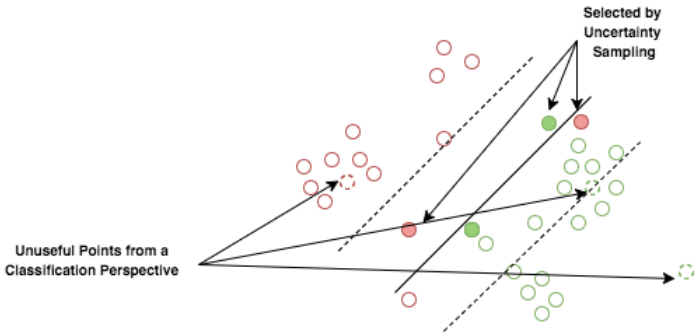


Figure 2.1: An Illustrative Example of Uncertainty Sampling Based on the Distance to the Decision Boundary in a Binary Classification Problem. Each Color Represents One Class. The Filled Points Are Query Candidates by Uncertainty Sampling. Image Modified From [154]. Best View in the Color Version.

2.1.2 Expected Error and Variance Reduction

The goal in expected error and variance reduction strategies for AL is to reduce the future classification error [123]. Because the actual labels are not available until after the query is made, the expected value of classification error over all labels is calculated instead. Let $\mathcal{L}^t$ and $\mathcal{U}^t$ be the set of labeled and unlabeled instances available to model Φ at time t respectively. An expected error reduction strategy queries the next data point according to

$$\begin{aligned} x^*_{EER} &= \underset{x}{argmin}\, E_{y|\Phi,x}\left[\sum_{x'\in\mathcal{U}^t} E_{y|\Phi^+,x'}[y \neq \hat{y}]\right] \\ &= \underset{x}{argmin} \sum_{y_i\in Y} P(y_i|x,\Phi)\left(\sum_{x'\in\mathcal{U}^t} 1 - P(\hat{y}|x,\Phi^+)\right) \end{aligned} \tag{2.4}$$

where Φ^+ is the model after it is trained using $\mathcal{L}\cup\{(x,y_i)\}$, and $E_{y|\Phi,x}$ is the expected outcome which we approximate using expectation over all possible class labels under the current model. Although variants of this approach have been successful in providing satisfactory decisions [115, 161], estimating the expected future error over a the entire pool of unlabeled data for each query makes expected error reduction strategies computationally intensive [29, 123]. Furthermore, these strategies are more suitable for labeling one instance at a time other than a batch-mode query.

It has been noted that the classifier expected future error is proportional to its variance [44, 143]. Therefore, instead of directly minimizing the expected generalization error (according to Eq. (2.4)), the variance of the classifier output can be minimized. Work by [122] discusses how the link between expected future error and variance for specific models. They argue that the main advantage of variance reduction strategies is that actual labels for query candidates are not needed to calculate the expected future variance. Therefore, these methods do not suffer from high computational costs as expected error reduction ones do. Moreover, unlike expected error reduction strategies, variance reduction approaches allow for batch-mode label queries because they are independent of the actual labels and their expected value [24, 90]. However, obtaining closed-form solutions for the variance of supervised classifiers is not easy and can become intractable for more complex models with a higher number of parameters [123].

2.1.3 Query-By-Committee

Proposed by [41], the Query-By-Committee (QBC) strategy involves building a committee of classifiers, each voting for the label of unlabeled instances. Depending on the level of label disagreement among the committee members, instances are selected for labeling.

Entropy and Kullback-Leibler (KL) divergence (measuring the difference between two probability distributions) [77, 78] are the two common information-theoretic measures used to characterize the degree of disagreement in QBC approaches [94, 121]. QBC strategy queries the unlabeled observation that maximizes the Entropy (or KL divergence) between the label distributions of any one committee member and the consensus [121].

2.1.4 Exploiting the Data Distribution

Uncertainty sampling methods can perform poorly, especially at early iterations, where many instances may appear to be informative since the model is not strong enough. Work by [161] and [115] illustrated how uncertainty and QBC based methods might fail to select the most useful instances and instead label outliers —instances that are in isolation and do not contain much information about the distribution of data —simply due to the controversy that they may create for the model. Instead, several strategies are proposed that measure if an instance well represents the overall input patterns of the unlabeled data [94, 100, 154].

Representativeness of a data sample is characterized in various ways. Among the strategies that explicitly formulate representativeness into the label query process, the majority define it based on a density score [94, 154, 158]. Density score of a sample can be evaluated based on how many samples there are similar or near to it. Depending on the application, this score can be defined in various ways. Distance metrics such as Cosine, Euclidean, or Gaussian kernel are examples of the standard measures to define distance between data points [74, 141]. A data point with high density score is less likely to be an outlier [59] as it is closer to its neighbors. Work by [124] argues that the best samples to query are those with the highest uncertainty and density scores, i.e.

$$x_D^* = \underset{x}{argmax}\, \Upsilon(x) \times \overline{\psi}(x, x_u) \tag{2.5}$$

where $\Upsilon(x)$ represents the uncertainty score based on any of the aforementioned methods and $\overline{\psi}(x, x_u)$ is the average similarity of the candidate instance x with all the unlabeled instances $\mathcal{U}^t$.

Although density scoring is a straightforward approach to account for representativeness of samples, other methodologies have been proposed in the AL literature

that implicitly take the data structure into account. For example, work by [161] proposed a graph-based strategy where instances are the graph nodes and edges are weighted based on the Euclidean distance between the corresponding instances. Their AL algorithm follows by propagating any obtained label to the unlabeled neighbor instances, and the unknown parts of the graphs are queried more. This is a well-structured scheme to leverage network data distribution into the query process. In another work, [28] developed a query strategy based on hierarchical clustering. They sequentially split clusters into more homogeneous ones as labeled instances become heterogeneous within each cluster.

Although successful performance of density-based and cluster-based approaches depend on whether the distance metric of choice is a proper one or whether there exists a cluster structure in the data associated with class labels, several studies have shown that they outperform approaches merely based on uncertainty sampling or QBC. Especially for cold-start AL, where no labeled instances are available initially [30, 100].

On top of uncertainty and density, another element that has been considered in AL query strategies is diversity [50, 74, 153, 157]. The main idea here is to prevent querying instances with high similarities to the ones that are already labeled, i.e.

$$x^*_{DD} = \underset{x}{argmax}\, \Upsilon(x) \times \overline{\psi}(x, x_u) \times \overline{\omega}(x, x_l) \tag{2.6}$$

where $\Upsilon(x)$ is uncertainty score, $\overline{\psi}(x, x_u)$ is the average similarity to all the unlabeled instances in $\mathcal{U}^t$, and $\overline{\omega}(x, x_l)$ is the average similarity to all the available labeled instances in $\mathcal{L}^t$. Similar to density, diversity can be characterized using different distance metrics.

2.2 Evaluation of Active Learning Algorithms

Evaluation for AL methodologies is done in various ways, but usually these evaluations involve some sort of learning curve, where classifier's performance is plotted against time. Performance is measured via metrics such as accuracy, Area Under the Receiver Operating Characteristic (ROC) curve (AUC), F1-score, etc. The ROC curve is created by plotting the true positive rate (recall) against the false positive rate at various threshold settings [12]. F1-score is the harmonic mean of precision and recall. Unlike accuracy that weighs all classes equally and so it favors the majority class, F1-score is focuses more on the minority class. Therefore, it can be a better performance metric.

Some studies such as [128] compare AL algorithms only via visual inspections of these curves. Work by [53, 74, 156], summarize the curves by using the area under them (Area Under the Learning Curve (ALC)). The ALC value indicates how fast the learning is happening.

Throughout this dissertation, we use area under the F1-score learning curve to evaluate AL strategies. However, following [53], we normalize the ALC values according the best possible learning curve as follows:

$$ALC_N = y_i = \frac{ALC - ALC_{rand}}{ALC_{MAX} - ALC_{rand}} \tag{2.7}$$

where ALC_{MAX} is the area under the best achievable learning curve and ALC_{rand} is the area under the average learning curve obtained by making random predictions [53]. The more ALC_N, the better is the performance of the AL algorithm.

2.3 Optimal Experimental Design and Bayesian Optimization

There are a few research areas that overlap with active learning. We elaborate on two of these areas, namely optimal experimental design and Bayesian optimization,

and their similarities as well as differences with AL.

2.3.1 Optimal Experimental Design

Experimental design has traditionally been used to optimally determine a set of N design points before performing any experiments. The main issue is to find approximately optimal designs for large N, as the exact problem is NP-hard [108]. In AL, the focus is moved from planning a whole set of experiments in one shot to actively planning the experiments one after another, while updating the learning algorithm after each query. Although AL and experimental design approaches are often different in their assumptions and goals, it is sometimes arbitrary whether to define a problem as an AL or experimental design. More specifically, there is an equivalence between expected error reduction AL frameworks and optimal experimental design which is elaborated on shortly.

To highlight the relationship between AL and experimental design, let's look at how experimental design works. Consider the problem of estimating $\beta \in \mathbb{R}^{m\times 1}$ from a set of measurements $(x_1, y_1), (x_2, y_2), \ldots (x_n, y_n)$ where each $x_i \in \mathbb{R}^{1\times m}$ is a vector of m predictor variables and $y_i \in \mathbb{R}$ is the scaler response corresponding to this measurement:

$$\Upsilon(x_i) = y_i = x_i\beta + \epsilon \ , \ \ i = 1, 2, \ldots, m \tag{2.8}$$

where ϵ is measurement error and $\epsilon \sim \mathcal{N}(0, 1)$. The maximum likelihood estimator of β which is unbiased and also has the minimum variance [99] is given by

$$\hat{\beta} = \left(\sum_{i=1}^{n} x_i x_i^T \right)^{-1} \sum_{i=1}^{n} y_i x_i \tag{2.9}$$

Now, suppose that observations $x_1, x_2, \ldots, x_n$ each can choose from $p \leq n$ possible types of candidate vectors $v_1, v_2, \ldots, v_p \in \mathbb{R}^{1\times m}$. In other words, we have a repeated measurements setup where each x_i corresponds to one experiment setup v_j.

Given model Υ in Eq. (2.8), experimental design aims to query observations that give the best estimate for β with minimum estimation variance [11]. In other words, we need to minimize the variance-covariance matrix of prediction errors (E) defined as following

$$E = \left(\sum_{i=1}^{n} x_i x_i^T \right)^{-1} = \left(\sum_{j=1}^{p} a_j v_j v_j^T \right)^{-1} \tag{2.10}$$

where scaler a_j is the numbers of each type of observations. Given that we only have n observations (which is equivalent to having a query budget of n in AL) and $\sum_{j=1}^{p} a_j = n$, the experimental design problem is converted to choosing a_j values, the numbers of occurrence of each v_j [11]. This problem is combinatorial, but is relaxed by ignoring that a_j values are integers [11].

Let $\lambda_j = a_j/n$ be the relative frequency of experiment j. Equation (2.10) is then converted to

$$E = \frac{1}{m} \left(\sum_{j=1}^{p} \lambda_j v_j v_j^T \right)^{-1} \tag{2.11}$$

by ignoring the fact that each λ_j is an integer multiple of $1/m$. This setup is equivalent to an AL framework where the goal is to learn the best model by minimizing the prediction error and we can only choose from a pool of observed instances.

Since E is a matrix, we cannot directly minimize it. Instead, several scalarizations have been proposed to solve the relaxed problem in Eq. (2.11) such as

- A-optimal (average) design which minimizes $trace\left(\lambda_j v_j v_j^T\right)^{-1}$
- D-optimal (determinant) design which minimizes $\log det\left(\lambda_j v_j v_j^T\right)^{-1}$
- E-optimal (extreme) design which minimizes the maximum eigenvalue of $\left(\lambda_j v_j v_j^T\right)^{-1}$

Note that optimal experimental design schemes illustrated above with the objective function in Eq. (2.10) are equivalent to variance reduction approaches in AL

when we have a regression problem at hand [122]. Therefore, they suffer from the same drawback as that of variance reduction methods. The key difference between optimal experimental design and AL is that the former relies on a model and because of this focuses only on the uncertainty type of measures to select instances while the latter can potentially combine uncertainty and density measures to select instances [107]. In other words, experimental design tends to choose extreme design points which automatically confirm the assumed model without exploiting any knowledge from the rest of the feature space [108].

Moreover, the optimal experimental designs above are suitable if model Υ is a linear model following the mathematical form of Eq. (2.8). Although, similar procedure can be carried out in case that $\Upsilon(x_i) = g(x, \beta)$ where g is a non-linear function of x by utilizing Taylor expansion around the current estimate of β [4] and there are several techniques such as sequential experimental design, Bayesian experimental design, and Maximin experimental design to estimate β in these cases [11], only a locally optimal design can be achieved. Furthermore, the relaxation of the combinatorial optimization model of choosing integer a_j values such that variance, E, is minimized, makes experimental design approaches more beneficial to problems where experiment v_j are not predefined, but rather can be synthesized. Therefore, these methods mostly fit the membership query synthesis AL scenario and not the pool-based one.

As illustrated above, one of the fundamental differences between the majority of AL algorithms and experimental design is that experimental design frameworks often offer theoretical criteria to select the full set of experiments in one shot using parametric models and under a specific set of assumptions which are often in a form of a prior belief over a continuous target [108]. Majority of AL schemes, on the other hand, are non-parametric heuristics that run sequentially where the learning model is updated in each round [148]. They avoid the computational burden of planning all

experiments by greedily planning only one step ahead. Unlike experimental design where we are allowed to select the data points in the feature space according to the design objective function, in most AL scenarios data instances are selected from the training data.

2.3.2 Bayesian Optimization

Another related area of research to AL is Bayesian optimization (BO), where the goal is to find the optimum of a complex black-box function by sequentially gathering information. The BO agent has a prior belief of this function which is updated each time after the evaluation of the function is made.

There is an extensive body of work on characterizing this belief, including improvement-based (e.g., probability of improvement (PI) or expected improvement (EI) over currently found maximum), entropy-based, and upper confidence bound (UCB) approaches [97]. The limitation of most BO algorithms is that they are either myopic meaning that they only consider one step ahead or their performances are not theoretically guaranteed [86]. The sequential and stochastic sampling procedure of BO is similar to that of AL. However, there are a few distinguishing points between the two. The main goal of AL is to learn a model (most often a classification one), whereas BO is only concerned with the optimum of a black-box function (usually in a regression framework). Similar to experimental design, BO is usually concerned with continuous feature domains, meaning any query instance can be synthesized, while the active learner can only choose from a set of training instances [15]. We will discuss BO more in Chapter 5.

2.4 Tree-based Ensembles

In statistics and machine learning, ensemble methods use several base learners in order to obtain better predictive performance compared to any of the constituent base learners alone [101]. It has been empirically and theoretically shown that the predictive performance of an ensemble model is positively correlated with the degree to which base learners' errors are uncorrelated [3, 116].

To reduce the correlation between the base learners, several strategies such as AdaBoost and Bagging can be adopted. *AdaBoost* [40] trains an ensemble of base models iteratively, and each time the misclassified instances have higher chances to be sampled while during *Bagging* [13], each base learner (decision tree) is trained only on a subset of data. This approach reduces model predictive variance by averaging the results from the base learners.

Random Forest (RF) [14] is an ensemble learning method for classification and regression, which aims at de-correlating the trees in bagging. It is known to be a flexible and easy to use machine learning technique that produces promising results even without much hyper-parameter tuning. Because RF is able to handle data with mixed features and even from different scales as well as categorical and numerical responses, RF a convenient alternative to the majority of current classifiers. As mentioned earlier, a RF model consists of several decision (regression) trees, each has a vote in the model final decision. Each tree tries to come up with a rectilinear decision boundary by splitting a bootstrap sample of instances into groups based on a decrease in an impurity measure such as Gini or Entropy. *Gini Impurity Index* [46] and *Entropy* [127] are both measures to assess the homogeneity of the target variable within the subset groups when a split happens. Intuitively, the Gini index measure how often a randomly chosen instance would be misclassified if it was randomly

labeled according to the label distribution of the subset group. Formally speaking, suppose we have a set of instances associated with $\mathbb{C}$, the set of class labels. Gini index is then defined as,

$$I_G = \sum_{c \in \mathbb{C}} p_c \sum_{c' \neq c} p'_c \tag{2.12}$$

where p_c are classification probabilities provided a classification model

Compared to bagging, Random Forest injects additional randomness to the model by searching for the best feature among a random subset of features instead of all features when splitting a node. This also reduces the correlation between trees as each tree only has access to a random subset of features [14]. The splits continue until a stopping criterion is met. In the end, The final decision on classification or prediction is made based on the majority of the votes or the average prediction made by the trees in the ensemble. The process makes RF intrinsically suitable for multi-class problems without any modifications.

RF model has several by-products, which are reviewed here shortly. As mentioned earlier, the root node of each tree contains a bootstrap sample of original data. Instances that are not in this sample are referred to as Out-Of-Bag (OOB) instances, which are roughly 1/3 of the data. OOB instances can be used to estimate the generalization error rate —measuring how accurately a model is able to predict outcome values for previously unseen data —of the RF model. For each OOB instance, predication is made from the total vote from those trees that did not include that instance in their bootstrap sample. This can be useful, specially in cases where data is not big enough to be partitioned into training and testing sets. Moreover, the structure of RF allows for the calculation of several features importance measures such as node purity-based variable importance. Another by-product of RF models is the RF dissimilarity, which can be used for unsupervised purposes and are discussed next [129].

2.5 Clustering

Clustering is organizing data instances into groups so that data points in the same group share the same properties. There are many popular clustering algorithms in machine learning and statistics including but not limited to k-means [91], the k-medoids [72] and the expectation maximization (EM) [32], each suitable for different applications.

Many clustering algorithms including but not limited to k-means [91] and the k-medoids [72] require a dissimilarity (distance) measure to calculate closeness of data points with each other. Euclidean, Mahalanobis, Cosine, Pearson correlation are among the popular metrics. Notice that some of these metrics provide similarity between data points. Dissimilarity is simply the negation of that. Please refer to [152] for a comprehensive study on different clustering algorithms and possible distance metrics for them.

Some supervised learning models, including RF, can be adapted for unsupervised methods such as clustering by creating a supervised model that classifies generated synthetic data from the actual one [14, 129]. The dissimilarity measures obtained from these supervised models can then be input to the well-known clustering algorithms where clusters can be identified that may or may not correspond to clusters in the Euclidean space [14, 42].

Among the supervised models that can be converted to unsupervised methods, RF has several substantial properties. It can naturally handle both categorical and numerical features from different scales without any necessary attention. Moreover, because it is invariant to the monotonic transformation of variables, it handles features with highly skewed distributions, which makes it even more suitable for some applications such as tumor marker expressions [129].

To achieve RF similarity measurements (also known as RF proximity), instances are assigned by each tree, and if two instances fall into the same terminal node, their similarity is increased by one. The final similarity values are divided by the number of trees so that they lie within $[0, 1]$ [14]. The RF dissimilarity between any two instances is then simply defined by $\sqrt{1 - PROX_{ij}}$ where $PROX_{ij}$ is the RF proximity measurement of x_i and x_j instances. The idea is to create pseudo labels for the actual instances, train a supervised model to distinguish them from synthetic data, and obtain the generated similarity matrix between all instances, including both actual and synthesized ones. This similarity measure strongly depends on how the synthetic data is generated [129]. Although there are several options for generating the synthetic data, we use the unsupervised RF model implemented in the *randomforest* package in R, which is also used in Breiman's FORTRAN implementation. The main idea of this sampling is to break the dependencies between feature variables by randomly sampling from the product of marginal distributions of the features.

Although RF has been used in various studies focusing on classification or regression tasks [9, 26, 34, 136], there is considerably less published work on using RF for AL despite all its useful properties. Aiming at utilizing all these properties, we chose RF to be the main classification algorithm used in this proposal. We also utilize RF dissimilarity to characterize our AL exploration elements because, on top of all the excellent properties that we counted for it, it is in coordination with our classification scheme.

Chapter3

INFORMATION DENSITY FOR ACTIVE BATCH LEARNING

3.1 Introduction

Traditional machine learning techniques, thanks to today's computational power, are able to analyze an enormous amount of data gathered at low cost in modern systems. However, before analysis, labeling this data can be of a hard or expensive endeavor. Active learning (AL) emerged to bridge the gap between data acquisition and data analysis. While data collection processes can accelerate learning by actively choosing the most informative instances to be labeled for model building, most of the data provided to analysts is collected in advance. In other words, models act passively [53].

Applications for AL are ubiquitous. Customer satisfaction measures are less readily available in models to interlink key process indicators to customer satisfaction for quality improvement. Similarly, in many pattern recognition tasks, large numbers of instances are typically available for training, but it is time-consuming for human experts to label all the instances [92, 132]. Rather than selecting images randomly, the most useful ones for model-building can be queried with an AL strategy. For example, many medical images can be generated for clinical diagnostics (etiology),

but labeling each one according to a disease state (a class label) can be difficult. Instead, active learning can be applied to select the images based on their benefits to a predictive classifier [56].

AL usually executes two main steps iteratively; first, building a classifier using the available set of labeled data, and second, querying informative instances to be added to this set. Queries for informative instances are often based on some common strategies. Uncertainty Sampling [81] selects instances in which the labels from the currently trained classifier are uncertain. Query by Committee [125] selects instances with the highest disagreement among several models, trained on the current labeled data. These query approaches define the informativeness of each instance from the supervised learning step. Other strategies employ unsupervised methods, like clustering, to take advantage of the underlying data distribution [28]. Several studies [36, 58, 62, 154] have benefited from both instance informativeness and representativeness criteria to speed up the learning process.

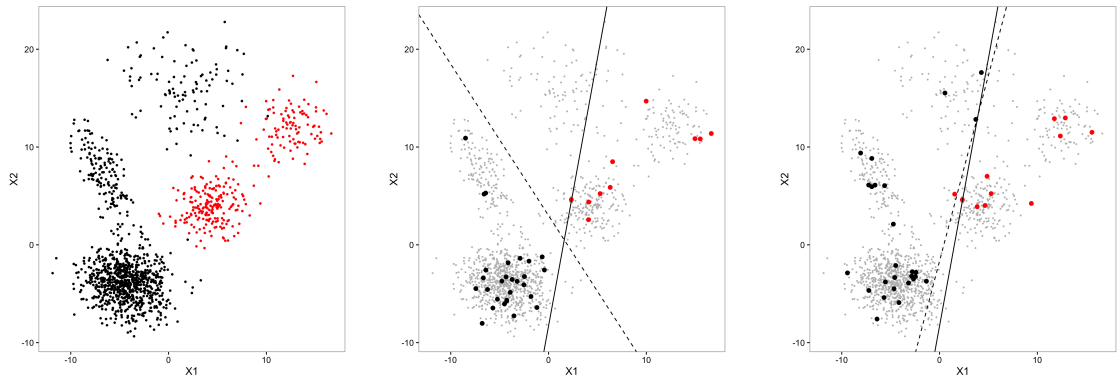
Despite its prevalence, multi-class AL is less investigated due to the difficulty of generalizing the concept of uncertainty from binary to multi-class setting. In order to solve this issue, many approaches for multi-class problems are based on probabilistic models. In short, uncertainty in these models is assessed based on the class membership likelihoods that they obtain. Similar to the binary class problem, to make the most use of labeling resources, many algorithms have utilized the underlying structure of unlabeled data as well as uncertainty [56, 106, 142].

To advance the AL field, the AISTATS Active Learning Challenge was conducted in 2010 [2]. Participants tried to solve six binary classification problems from different domains of chemo-informatics, handwriting recognition, text processing, marketing, ecology, and embryology in which labeling the data can be challenging [53]. The overall best results were obtained by the Stochastic Query-by-Forest (SQBF) algorithm

[8]. This nonparametric AL algorithm, obtained model uncertainties from a random forest (RF) classifier. The ability to handle noisy, mixed (including both categorical and numerical features) and large datasets as well as being able to adapt for regression or multi-class problems are advantages of SQBF over alternatives. SQBF, however, is only based on model uncertainty and does not utilize any information from the underlying distribution of the input data in the instance selection process. Therefore, the good performance can potentially be improved with this additional information. Moreover, the uncertainty measure defined by SQBF can be unstable depending on the data class distribution and how the trees in the RF model are grown. In this study, we present the Cluster-Based Stochastic Query-By-Forest (CSQBF) algorithm, which incorporates the unsupervised knowledge obtained from the unlabeled data in the utility function by simultaneously focusing on cluster information and model uncertainty. CSQBF introduces a more diverse batch querying strategy by searching different areas of the input space for the most informative instances. Like SQBF, our approach constructs a committee of base learners (trees) to obtain measurements of model uncertainty. However, it formulates instance uncertainty within each cluster in a more robust way by utilizing the overall committee decision uncertainty instead of individual base learners. Furthermore, CSQBF uses RF dissimilarity measurements for clustering, which brings the advantage of efficiently handling mixed and/or high-dimensional data for meaningful clustering results in many domains. To evaluate the effectiveness of our proposed AL algorithm, we apply it on several binary class datasets of different sizes.

Figure 3.1 is an illustrative example to show the potential performance of CSQBF over SQBF by accounting for the data distribution. Figure 3.1a shows a simulated dataset of 1350 instances (1050 from class 0 and 300 from class1), and the line represents the decision boundary of a SVM [144] classifier with a linear kernel. Fig-

ures 3.1b–3.1c show the large difference between the quality of classification boundaries based on the queried instances only after five iterations of SQBF and CSQBF, respectively.



(a) Training Data (b) SQBF (c) CSQBF

Figure 3.1: An Illustrative Example of SQBF and CSQBF Performances. Figure (a) is a Dataset of 1,050 Class 0 (Black) and 300 Class 1 (Red) Instances. (b) and (c) Queried Instances After Five Iterations of SQBF and CSQBF, Respectively. Dashed Lines Are the SVM Boundaries With a Linear Kernel Based on the Current Labeled Data, Whereas the Solid Lines Show the SVM Boundary for the Whole Dataset. CSQBF Achieves a Better Classification Performance by Considering Data Clusters. See the Color Version for the Best View.

The remainder of the paper is organized as follows. Section 3.2 provides background on different AL querying strategies and details on the SQBF algorithm. Section 3.3 presents our proposed AL algorithm. Experimental results of our algorithm are provided in Section 3.4, and finally Section 3.5 concludes our study.

3.2 Background

3.2.1 Query Strategies

In "membership query synthesis" AL scenario, a model generates instances from a predefined distribution and asks for the label of those instances. "Stream-based selective sampling" (sequential AL), however, queries synthesized unlabeled instances one by one from a natural distribution, and the learner decides whether to request for

the labels of those instances or discard them [123]. In many domains, a large amount of unlabeled data, compared to labeled data, is available, and instance labels are queried based on their informativeness from the pool of unlabeled data. This "pool-based active learning" scenario [81] is a reasonable approach in many domains [122]. Unlike the first two approaches, pool-based AL assesses the full pool of unlabeled data to select the best query in each iteration. Designing an appropriate criterion for selecting the most valuable instances to query is the main component of any AL strategy. *Informativeness* of an instance is associated with learner's confidence in labeling that instance whereas *Representativeness* measures whether a queried instance is representative enough of the overall input data distribution [63]. There are several querying methods used in the literature, of which we outline the most commonly used ones.

Uncertainty Sampling queries instances for which the trained model is least confident about their labels. For probabilistic learning models, model confidence can be simply defined as the posterior class probability, $p_c(x)$, for data instance x where $c \in \{0, 1\}$ denotes the classes. The greater the $p_c(x)$ value, the more certain the model is about the instance label. Uncertainty measures are then formulated using these probabilities. The simplest algorithm selects instances which maximize $1-\max_c p_c(x)$, whereas entropy-based strategies are the most common ones. See [81] and [122] for more details. Many studies have non-probabilistic strategies such as margin-based approaches in which unlabeled instances closest to the decision boundary are chosen to be queried [100, 119]. While empirical comparisons suggest that the best uncertainty measure is to some extent application-dependent, generally utilizing any of these measures results in a better performance compared to the passive learning approach when the same amount of labeled data is provided to both [122].

Query-By-Committee(QBC) [41, 125] is a query approach where the most valuable

instances are chosen from the pool of unlabeled data based on their utility evaluated by an ensemble of learners. Hence, the most uncertain instances are the ones that most committee members disagree upon their class label [122]. In other words, QBC measures the variance of the data distribution indirectly by constructing the committee members, which together approximate the classification distribution as well as the classification variance [94]. RF [14] is an ensemble of decision tree learners that uses the trees votes to classify a new instance. The class with the majority of votes would be the forest prediction for a new observation.

On top of scalability for large data and handling noisy as well as mixed data (including both categorical and numerical features), RF is naturally applicable to multi-class problems, which avoids one-vs-one and one-vs-others type approaches. Although most of the AL literature is focused on variations of SVM-based strategies, several studies take advantage of RF properties in their works. Work by [30] proposes an RF-based framework where instead of randomly selected instances, data representatives are used to form the initial labeled set. However, data clusters are not benefited from further throughout their algorithm. Study by [8] proposed a different approach based on RF. In a binary class setting, class probabilities in the trees' final nodes are calculated for one class. Then, uncertainty is defined as the standard deviation of a modified version of these probabilities across all the trees. They introduce a cutoff parameter, α, to make a trade-off between randomness and large utility values as inspired by Cawley [16]. Cawley in [16] reckoned that uncertainty sampling methods most likely can achieve a better performance if they explore the input data space more. In another AL study using RF [50], they proposed a query criterion that combines uncertainty, density, and diversity factors in a linear fashion (details follow later). They defined density as a function of average distance to the k-nearest neighbors. In contrast, diversity is calculated based on the distance between an unla-

beled data instance and its nearest labeled neighbor. Even though their method tries to exploit the unlabeled data structure, we show that a better performance can be achieved if randomness is incorporated with the sampling strategy.

Uncertainty sampling and QBC, despite their ease of use and popularity, may suffer from what described as sampling bias [28]. During AL, instances are queried based on their informativeness assessed by model confidence in labeling. However, models built around these querying strategies are more likely to choose outliers in the input space of predictors (because they are probable to look informative based on the employed uncertainty measure).

Density-based methods, on the other hand, take into account the data density in different regions of the input space by modifying the utility function to $U(x) \cdot D(x)^{\lambda}$, where $U(x)$ is a measure accounting for uncertainty, and $D(x)$ is a measure of the input density. Parameter $\lambda \in \mathbb{R}^{+}$ controls the effect of the density factor [8, 53]. Previous studies have shown the contribution of the global input distribution in AL by bringing the data density into account (e.g., [100, 137] for more information).

3.2.2 Multi-class Scenario

Multi-class active learning scenarios are less studied mostly due to the difficulty of generalizing the concept of uncertainty from binary to multi-class problems. As we mentioned, uncertainty can be defined as the likeliness of a data instance to be misclassified. Distance from the classification hyperplane has been used as a notion of uncertainty where margin-based classifiers such as support vector machines (SVM) are utilized [122]. However, these classifiers are only directly applicable to binary class problems. Converting multi-class problems to several pairwise comparisons between classes or to multiple one-vs-all subproblems have been employed to overcome this issue [66, 83, 106, 155]. Work by [106] for example, adopts a one-vs-all strategy to

solve a multi-class active learning problem with SVM classifier. In their algorithm, the most uncertain samples corresponding to each binary SVM are identified using SVM output scores and a threshold value. The first shortcoming of such approaches is that the classifiers are not independent from each other, and uncertainty cannot be assessed across multiple classes. For example, an uncertain instance in one pairwise setting can be considered as certain in another one. Consequently, the model cannot tell which classes need more data instances to be queried from the pool of unlabeled data [157]. In other words, due to the presence of multiple hyperplanes, defining uncertainty by the closeness to the classification boundary does not extend well to the multi-class scheme. In order to tackle this problem, many recent studies have developed probabilistic models to obtain instance uncertainties. Work by [66] first used a modified version of Platt's method to obtain class membership probabilities from a one-vs-one SVM classifier. Later, a Best-vs-Second-Best approach was used as their measurement of uncertainty i.e., instances with the lowest difference between the two highest class membership probabilities, are taken as the more uncertain ones.

In order to make use of both labeled and unlabeled data to enhance the learning performance, semi-supervised learning methods are becoming more and more popular in AL studies [160]. Algorithms attempt to exploit the relationship between clusters and labels by getting help from a common assumption in semi-supervised paradigm which states that data can be grouped into several clusters and instances of a cluster are more likely to have similar class labels [20, 150]. Among heuristic methods that have been proposed to integrate classification uncertainty and the data distribution elements, many of them encourage the selection of cluster centers. For example, as we mentioned before, medoids of the clusters obtained from the Partition Around Medoids (PAM) clustering [71] were used as the initial set of emails to label in a spam detection problem [30]. However, no knowledge about the input data space is

conveyed to future iterations. The unlabeled data points lying in the margin of an SVM classifier, trained with the current labeled data, were clustered and medoids were selected to be labeled [154]. Work by [100] incorporated clustering with AL based on logistic regression and SVM classifiers. Their algorithm builds a model using cluster representatives (such as medoids), and iteratively labels instances closer to the decision boundary and cluster representatives. At each iteration, after rebuilding the classifier, the algorithm assigns the same label as the cluster representative to all other cluster members. By doing so, the algorithm diversifies the samples by avoiding querying instances from the same clusters. They also use a so-called coarse-to-fine strategy to update the clusters. Work by [62] has designed a min-max margin-based framework that takes into account both representativeness and informativeness. Their method labels one instance at a time. One major limitation of such approaches is that they can be more costly than running a batch-mode active learner once and perform labeling in parallel. Furthermore, they would not benefit from the potential good performance of ensemble models since it would be computationally expensive to assess the unlabeled data pool every time after only one label is queried [122].

Research by [64] has introduced an AL strategy that uses a variant of K-Nearest Neighbors (KNN) as a classifier along with a similarity measure suitable for multi-class problems with large number of classes. In [157], Yang *et al.* have used a Markov Chain model to develop a compound criteria based on uncertainty and diversity in a visual concept recognition task. Their algorithm tries to avoid labeling similar data instances simultaneously by maximizing uncertainty and diversity at the same time. Work by [58] uses k-means clustering, which benefited from lower computational complexity compared to the other clustering methods. However, Euclidean-based methods, like k-means, may not be suitable for high dimensional data, and our tree-based ensembles naturally handle attributes scales, interactions, high-dimensional,

and categorical attributes.

RF, which was originally developed as a supervised classifier, can also be used in unsupervised studies, and has been shown to perform well in many domains [14, 130]. Like many other unsupervised algorithms, RF clustering needs a measure of dissimilarity between instances. The similarity of two instances i and j, S_{ij} in unsupervised RF is the proportion of trees in which those instances are assigned to the same terminal nodes in the forest. Dissimilarity then is simply defined as $(1 - S_{ij})$ or $(\sqrt{1 - S_{ij}})$ between instances i and j. When data are of a high-dimension, measures such as Euclidean distance might be challenged by data sparsity; instances might appear equally-distanced from each other [105]. Moreover, in the case of mixed data, preprocessing or alternatives methods are needed to discover the clusters [80]. RF clustering, on the other hand, is capable of handling high-dimensional, mixed data, and missing values. Furthermore, several studies have shown that RF clustering results are more meaningful in their study domain than Euclidean-based ones [14, 130].

3.2.3 Stochastic Query-By-Forest (SQBF)

In this section we re-examine the stochastic sampling process of SQBF since our approach is derived from that. Consider a binary class classification problem. SQBF builds an ensemble of shallow trees as committee members on an initially queried labeled dataset. For each unlabeled instance, class probabilities are estimated based on the class distribution in the leaf node of the t^{th} tree to which that instance is assigned. For $x_u \in X_{unlabeled}$, the estimated rare class probability in the t^{th} tree is denoted by $p_{tc}(x_u)$, $t = 1, 2, \ldots T$, where T is the number of trees in the forest, $c \in \{0, 1\}$ is the rare class in the final node, and $X_{unlabeled}$ is the pool of unlabeled

data. These probabilities are weighted with class priors as

$$p'_{tc}(x_u) = \frac{p_{tc}(x_u)/p_c}{\sum_c p_{tc}(x_u)/(1-p_c)} \tag{3.1}$$

where $t = 1, 2, \ldots T$. The standard deviation of the weighted rare class probabilities is used to calculate the uncertainty measurement, $q(x_u)$, for unlabeled instance x_u as

$$q(x_u) = sd(p'_{tc}(x_u)) \tag{3.2}$$

The higher the $q(x_u)$ is for an unlabeled instance, the more uncertain the model is about the label, making that instance more likely to be queried [8]. Although the standard deviation in Eq. (3.2) might not be an accurate measure of model variance, the relative magnitudes of $q(x_u)$ values provided effective performance for the AL strategy in [8].

However, as instances with the highest uncertainty measurements might be very similar to each other, they may provide the same information to the model [56]. To avoid this problem, SQBF brings some randomness into the instance selection process by introducing parameter α. Query candidates are the top α proportion of the ranked instances based on their $q(x_u)$ values. A sampling probability distribution $p(x_u)$ is generated consisting of the normalized $q(x_u)$ values for all query candidates. Instances to be labeled are queried from the candidates randomly, proportional to $p(x_u)$. Queries for labels are made in a batch, where batch sizes increase exponentially in a way that they sum up to the unlabeled data pool size.

3.3 Cluster-based Stochastic Query-By-Forest

We propose a new batch-mode active learning algorithm which not only advances SQBF by introducing cluster-wise utilities incorporated with instance utilities, but also can be extended to multi-class scenarios and is scalable for large datasets. To

Algorithm 1 Cluster-based Stochastic Query-By-Forest (CSQBF)

$D(X) = 1 - S(X)$ ▷ Transform matrix of similarities to dissimilarities

$l = 1$ ▷ Initialize iteration

PAM $\leftarrow \{D(X), K\}$ ▷ Implement PAM to K clusters with dissimilarities

Query medoids for labels

while Termination condition not reached **do**

  Build RF, $\mathcal{G}^{(l)}$, with $X^{(l)}_{labeled}$ ▷ Build classifier

  Compute $Mrg(x_u)$, where $x_u = \{x | x \in X^{(l)}_{unlabeled}\}$ ▷ Compute instance uncertainty

  for each cluster k **do**

    Obtain $|C^{(l)}_k|$ and compute $\overline{Mrg}^{(l)}_k$ ▷ Compute cluster uncertainty

    Compute $u^{(l)}_k$ ▷ Compute cluster utilities

    Compute $a^{(l)}_k$ ▷ Assign query sizes

    Select instances by $p^{(l)}_k(x)$, where $x = \{x | x \in C^{(l)}_k\}$

  end for

  Query selected instances for labels

  Adjust $X_{labeled}, X_{unlabeled}$

  $l = l + 1$ ▷ Proceed to next iteration

end while

Return $\mathcal{G}$

highlight these differences, we call our new algorithm cluster-based SQBF (CSQBF). Initially, CSQBF performs pre-clustering to build the first query with the most representative instances. In order to do so, we use PAM with RF dissimilarity. The resulting medoids of PAM are then queried for labels. Alternative clustering algorithms can be used, but we prefer to handle mixed data in the clustering algorithm

as easily as it is done in the classification as well as the instance selection process of CSQBF. Here, stratified sampling is used to address the class imbalance. To balance classification accuracies across the classes, we downsample instances from the majority class in the training sample for each tree, so that base learners are trained on equal counts from both classes.

Next, to account for uncertainty, we define margin, $Mrg(x)$, as the difference between the two class probability estimates for instance x, i.e. for each $x_u \in X_{unlabeled}$

$$Mrg(x_u) = |Pr_1 - Pr_0| \tag{3.3}$$

where Pr_c is the relative number of votes (predicted class probability) for class c, $c \in \{0, 1\}$.

From a classification perspective, a classifier is more uncertain about the label of those instances that have a low margin value because the two class membership probabilities would be closer in this case. Therefore, $1 - Mrg(x)$ would be model uncertainty for instance x. A similar approach is used by [30] in which instances with their greatest class membership probability closest to 0.5 are queried for labeling. This approach would be consistent with ours as shown by [122]. In the following iterations of the algorithm, utility of cluster k, u_k, where $k = 1, 2, \ldots, K$, is evaluated by cluster size and cluster uncertainty. The size of cluster k, $|C_k|$, is defined as the number of unlabeled instances in the cluster. Next, for $k \in 1, 2, \ldots K$,

$$\overline{Mrg}_k = \frac{1}{|C_k|} \sum_{x \in C_k} 1 - Mrg(x) \tag{3.4}$$

is calculated in order to obtain cluster uncertainty (average instance uncertainty within a cluster) as $1 - \overline{Mrg}_k$, where instance uncertainties, $Mrg(x)$, are computed by Eq. (3.3). Cluster utility, u_k, is then obtained as a product of normalized cluster

uncertainty and normalized cluster size

$$u_k = \frac{\overline{Mrg}_k}{\sum\limits_k \overline{Mrg}_k} \cdot \left[\frac{|C_k|}{\sum_k |C_k|}\right]^{\lambda} \tag{3.5}$$

where $\lambda \in \mathbb{R}^+, k = 1, 2, \ldots, K$ controls the relative importance of the two factors.

After cluster utility u_k is obtained, each cluster is assigned a query size. Given the total query size at iteration l, $a^{(l)}$, where $l = 1, 2, \ldots, L$, the query size of cluster k, $a_k^{(l)}$, is determined proportionally to its cluster utility

$$a_k^{(l)} = a^{(l)} \cdot \frac{u_k^{(l)}}{\sum\limits_k u_k^{(l)}} \tag{3.6}$$

for $k = 1, 2, \ldots, K$. Next, the top α proportion of the instances arranged in an ascending order based on their $Mrg(x)$ value are taken as query candidates within each cluster represented by QC_k where $k = 1, 2, \ldots, K$. Sampling probabilities for $x \in QC_k$ are obtained using Eq. (3.7).

$$p_k(x) = \frac{Z(x)}{\sum\limits_{x \in QC_k} Z(x)} \tag{3.7}$$

where $Z(x)$ is the normalized version of sampling probabilities for $x \in QC_k$ defined as

$$Z(x) = \frac{Mrg_{max} - Mrg(x)}{Mrg_{max} - Mrg_{min}} \tag{3.8}$$

In other words, unlike SQBF that defines one sampling distribution for the pool of unlabeled data at each iteration, CSQBF considers separate ones for each cluster at each iteration. Instance sampling in cluster k is performed with regard to the corresponding sampling distribution, $p_k(x)$, where $x = \{x | x \in C_k^{(l)}\}$, $C_k^{(l)}$ denotes the unlabeled instances remained in cluster k at iteration l.

Instance utility, $Mrg(x)$, cluster utility, $\overline{Mrg}_k$, cluster size $|C_k|$, and accordingly $p_k(x)$ as well as u_k are updated at each iteration with respect to the remaining unla-

beled instances. The RF model $\mathcal{G}$ is also rebuilt whenever new labeled instances are added. Algorithm 1 summarizes this process.

A good strategy for AL, is to query labels for the most uncertain instances, sample from dense regions of input space, and impose diversity of query instances [153]. CSQBF addresses all these through cluster and instance utilities. Large $\overline{Mrg_k}$ represents that instances remaining in the cluster have high uncertainties. Choosing more samples from clusters that contain more uncertain instances will be more effective in estimating decision boundaries. Cluster size is expected to be associated with density. If each cluster covers regions of similar size in input space, the clusters with greater counts are denser. In this sense, taking more samples from large clusters encourages sampling from dense regions. Moreover, sampling from each cluster increases the diversity of the selected instances.

3.4 Experiments and Results

The SQBF algorithm was compared to several other algorithms during the AISTATS 2010 AL challenge [2] in which it was ranked first. Therefore, we compared CSQBF with the following methods:

- *MARGIN*: Non-parametric margin-based AL approach by [140] that calculates instance informativeness based on distance to the SVM decision boundary.
- *ENTROPY*: Entropy-based uncertainty sampling using SVM [109].
- *SQBF*: Non-parametric AL approach based on work by [8].

In all the experiments, an RBF kernel is used for SVM-based models (MARGIN and ENTROPY). Several binary-class datasets were used to test the performance of CSQBF. Experimental datasets were chosen from a wide range of characteristics including domain, class imbalance ratio, number of instances, and feature dimension.

Table 3.1: Datasets Used in This Study.

Dataset	Train	Test	Features	Classes	Min %
Banknote	686	686	4	2	23.52 %
Credit	15000	15000	23	2	22.12 %
Digit1	750	750	241	2	48.93 %
EEG	7490	7490	14	2	44.88%
Ibn Sina	10361	10361	92	2	37.84%
Letter NvsM	788	787	16	2	49.71%
Letter VvsY	788	787	16	2	49.71%
Madelon	1300	1300	500	2	50 %
MNIST35	5776	5776	784	2	46.93%
Mushroom	4062	4062	22	2	48.20%
Occupancy	10280	10280	5	2	2.53 %
Spambase	2300	2301	57	2	39.40%
Steel	970	970	27	2	34.67%
Transfusion	374	374	4	2	23.80%
Twitter	490	123	291	2	44.7%

Ibn Sina is obtained from AISTATS 2010 AL challenge [2]. Banknote, Default Credit, EEG, Madelon, Mushroom, Letter NvsM, Letter VvsY, Occupancy, Spambase, Steel, and Transfusion are from the UCI Repository [84]. Letter NvsM and Letter VvsY are subsets of the Letter dataset. Digit1 is a benchmark dataset for semi-supervised learning [19]. MNIST35 is a subset of MNIST Handwritten Digits database [79], which includes only images of numbers 3 and 5. The Twitter dataset is annotated Twitter data by cyberbullying experts in work by [151]. In order to show the severeness of cyberbullying, from the publicly available 2011 TREC Microblog track corpus, 990 tweets were uniformly sampled for manual inspection by five annotators. As it is a hard task to follow bullying traces among all the tweets, authors

created an enriched dataset by collecting tweets from the public Twitter streaming API that contain at least one of the words "bully", "bullied", and "bullying". Retweets were further removed. The same annotators who labeled the TREC corpus labeled 1762 tweets sampled uniformly from the enriched dataset. Among them, 684 (39%) were labeled as bullying traces.

As per terms of services of Twitter, the original labeled tweets are not included in the dataset used by [151]. So, we crawled the original tweets through Twitter API by the tweet ids provided online at [1]. By the time we crawled the data, some of the tweets were no longer available, most likely due to deletion by their owners (obtained only 1092 out of 1762 original labeled tweets). After the non-English tweets were discarded (resulting in 613 tweets), the tweets were case-folded. English stopwords and excessive punctuations were removed, and any user mentions preceded by an "@" were replaced by the anonymized user name $@USER$. Any URLs starting with "*http*" were replaced by the token "URL". Hashtags were also replaced by "$\#HASHTAG$". Moreover, based on our experiments, all the single-character words could be removed since they are used differently by users. Some of these words are used as a substitute for English stopwords such as "*and*" that appears as "n" or "*the*" that shows up as "d". We used unigrams and bigrams (which would automatically include the number of hashtags as well as referred users and websites since we treated them as tokens and did not discard them), number of words and length of the tweet as the textual features. This process created over 6000 features. We used a frequency threshold of 5 to choose the final set of bigrams and unigrams and discarded those tokens that did not have enough occurrences among all the tweets. This reduced the number of features to 291.

We split all the original datasets into two halves; one used for training and the other for testing and estimating the generalization error. For Ibn Sina dataset, only

the development (training) datasets of the challenge were used here. Our data partition reduced the training set sizes to 50% of the challenge. For the UCI datasets, we partitioned the whole datasets into two halves. Table 3.1 summarizes the experimental datasets.

3.4.1 Clustering

We used PAM with RF dissimilarity as the distance to cluster instances of each dataset. In order to choose the best number of clusters for CSQBF, we compared the value of mean dissimilarity between cluster medoids and instances for different values of K ranging from 2 to 15. Figure 3.2 presents the results. We can observe the gradually decreasing trends on every dataset, but without any noticeable steep decline (knee) or turn-over. Here, the maximum number of terminal nodes in each tree is set to 5 ($maxnodes = 5$). Because pruned decision trees typically consist of 3 to 12 terminal nodes [26], we also tested $maxnodes = 4, 8$, and 12, but similar patterns in dissimilarity occurred as for $maxnodes = 5$. Therefore, we set the number of clusters to 7 for all datasets, and the RF dissimilarity is obtained with $maxnodes = 5$. Potentially better performance could be achieved with the number of clusters customized for each data set. The initial labeled set for all methods consists of a common pair of initial positive instances (different pairs in each replicate) and $K = 7$ other instances, which are cluster medoids for CSQBF and randomly selected instances for MARGIN, ENTROPY, and SQBF. The 2 positive instances were selected to ensure that RF models, $\mathcal{G}$, at the first iteration were built with the same instances from the positive class especially in case of imbalanced datasets.

For the next 10 iterations, after the initial query, we follow the approach used by [8] to set the query sizes. They designed the sequence of query batch sizes such that they exponentially increase over iterations and eventually sum to the total number of

instances, N. Therefore, Eq. (3.9) is solved for b as the base of an exponential query sequence.

$$\sum_{l=1}^{11}(K+2)\cdot b^{l-1} = N \tag{3.9}$$

Alternative approaches such as a fixed query size at each iteration can be used based on the domain, but they do not alter the nature of the problem.

Unlike SQBF that grows shallow trees based on the labeled data size at each iteration, we use fully grown ones in the supervised RF model. This setting was made based on our preliminary test results that showed a noteworthy performance improvement when allowing trees to grow fully.

To account for class imbalance, stratified sampling was used for all methods. Also, throughout the experiments, we noticed that sometimes instances within clusters were occasionally exhausted since a query assignment for a cluster (with greater uncertainty) outnumbered the cluster size, $|C_k|$, by a small number at later iterations. In this case, assignment backlogs were redistributed to available clusters proportionally to their cluster utilities. Also, for each dataset, we run the experiments 15 times to

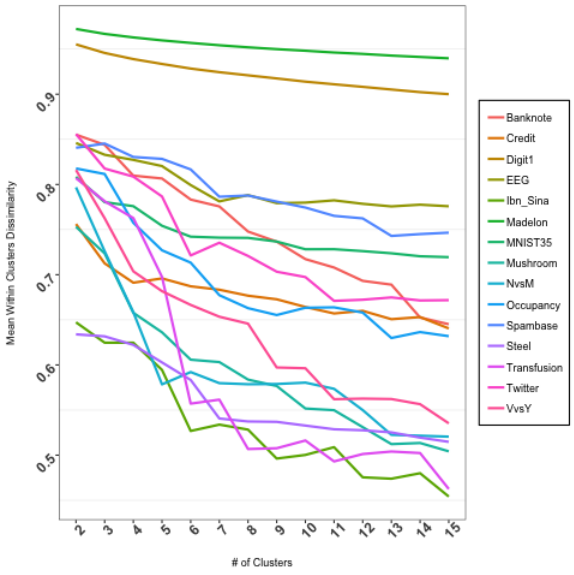


Figure 3.2: Mean Dissimilarity Between Instances and Cluster Medoids Obtained From PAM. See Color Version for the Best View.

address the inherent randomness of RF bagging procedure that would cause slightly different dissimilarities.

In active learning classifications, prediction accuracy at early iterations plays a key role in the algorithm performance as it tends to improve with more labeled instances. Due to the limitation of accuracy measure for imbalanced datasets, we evaluated the performance of CSQBF by F1-score which is the harmonic mean of precision and recall [133] i.e. if ρ_i and π_i are precision and recall for each class i then:

$$F_i = \frac{2\pi i \rho i}{\rho_i + \pi_i}, \qquad \text{F1-score } = \frac{\sum_{i=1}^{C} F_i}{C} \tag{3.10}$$

where C is total number of classes. F1-score gives equal weight to all classes so it is more influenced by the classifier's performance on the rare categories [103]. Although not provided here, we furthered examined CSQBF against other methods based on accuracy. The results were compatible with F1-score results and they show the outperformance of CSQBF.

Table 3.2 presents the mean F1-Score of 15 replicates for each method through 11 iterations. For each iteration, paired t-tests at 95% significance level were conducted and the best performance and its comparable performances are highlighted in boldface. This table illustrates how CSQBF performs significantly better than all methods constantly throughout all iterations for most of the datasets. In particular, CSQBF showed greater F1-scores at the early iterations on many datasets such as Banknote, Ibn Sina, Mushroom, Letter NvsM as well as VvsY, and Transfusion.

The most substantial improvement of CSQBF at the first iteration was observed in Letter VvsY, followed by Transfusion and Ibn Sina datasets. Overall, CSQBF surpasses all methods, including SQBF, for almost all datasets from the first iteration except for Madelon for which CSQBF results are similar to those of SQBF for the first 8 iterations. The 0 F1-score values in some of the final iterations of the ENTROPY

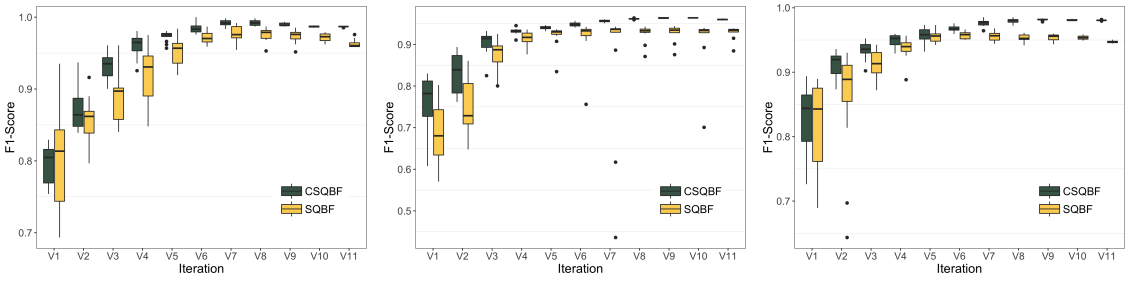
algorithm indicate the failure of SVM to correctly classify any of the positive class instances. Note that in F1-score calculations, we chose the minority class to be the positive one for all the datasets. Also, in these cases the accuracy is still more than 50%. Moreover, despite the fact that we tried to tune the parameters for the baseline SVM-based models, poor performance (especially in the early iterations) compared to SQBF and CSQBF might be due the natural classification difference between RF and SVM.

Table 3.2: Mean F1-Score Values Comparison. *At Each Iteration, the Best Performance and Its Comparable Performances Based on Paired T-tests at 95% Significance Level Are Highlighted in Boldface.*

Data	Algorithm	Iteration										
		1	2	3	4	5	6	7	8	9	10	11
Banknote	MARGIN	**74.12%**	**83.95%**	93.66%	94.21%	94.95%	95.85%	97.68%	97.66%	98.24%	99.19%	99.35%
	ENTROPY	**74.12%**	79.48%	85.28%	**92.2%**	**95.31%**	**98.48%**	98.37%	98.81%	99.35%	**99.51%**	**99.51%**
	SQBF	**77.38%**	82.58%	88.72%	91.53%	95.57%	97.01%	98.06%	97.65%	97.64%	97.2%	96.05%
	CSQBF	**79.59%**	**86.95%**	**93.61%**	**96.63%**	**98.14%**	**99.09%**	**99.41%**	**99.33%**	**98.78%**	98.64%	98.69%
Credit	MARGIN	15.16%	18.54%	16.45%	27.71%	36.92%	34.46%	33.53%	34.46%	41.4%	42.78%	45.34%
	ENTROPY	15.16%	11.78%	7.39%	8.98%	13.57%	20.81%	25.16%	35.57%	42.24%	44.96%	44.16%
	SQBF	**34.67%**	**35.43%**	**38.01%**	39.35%	42.27%	43.39%	43.31%	42.19%	41.02%	39.96%	42.95%
	CSQBF	**33.43%**	**35.2%**	**38.92%**	**44.09%**	**46.56%**	**49.68%**	**51.45%**	**52.17%**	**52.98%**	**53.3%**	**53.81%**
Digit1	MARGIN	51.12%	47.52%	56.68%	60.58%	65.98%	73.62%	85.96%	91.37%	93.52%	95.88%	96.58%
	ENTROPY	51.12%	42.03%	57.55%	65.4%	**78.41%**	**79.25%**	91.98%	**95.25%**	**96.9%**	**97.43%**	**97.2%**
	SQBF	72.8%	**75.58%**	80.34%	85.26%	89.32%	**92.03%**	93.24%	94.01%	94.22%	94.12%	94.03%
	CSQBF	**77.81%**	**79.47%**	**84.92%**	**87.38%**	**90.89%**	**92.56%**	**94.62%**	**95.69%**	**96.76%**	97.23%	**97.27%**
EEG	MARGIN	**39.85%**	34.49%	**51.43%**	**57.85%**	60.81%	65.34%	64.95%	66.68%	52.99%	70.61%	74.57%
	ENTROPY	**39.85%**	33.84%	29.8%	34.8%	47.03%	58%	66.47%	68.14%	68.41%	63.86%	72.55%
	SQBF	**52.72%**	**52.82%**	**57.04%**	**59.49%**	61.99%	**65.56%**	66.59%	67.9%	68.11%	68.01%	66.11%
	CSQBF	**48.45%**	**52.94%**	**56.28%**	**59.66%**	**63.36%**	**68.07%**	**73.99%**	**79.64%**	**85.43%**	**89.45%**	**92.14%**
Ibn_sina	MARGIN	41.74%	15.56%	23.53%	43.93%	61.52%	88.25%	90.95%	93.01%	93.61%	93.66%	94.18%
	ENTROPY	41.74%	34.79%	42.3%	47.37%	66.88%	62.71%	74.48%	85.9%	93.86%	93.62%	93.98%
	SQBF	70.54%	77.6%	87.19%	91.27%	92.72%	91.49%	86.92%	**87.73%**	93.03%	91.48%	93.37%
	CSQBF	**76.24%**	**85.02%**	**91.69%**	**93.32%**	**94.12%**	**94.9%**	**95.76%**	**96.26%**	**96.56%**	**96.38%**	**95.92%**
Letter_NvsM	MARGIN	68%	78.49%	84.91%	93.68%	94.48%	95.57%	96.01%	97%	97.22%	97.49%	98.43%
	ENTROPY	68%	77.54%	84.04%	88.55%	92.56%	**96.18%**	**97.68%**	98.52%	98.87%	98.77%	98.89%
	SQBF	**81.56%**	88.76%	92.65%	93.97%	94.99%	95.33%	95.19%	**95.55%**	95.47%	95.42%	94.7%
	CSQBF	**82.49%**	**92.01%**	**94.86%**	**95.5%**	**96.67%**	**97.44%**	**97.93%**	**98.22%**	**98.18%**	**98.1%**	**98.05%**
Letter_VvsY	MARGIN	64.9%	78.38%	81.66%	90.41%	94.7%	**96.75%**	98.09%	**98.57%**	**98.54%**	**98.88%**	**98.77%**
	ENTROPY	64.9%	75.32%	83.1%	87.51%	90.82%	**94.3%**	**96.11%**	97.44%	97.81%	98.39%	98.6%
	SQBF	78.48%	87.76%	93.27%	94.92%	95.97%	96.04%	96%	96.04%	96.18%	96.18%	95.88%
	CSQBF	**86.98%**	**93.79%**	**95.28%**	**96.04%**	**96.55%**	**97.37%**	**98.39%**	**98.51%**	98.35%	98.29%	98.36%
Madelon	MARGIN	**57.84%**	**41.17%**	**40.33%**	31.22%	36.72%	41.61%	51.55%	56.02%	54.89%	55.29%	56.94%

	ENTROPY	**57.84%**	**54.04%**	**44.57%**	**48.98%**	**56.05%**	**54.07%**	**53.11%**	**60.36%**	59.39%	58.06%	58.50%
	SQBF	**60.29%**	**52.07%**	**51.94%**	**53.91%**	**55.11%**	**56.72%**	**59.86%**	**62.21%**	**65.31%**	**67.22%**	**70.16%**
	CSQBF	**51.16%**	**52.44%**	**49.85%**	**47.67%**	**49.59%**	**50.91%**	**55.28%**	**58.62%**	61.79%	63.64%	67.29%
MNIST35	MARGIN	29.35%	29.35%	4.19%	12.6%	4.19%	0%	62.9%	62.9%	25.16%	12.58%	16.77%
	ENTROPY	29.35%	29.35%	4.19%	12.6%	4.19%	0.01%	0%	0%	0%	0%	0%
	SQBF	**69.54%**	**78.7%**	**84.82%**	**89.7%**	91.76%	93.8%	94.32%	94.32%	94.33%	94.5%	94.43%
	CSQBF	**73.44%**	**81.31%**	**87.22%**	**90.76%**	**94.15%**	**96.09%**	**97.61%**	**98.37%**	**98.79%**	**98.79%**	**98.66%**
Mushroom	MARGIN	64.94%	67.04%	71.24%	83.18%	92.15%	93.78%	95.13%	97.33%	99.21%	99.91%	**99.99%**
	ENTROPY	64.94%	70.68%	74.87%	80.99%	88.29%	89.76%	94.38%	98.83%	99.75%	**100%**	**100%**
	SQBF	**83.66%**	**91.93%**	**94.88%**	**98.47%**	99.2%	99.57%	99.72%	99.86%	99.88%	99.95%	**99.95%**
	CSQBF	**89.43%**	**93.53%**	**97.08%**	**98.92%**	**99.58%**	**99.9%**	**99.99%**	**100%**	**100%**	**100%**	**100%**
Occupancy	MARGIN	76.13%	84.84%	92.75%	92.92%	94.22%	96.92%	97.41%	97.68%	97.65%	97.68%	97.68%
	ENTROPY	76.13%	**84.37%**	87.49%	85.63%	**85.41%**	**95.65%**	97.58%	97.64%	97.63%	97.65%	97.69%
	SQBF	**83.49%**	86.49%	88.7%	91.37%	92.73%	92.75%	92.78%	92.76%	90.45%	91.67%	96.5%
	CSQBF	**84.96%**	**93.24%**	**96.78%**	**97.42%**	**97.57%**	**97.63%**	**97.74%**	**97.85%**	**98.06%**	**98.13%**	**98%**
Spambase	MARGIN	32.57%	13.41%	6.14%	6.3%	6.56%	77.97%	82.19%	86.54%	88.99%	89.12%	90.21%
	ENTROPY	32.57%	27.04%	25.06%	11.5%	18.44%	28.01%	41.94%	70.96%	85.79%	90.44%	91.77%
	SQBF	**75.27%**	79.88%	**84.14%**	86.13%	87.67%	88.52%	88.64%	88.13%	88.03%	88.05%	88.59%
	CSQBF	**76.17%**	**82.95%**	**85.61%**	**87.99%**	**90.53%**	**91.96%**	**92.42%**	**93.41%**	**93.94%**	**94.15%**	**93.87%**
Steel	MARGIN	**37.75%**	41.96%	41.92%	43.5%	44.27%	46%	55.4%	58.18%	57.18%	59.97%	62.68%
	ENTROPY	**37.75%**	35.66%	37.82%	40.45%	46.66%	49.82%	53.66%	56.49%	58.7%	61.84%	63.43%
	SQBF	**47.19%**	**49.58%**	**51.63%**	**52.29%**	**54.19%**	**56%**	57.98%	58.98%	60.56%	60.81%	60.68%
	CSQBF	**49.9%**	**48.44%**	**48.36%**	**52.26%**	**54.14%**	**57.91%**	**62.17%**	**64.46%**	**66.03%**	**67.4%**	**68.02%**
Transfusion	MARGIN	22.08%	10.56%	7.25%	17.03%	6.94%	6.21%	8.47%	4.17%	3.26%	2.57%	10.82%
	ENTROPY	22.08%	18.91%	18.88%	12.68%	13.99%	12.14%	16.96%	18.27%	20.36%	11.19%	9.57%
	SQBF	34.27%	**38.09%**	**42.78%**	**43.85%**	**45.84%**	**48.04%**	**48.93%**	**48.95%**	48.98%	48.54%	48.39%
	CSQBF	**41.89%**	**43.28%**	**45.25%**	**44.17%**	**45.33%**	**46.88%**	**48.51%**	**49.6%**	**50.53%**	**51.85%**	**52.39%**
Twitter	MARGIN	48.75%	**41.61%**	13.28%	12.66%	12.04%	11.47%	6.06%	4.57%	13.58%	7.17%	13.2%
	ENTROPY	**48.75%**	28.43%	39.87%	29.21%	31.42%	30.51%	28.07%	26.73%	38.13%	43.75%	53.77%
	SQBF	**55.87%**	**56.18%**	59.62%	**59.64%**	**61.98%**	**65.42%**	**65.71%**	**66.29%**	**66.33%**	**67.52%**	67.01%
	CSQBF	21.12%	**56.92%**	**64.51%**	**65.54%**	**67.72%**	**67.4%**	**67.36%**	**68.96%**	**68.29%**	**67.8%**	**68.92%**

We further compared CSQBF and SQBF in terms of variation in their performances to show the stability of CSQBF from early on. Figure 3.3 compares the inter-quartile range (IQR) of F1-scores throughout the iterations for *Banknote*, *Ibn Sina* and *Letter NvsM* datasets. The results for other datasets were similar. CSQBF had smaller IQRs than SQBF, i.e., more stable results for most datasets, especially at the early iterations. There were a few cases in which CSQBF had a wider IQR at some iterations, but it still outperformed SQBF in terms of F1-score and accuracy.



(a) *Banknote* (b) *Ibn_Sina* (c) *Letter_NvsM*

Figure 3.3: Variance (IQR) of F1-Score Over 15 Replicates ($\lambda = 1$; $\alpha = 1/3$). CSQBF Results Show Smaller Variance Than SQBF.

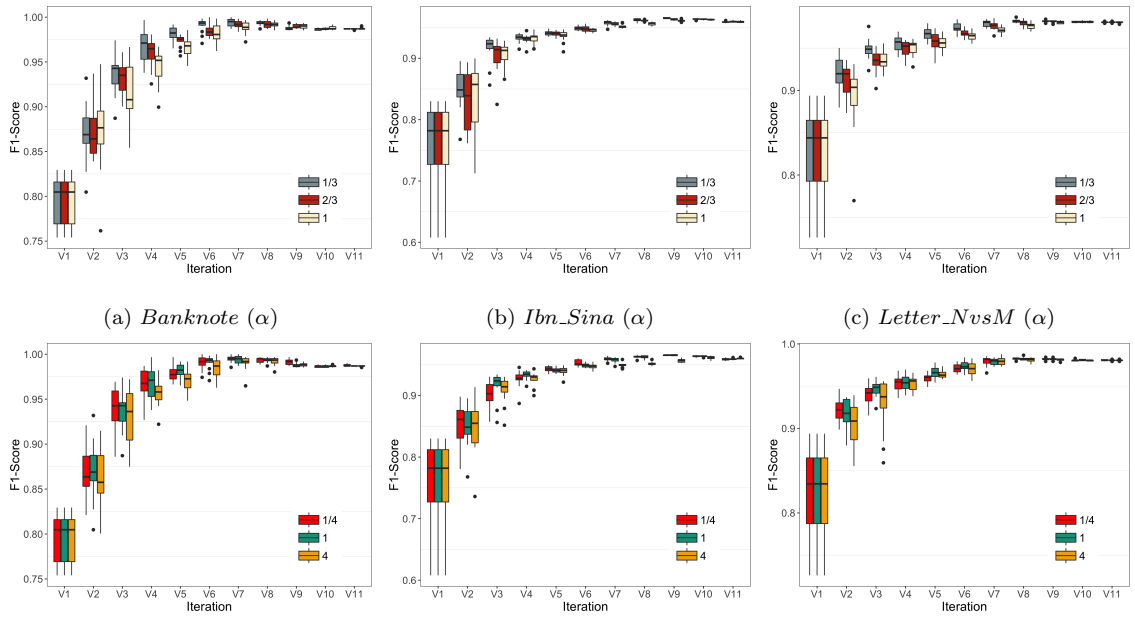
3.4.2 Complexity Analysis

CSQBF is comparatively fast because it benefits from the computational efficiency of RF. Furthermore, unlike SQBF it does not explore the detail class distribution of the final nodes and despite its extra computation in pre-clustering phase, clustering is performed only once at the beginning. One extra forest is the penalty which becomes less important as more unlabeled data gets labeled. The time complexity of the pre-clustering phase is $\mathcal{O}(tFN\log(N)) + \mathcal{O}(K(N-K)^2)$. The first term is due to random forest complexity to calculate dissimilarities between instances in a dataset of size N. t and F are number of trees and features used in each split respectively. The second term is the computational complexity of PAM algorithm where K is number of desired clusters.

The RF algorithm caps the complexity of the main part of CSQBF sampling procedure. In each iteration for data of size N it takes $\mathcal{O}(tFN\log N)$ time to train an RF model on labeled instances and get class probability predictions for unlabeled ones. The total complexity of CSQBF would be$\mathcal{O}(tFN\log(N)) + \mathcal{O}(N_{iter} * tFN\log N)$ where N_{iter} is the preset number of iterations. For large datasets, it is common to constrain the maximum depth of trees, as done in CSQBF, to reduce the time complexity of tree-based models. This reduces the RF complexity to $\mathcal{O}(N)$. Furthermore, one can alternatively use CLARA [71], which draws multiple samples from the data set, and applies PAM to each of them. The complexity of CLARA is reduced to $\mathcal{O}(KS^2 + K(N-K))$, where S is the sample size at each iteration. CLARA makes K-medoids clustering applicable to large data sets at the cost of cluster quality. Therefore, the total time complexity of CSQBF would be $\mathcal{O}(N) + \mathbb{O}(KS^2 + K(N-K)) + N_{iter} * \mathcal{O}(N)$. In general, when $N \gg K, S, N_{iter}$ complexity of CSQBF can be reduced to $\mathcal{O}(N)$.

3.4.3 Sensitivity Analysis

We examined the influence of the model parameters λ and α on the algorithm's performance. A larger λ decreases the weight on the (unlabeled) cluster size, $|C_k|$, in Eq. (3.5) and, thereby, increase the importance of cluster uncertainty. Values of 1/4, 1, and 4 were considered (other values were tested and results were similar). Parameter α controls the portion of the unlabeled instances which are considered in query instance selections. Values of 1/3, 2/3, and 1 were considered (we tested other values and obtained similar results). Results for *Banknote*, *Ibn_sina* and *NvsM* datasets based on 15 random replicates are provided here and Fig. 3.4 illustrates the results (other datasets showed similar patterns). We also compared CSQBF with SQBF both having $\alpha = 2/3$ since this was the value used in the winning SQBF



(a) *Banknote* (α) (b) *Ibn_Sina* (α) (c) *Letter_NvsM* (α)

(d) *Banknote* (λ) (e) *Ibn_Sina* (λ) (f) *Letter_NvsM* (λ)

Figure 3.4: Sensitivity Analysis Based on AUC Results With $\lambda = 1$, $\frac{1}{4}$ and 4 on the Top, $\alpha = \frac{2}{3}$, $\frac{1}{3}$ and 1 on the Bottom.

algorithm during the Active Learning Challenge [8]. Comparisons for $\alpha = 2/3$ are not provided here for the sake of conciseness. However, CSQBF performed similarly better than SQBF. Note that iteration 1 results do not depend on the values of λ and α because the algorithm initially selects the cluster medoids before computing instance and cluster utilities.

The graphs on top of Fig. 3.4 use λ values of 1/4, 1, and 4, while α is set to 1/3. F1-score degrades for $\lambda = 4$, especially for the *Banknote* and *Letter NvsM* datasets in Fig. 3.4d and Fig. 3.4f respectively. This extreme value for λ illustrates that cluster utility is affected by relative weights of uncertainty and size factor. Otherwise, the results are not overly sensitive to λ values of 1/4 or 1. Therefore, the equally weighted case with $\lambda = 1$ is used to explore the effects of α. The graphs on the bottom of Fig. 3.4 use α values of 1/3, 2/3, and 1. From these figures, results degrade for $\alpha = 1$ and $\frac{2}{3}$. Therefore, we chose $\alpha = \frac{1}{3}$ for all the datasets.

3.5 Conclusion

In this work, a new active batch learning algorithm, the Cluster-based Stochastic Query-By-Forest, is proposed. CSQBF takes advantage of the stochastic procedure used by SQBF [8], which is recognized for its good performance and improves upon it by utilizing the unsupervised learning information throughout the learning process. Pre-clustering enables CSQBF to further improve the learning process by exploring the feature space and querying representative instances with a small extra cost. Employing cluster utility reinforces diversity and density in the sampling procedure. Moreover, CSQBF uses a robust measure for uncertainty compared to SQBF which results in a more stable performance throughout the learning.

Chapter4

A DUAL MODEL AGNOSTIC STRATEGY TO EXPLORE REPRESENTATIVENESS AND INFORMATIVENESS IN ACTIVE LEARNING

4.1 Abstract

One of the learning obstacles in today's information revolution is the large amount of annotated data required for statistical learning. Active learning techniques have shown promising results in accelerating the learning procedures while reducing the cost of data annotation. Uncertainty and density-based active learning strategies have been widely adopted separately and simultaneously for various application domains. Most of the work to understand the high dimensional features spaces to be incorporated as a density component is based on finding a set of instance clusters. However, our understanding of high-dimensional space is not reliable due to the ambiguity of distance in these spaces. In this paper, we propose two novel probabilistic Query-By-Committee (QBC) algorithms that integrate classification and clustering uncertainty into a unified measure for active learning. Although formulated differently, the novelty of the proposed strategies aside from counting for both classification uncertainty and input space inherent ambiguity is taking advantage of the relation-

ship between data clusters and label structure. Our methods are naturally capable of handling multi-class problems without using one-vs-all approaches. Test results on several real-world datasets of different features dimension sizes and class ratios show how combining class and cluster uncertainties can achieve significantly better performances.

4.2 Introduction

A learning algorithm receives a set of pre-labeled training data in a typical classification problem, and it does not contribute to the time-consuming and labor-intensive task of data collection and annotation. Hence, training a model capable of dealing with limited annotated data is crucial. Active learning has been a promising strategy for when obtaining data is easy and cheap, while annotation is expensive [122]. The main idea in active learning is that not all the data points are of equal value to a classifier, so it is imperative to prioritize instances with more potential benefits to the learner for the labor-intensive and tedious manual annotation. Instead of being a passive recipient of the data, the active learner determines the most beneficial instances for classification and queries their label from the oracle.

Active learning (AL) has received a great amount of attention in different application domains such as object recognition, recommender systems, video and image classification and etiology [39, 66, 92, 132]. These approaches rely on different heuristics and can be categorized roughly into two main groups. *Uncertainty-based* approaches evaluate instances based on their ability to decrease a model's uncertainty in classification. The intuition behind these foremost used techniques is that the classifier should be reasonably confident about its predictions for those instances that are relatively far from the decision boundary. Therefore, labeling less certain ones can potentially offer more information. Informativeness in these approaches is modeled

using different strategies such as expected error reduction [115], variance reduction [118] and query-by-committee (QBC) [125]. Although they are the most common methods in the literature, uncertainty-based strategies are prone to selecting outliers, especially at early stages when the model is not strong enough. Introducing a density element to the query process can discourage the selection of outliers. *Clustering-based* approaches, on the other hand, focus more on exploiting input data distribution either explicitly by using clustering techniques [154] or implicitly through variance reduction methods. Many studies have developed balanced approaches combining the two criteria of representativeness and informativeness either directly by selecting the most representative instances among the most uncertain ones [87] or using density-weighted approaches [100, 159]. Also, various studies have introduced a diversity element to avoid sampling instances similar to the already labeled ones [50, 74, 153]. Work by [74] investigates the effectiveness of density and diversity components based on several distance metrics. Few studies in the literature have focused on using non-parametric methods for multi-class problems while exploring informativeness and representativeness of data instances. Even fewer researches like [157] have taken the uncertainty associated with our understanding of perplex high-dimensional feature space into account.

In this paper, we propose two active learning approaches trying to achieve a good classification performance while selecting the most useful data instances from early on. The first main idea is to take the dependency between class distribution and input data structure into account by transforming the problem to a label powerset framework in multi-label classification. The idea is ruled by the well-known cluster assumption in semi-supervised learning proposed by [113] which states that data points tend to form clusters that are more likely to have matching labels. The second proposed algorithm indirectly implements this idea by proposing a coherent and con-

sistent measure that unifies uncertainty and density elements for label querying. The key idea behind this algorithm is to formulate measures for uncertainty and density governed by same principles. The primary contribution of this work is threefold: i) our active learning criteria in both proposed algorithms not only exploit classification uncertainty and the underlying data structure synchronously, but also they consider the reliability of clustering. ii) our algorithms inherently handle both binary and multi-class problems as well as data with mixed features, iii) they are computationally efficient, allowing applicability across sizable datasets.

The rest of this article is organized as follows. Section 4.3 provides background on different active learning scenarios and justifies our query strategy design. Section 4.4 discusses the proposed two active learning algorithms. Section 4.5.1 provides details of the experiments, and results and finally Section 4.6 concludes the study.

4.3 Background

4.3.1 Active Learning

Most early studies on AL were primarily focused on formulating instance informativeness, which measures the ability of an instance in reducing the uncertainty of a statistical model. The main drawback of these approaches is twofold; First, at early iterations, the instance selection scores are determined by a classifier trained only on a small number of labeled instances. Thus, they run the risk of selecting outliers and the labeling is susceptible to sampling bias. Second, the unsupervised knowledge of the pool of unlabeled data, which can potentially boost the performance of supervised learning, is ignored [62]. Incorporating a measure for representativeness of the underlying data structure can help to address this problem.

There has been an extensive amount of work on incorporating informativeness and

density into the label query process. Variance reduction approaches tacitly consider the prior data distribution, but they usually come at high computational expenses [123]. Studies such as [36, 57, 62, 65, 100] formulate density explicitly and benefit from informativeness and density synchronously. A typical way to frame density is to employ a specific clustering algorithm and evaluate the informativeness of instances based on their similarity to cluster centroids. Although in many applications, such as work by [21], cluster assumption (meaning data clusters have homogeneous labels) [113] is a fair premise, the performance of these combined approaches is dependent to the clustering quality. In other words, it is not determined how these approaches perform under insufficiently accurate clustering. Therefore, incorporating clustering uncertainty can potentially improve the sampling process.

4.3.2 Multi-Class Scenario

Recent work in the area of active learning rely mostly on a wide range of heuristics that characterize instance informativeness based on a variety of criteria such as prediction of variance directly [156] or indirectly using the disagreement among a committee of classifiers [31, 94], version space of SVMs [22, 70] and expected informativeness [57, 65]. Most of these methods use SVMs and Gaussian Process classifiers, so they are intrinsically applicable to binary class problems. Although most of the literature has adapted these heuristics to multi-class problems by using pairwise comparisons between classes or one-vs-all strategies, this extension is not straightforward and can be faulty due to the presence of multiple hyperplanes [157]. Another group of the existing multi-class approaches deal with these problems directly [89, 157] by using graph-based methods which are independent of the number of classes, but can be computationally expensive for sizable datasets.

4.3.3 Multi-Label Classification

Unlike in multi-class problems where a single class label is assigned to each data point, the goal in a multi-label classification problem is to learn a model that can predict a set of labels associated with each data instance [145]. Many existing work in this area can be categorized into two main groups: *problem transformation* and *algorithm adaptation* approaches [111, 141]. In the problem transformation methods, the multi-label problem is converted to one or several single-label problems so that common classification algorithms can be applied, whereas in algorithm adaptation approaches, well-known single-label methods are extended to adapt to multi-label scenarios. Algorithm adaptation methods are inherently a problem transformation, and most of their literature is focused on modifications of decision trees and AdaBoost [117]. Problem transformation approaches can be further categorized into three groups of Binary Methods (BM), Ranking Methods (RM), and Combination Methods (CM) [111]. BM methods learn binary models for each label to be relevant independently from the rest of the labels [88], whereas RM methods come up with a probability distribution over all labels and the final label set is chosen based on a predefined threshold. The CM methods, however, transform the problem into a single-label one by creating an atomic label based on the label set for each data point [111]. The main drawback of BM and RM methods is that they assume the labels to be disjoint. In contrast, multi-labeled data, can consist of highly correlated classes [47]. CM methods tackle this issue in a sense. Although creating a class imbalance problem can be an issue in CM approaches, studies such as [111, 114] have tried to overcome this problem by using an ensemble of CM classifiers trained on a subset of randomly selected labels from all labels and ensemble of pruned sets respectively. We use the idea behind combining the labels in one of our proposed algorithms (Sec-

tion 4.4.1). However, applying CM approaches proposed in [111, 114] directly is not beneficial for our algorithms, since we are not solving a multi-label problem, but rather creating a pseudo-label that accounts for density element of our AL tactics.

4.4 Methodology

In this work, we consider the cold-start AL, i.e., we assume that no labeled data is available initially. One approach to start the label query procedure is to initialize the labeled set $\mathbb{L}$ by labeling the most representative instances from the unlabeled pool $\mathbb{U}$. To do so, Random Forest Clustering (RFC) which consists of unsupervised RF followed by the Partitioning Around Medoids(PAM) algorithm is applied to the unlabeled set $\mathbb{U}$ to generate K initial clusters, where K is determined a priori and all instances are assigned a cluster label. The outcome of this step is cluster medoids, which are queried for labeling to form the initial labeled set. The effectiveness of this approach is shown in several studies [30, 131]

4.4.1 Double Margin Active Learning (DMAL)

Initialization

Given the labeled and unlabeled data sets at iteration t denoted by $\mathbb{L}^{(t)}$ and $\mathbb{U}^{(t)}$ respectively, the initial labeled set, $\mathbb{L}^{(0)}$, consists of K cluster medoids (x_1, c_1), (x_2, c_2), $\ldots (x_K, c_K)$ from the previous step where c_i is the label for instance x_i. In a single-label classification setting, each instance x is assigned a single label c from a previously known finite set of labels $\mathbb{C}$. A single-label dataset $\mathbb{D}$ is then simply constructed from n instances $\{(x_1, c_1), (x_2, c_2), \ldots (x_n, c_n)\}$. We frame a multi-label classification task where two labels are assigned to each instance: a class label $c \in \mathbb{C}$ and a cluster label $k \in \mathbb{K}$ where $\mathbb{C}$ and $\mathbb{K}$ are finite sets of class labels and clusters respectively. We combine the two labels and create a single-label problem simply by

treating the two labels, c_i and k_i, as an atomic label l_i. Therefore, the target feature set is transformed to $L = \{l_1, l_2, \ldots, l_n\}$ where $l_i = c_i k_i$. Thus, $\mathbb{L}^{(0)}$ would consists of K cluster medoids (x_1, l_1), (x_2, l_2), ... (x_k, l_k) where l_i is the new label.

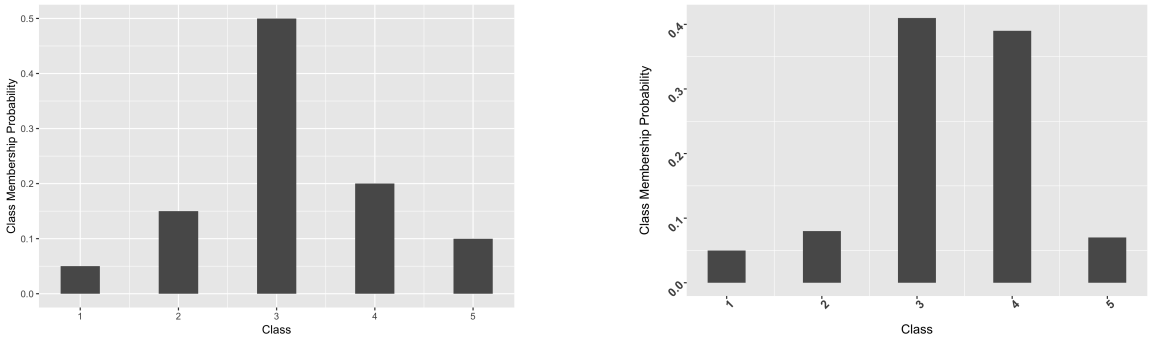
Main Step

After the initial step, the algorithm is run iteratively for a fixed number of steps or until a threshold is met. At each iteration t, using the available labeled data $\mathbb{L}^{(t)}$, a supervised RF model $\Phi(\mathbb{L}^{(t)}|\Theta)$ is built where Θ is the parameter set for Φ. Class membership probability estimates for each instance is provided by Φ and is calculated as the proportion of trees that vote for each class. For each $x_u \in \mathbb{U}$ we denote these probabilities by $P(c = j|\Phi, x_u)$ where $j \in \mathbb{C}$. Works by [50, 66] provide illustrative examples on how treating instance with the highest value for $P(.|\Phi, x_u)$ as the most certain instance can be misleading to the model. The intuition behind why this metric is not always proper is that only the pick point of $P(.|\Phi, x_u)$ is considered instead of its distribution. Fig. 4.1 shows the distribution of the class membership probabilities for two data instances. If a prediction is meant to be made, an RF model would assign both of these instances to class 3 since it is the most likely one. However, the model is much more confident for the case in Fig. 4.1a since the instance is much more likely to be associated with class 3 compared to all the other classes whereas for the case in Fig. 4.1b both classes 3 and 4 are almost equally likely.

Therefore, as a substitute for the majority of votes to characterize uncertainty, we define margin as the difference between the first and second highest class membership probabilities, i.e. for each $x_u \in \mathbb{U}$

$$M(x_u) = \underset{y}{Max}\{P(y|\Phi, x_u)\} - \underset{y}{SecMax}\{P(y|\Phi, x_u)\}, \tag{4.1}$$

where $y \in \mathcal{Y}$ is the target feature used by model Φ and $SecMax(x)$ function obtains



(a) (b)

Figure 4.1: Class Probabilities of Two Data Instances: X and Y Axis Indicate the Class Label and Class Membership Probabilities Respectively.

the second greatest value from the set of class probabilities. Next, for each $x_u \in \mathbb{U}$ two separate margins are calculated based on $M(x_u)$. First, margin of class, $M_c(x_u)$ which denotes how confident model $\Phi(.)$ is in class label for the unlabeled instance x_u and it is defined as

$$\begin{aligned} M_c(x_u) &= \underset{c}{Max}\{\sum P(l|\Phi, x_u)\} - \underset{c}{SecMax}\{\sum P(l|\Phi, x_u)\} \\ &= \underset{c}{Max}\{\sum_k P(c, k|\Phi, x_u)\} - \underset{c}{SecMax}\{\sum_k P(c, k|\Phi, x_u)\} \end{aligned} \tag{4.2}$$

Subscript l in Eq. (4.2) indicates that margin is calculated based on the class probabilities provided by model $\Phi(.)$ by using the new label set L instead of the original class labels as the target feature. Second, we define margin of cluster, $M_k(x_u)$, to account for model certainty in assigning cluster labels as follows:

$$\begin{aligned} M_k(x_u) &= \underset{k}{Max}\{\sum P(l|\Phi, x_u)\} - \underset{k}{SecMax}\{P(l|\Phi, x_u)\} \\ &= \underset{k}{Max}\{\sum_c P(c, k|\Phi, x_u)\} - \underset{k}{SecMax}\{\sum_c P(c, k|\Phi, x_u)\} \end{aligned} \tag{4.3}$$

Intuitively, the lower the M_c for an unlabeled instance, the lower is the model certainty in labeling that instance, making that instance a more informative candidate for the model. On the other hand, instances with higher values of M_k are better representatives of their clusters and are less likely to be outliers. Therefore, we use these

two margins to define an uncertainty score for each unlabeled instance $x_u \in \mathbb{U}$. This score accounts for density, informativeness and representativeness simultaneously and is defined as:

$$U(x) = \beta\Big(M_c(x)\Big) + (1-\beta)\Big(1 - M_k(x)\Big) \tag{4.4}$$

where β control the relative importance of the two margins. Now, when it comes to selecting the query batch for the next iteration, a very natural way is to select instances with the lowest value of $U(x)$. However, in a batch-mode AL setting selecting the most uncertain instances can cause selection of outliers and/or redundant instances. It has been showed by various studies that using uncertainty alone can obtain mediocre results and how adding an exploration element can boost the performance [52]. Thus, we employ two tactics similar to [8, 16] to make a trade-off between exploitation and exploration. First, we introduce parameter α as a second filter for outliers and redundant samples. The top α proportion of the unlabeled instances ordered by their $U(x)$ value are the ones that form the label query candidate set $\mathbb{Q}$, i.e. $\mathbb{Q} = \{x_u^{(1)}, x_u^{(2)}, \ldots, x_u^{(\alpha)}, \}$ where $x_u^{(i)} \in \mathbb{U}$ is the unlabeled instance that corresponds to the i^{th} smallest value of $U(x)$ (Eq. (4.4)). Although, this is considered as a less strategic approach to impose diversity, it is computationally efficient in the case of relatively big feature spaces. Second, in order to diversify our search in the feature space, we employ a weighted random sampling approach based on the normalized values of uncertainty scores to obtain query sampling probabilities, i.e. for each $x \in \mathbb{Q}$

$$U^{norm}(x) = \frac{U(x_u^{(\alpha)}) - U(x)}{U(x_u^{(\alpha)}) - U(x_u^{(1)})} \tag{4.5}$$

where $U(x)$ is the uncertainty score associated with the query candidate x and $U(x_u^{(\alpha)})$ and $U(x_u^{(1)})$ correspond to the highest and lowest values of $U(x)$ for instances in $\mathbb{Q}$

respectively. Sampling probabilities are then calculated as

$$Pr(x) = \frac{U^{norm}(x)}{\sum_{x \in \mathbb{Q}} U^{norm}(x)} \tag{4.6}$$

Next, labels are queried for a batch of size $q < |\mathbb{Q}|$ for the randomly selected candidates based on $Pr(x)$ probabilities. Newly labeled instances are then added to the labeled set $\mathbb{L}$ and excluded from $\mathbb{U}$. Model Φ is retrained, evaluated using test data and again, all the steps above (Equations (4.1)–(4.6)) are repeated iteratively until all instances are labeled (i.e. $\mathbb{U} = \varnothing$), or a stopping criterion is met. Algorithm 2 summarizes the DMAL algorithm.

Algorithm 2 Double Margin Active Learning (DMAL)

$\mathcal{D}(x) = \{dist(x_i, x_j) | x_i, x_j \in \mathcal{U}\}$ ▷ Obtain RF dissimilarity matrix

$l = 1$ ▷ Initialization

PAM $\leftarrow \{\mathcal{D}(X), K\}$ ▷ Implement PAM to partition the data into K clusters

$\mathbb{L}^{(0)} =$ **QueryClusterMedoids**

for t: 0 to T **do**

  $\forall x \in \mathbb{L}^{(t)} : l'_i = l_i, c_i$ ▷ Create new labels based on class and cluster labels

  Construct $\Phi^{(t)}(\mathbb{L}^{(t)}|\Theta)$ ▷ Build RF classifier

  Evaluate model $\Phi^{(t)}$ on the test set ▷ Calculate F1 score

  for $x_u \in \mathbb{U}$ **do**

    Compute $M_c(x_u)$ ▷ Class uncertainty (Eq. (4.2))

    Compute $M_k(x_u)$ ▷ Cluster uncertainty (Eq. (4.3))

    Compute $U(x)$ ▷ Compute instance uncertainty score (Eq. (4.4))

  end for

  QueryLabels$(x \in \mathbb{Q}|Pr(x))$ ▷ (Eq. (4.6))

  $\mathbb{L}^{(t)} \rightarrow \mathbb{L}^{(t+1)}$ & $\mathbb{U}^{(t)} \rightarrow \mathbb{U}^{(t+1)}$ ▷ Adjust labeled and unlabeled sets

end for

4.4.2 Cluster Agnostic Active Learning (CAAL)

Initialization

Similar to DMAL initialization step, the data is clustered into K groups and the initial labeled set $\mathbb{L}^{(0)}$ is formed based on cluster medoids. The next step is to calculate cluster margins. In order to do so, the problem is transformed to a supervised learning model where the data features are used to predict the cluster label obtained earlier. The cluster membership probabilities are then simply estimated by making a prediction for unlabeled instances. We denote these probabilities by $P(k = j|\Psi, x_u)$ for each $x_u \in \mathbb{U}$ where $j \in \{1, 2, \dots K\}$.

Main Step

After the initial step the algorithm is run iteratively for a fixed number of steps or until a threshold is met. Following [8], we increase the batch size exponentially until the whole unlabeled set is labeled. Therefore, our main focus here is to accelerate learning which is equivalent to querying fewer labels. At each iteration t, using the available labeled data $\mathbb{L}^{(t)}$, a supervised RF model, Φ, is built consisting of T trees. Class membership probability estimates for each instance is provided by RF and is calculated as the proportion of trees that vote for each class. For each $x_u \in \mathbb{U}$ we denote these probabilities by $P(c = j|\Phi, x_u)$ for each $x_u \in \mathbb{U}$ where $j \in \mathbb{C}$. Margins of class and clustering are then calculated by

$$M_c(x_u) = \underset{c}{Max}\{P(c|\Phi, x_u)\} - \underset{y}{SecMax}\{P(c|\Phi, x_u)\} \tag{4.7}$$

$$M_k(x_u) = \underset{k}{Max}\{P(k|\Psi, x_u)\} - \underset{y}{SecMax}\{P(k|\Psi, x_u)\} \tag{4.8}$$

Next, similar to DMAL the utility scores are calculated for instances of $\mathbb{U}$, the candidate set $\mathbb{Q}$ is formed and labels are queried for the selected instances based on

selection probabilities.

Algorithm 3 Cluster Agnostic Active Learning (CAAL)

1: $\mathcal{D}(x) = \{dist(x_i, x_j) | x_i, x_j \in \mathcal{U}\}$ ▷ Obtain RF dissimilarity matrix
2: $l = 1$ ▷ Initialization
3: PAM $\leftarrow \{\mathcal{D}(X), K\}$ ▷ Partition the data into K clusters
4: Construct $\Psi(\mathbb{L} \cup \mathbb{U}|\Theta')$ and $M_k(x_u) \forall x \in \mathbb{U}$
5: $\mathbb{L}^{(0)} =$ **QueryClusterMedoids**
6: **for** t: 0 to T **do**
7: Construct $\Phi^{(t)}(\mathbb{L}^{(t)}|\Theta)$ ▷ Build RF classifier
8: Evaluate model $\Phi^{(t)}$ on the test set
9: **for** each $x_u \in \mathbb{U}$ **do**
10: Compute $M_c(x_u)$ ▷ Compute Class uncertainty (Eq. (4.7))
11: Compute $M_k(x_u)$ ▷ Cluster uncertainty (Eq. (4.8))
12: Compute $U(x)$ ▷ Compute instance uncertainty score (Eq. (4.4))
13: **end for**
14: Construct candidate set $\mathbb{Q}$
15: **QueryLabels**$(x \in \mathbb{Q}|Pr(x))$ ▷ (Eq. (4.6))
16: $\mathbb{L}^{(t)} \rightarrow \mathbb{L}^{(t+1)}$ & $\mathbb{U}^{(t)} \rightarrow \mathbb{U}^{(t+1)}$ ▷ Adjust labeled and unlabeled sets
17: **end for**

4.5 Experiments and Results

In this section, we investigate the effectiveness of the CAAL algorithm on several binary and multi-class datasets of different sizes. We also compare the performance of CAAL and DMAL with the following baseline approaches:

- ***DMAL*** Double Margin Active Learning described in Section 4.4.1

- ***CSQBF*** Cluster-based Stochastic Query by Forest (CSQBF) algorithm described in Chapter 3

- ***MUDD_IMP*** Improved version of Maximizing Uncertainty, Density and Diversity by [50]

Similar to the proposed methods, CSQBF characterizes uncertainty based on classification margins, and ensures sampling from all clusters. The main point of difference between CAAL and CSQBF is that CAAL does not adhere rigidly to the clusters, yet it still explores the features space. For fair comparisons, we modified the MUDD approach to provide it with the same initial labeled set consisting of cluster medoids. We call this improved version MUDD_IMP.

4.5.1 Experimental Settings

Following [8], we assume that the batch sizes increase exponentially over 15 iterations until all the labels are queried. This is equivalent to assuming that labeling is more expensive at the beginning. Alternatively, a fixed batch size or budget can be considered similar to works by [61, 128]. We chose Macro-Averaged F1-score as the measure of comparison. The F1-score is the harmonic mean of precision and recall [133] and is suitable substitute for AUC in multi-class AL problems [110, 128]. In macro-averaging, first for each class the F1-score is computed against all the others and then the average over all classes is taken, i.e., if ρ_i and π_i are precision and recall for each class i then:

$$F_i = \frac{2\pi i \rho i}{\rho_i + \pi_i}, \qquad \text{F1-score } = \frac{\sum_{i=1}^{C} F_i}{C} \tag{4.9}$$

where $C = |\mathbb{C}|$ is total number of classes. Since F1-score gives equal weight to all classes, it is more influenced by the classifier's performance on rare categories

[103]. Therefore, it does not have the limitation of accuracy for imbalanced datasets. Regardless, we provide the results based on accuracy measure as well. We split the data randomly into two halves for those datasets that do not have a separate train and test splits. RF classifier consisting of 700 fully grown trees was used, and to account for class imbalance, we use stratified sampling in the RF bagging process. Since the DMAL algorithm creates a new set of labels, L, it would initially need at least one instance from each new class. In other words, from each cluster, at least one instance from the original class categories should exist in the initial labeled set $\mathbb{L}^{(0)}$. Although, this is not a requirement for CAAL and CSQBF algorithms, the same $\mathbb{L}^{(0)}$ is provided to them for fair comparisons. Our experiments show that the initial labeled set plays a great role in AL algorithms performance. Therefore, for the sake of comparisons, it is critical to use the same $\mathbb{L}^{(0)}$ for all algorithms.

Several public real-world datasets (summarized in Table 4.1) were used to compare the performance of CAAL against the aforementioned algorithms. Ibn Sina is one of the dataset used in AISTATS 2010 Active Learning Challenge [53]. MNIST35 is a subset of the well-known MNIST handwritten digit data [79] which includes only images for numbers 3 and 5. Twitter dataset, is annotated Twitter data by cyberbullying experts in work by [151]. In order to show the severeness of cyberbullying, they annotated a sample of 990 tweets from the publicly available 2011 TREC Microblog track corpus and roughly estimated that 50,000 English bullying traces are produced daily. Since following bullying traces among millions of tweets is a hard (and probably impossible) task, they created and enriched dataset which is obtained by collecting tweets that contain at least one of the "bully", "bullied", and "bullying" keywords from the public Twitter streaming API. They further removed re-tweets by excluding those that include the acronym "RT". They also note that the enrichment process is meant to retain many first-hand bullying traces at the cost of selection

bias. The same annotators who labeled the TREC corpus labeled 1762 tweets sampled uniformly from the enriched dataset. Among them, 684 (39%) were labeled as bullying traces.

Table 4.1: Test Datasets

Dataset	Train	Test	Features	Classes	Min %
Banknote	686	686	4	2	23.52 %
Car	864	864	62	2	44.46%
Coil2000	4911	4911	85	2	5.97 %
EEG	7490	7490	14	2	44.88%
Ibn Sina	10361	10361	92	2	37.84%
Letter	10000	10000	16	26	3.67%
Letter NvsM	788	787	16	2	49.71%
Letter OQG	1155	1154	16	3	32.61%
Letter TIL	1156	1156	16	3	32.61%
Letter VvsY	788	787	16	2	49.71%
Mamo	415	415	5	4	3.76 %
MNIST35	5776	5776	784	2	46.93%
Mushroom	4062	4062	22	2	48.20%
Nursery	6479	6479	8	2	48.55%
Occupancy	10280	10280	5	2	2.53 %
OptDigits	3823	1797	64	2	9.86%
Pen	7494	3498	16	10	9.6%
Segmentation	1155	1155	19	7	14.29%
Spambase	2300	2301	57	2	39.40%
Transfusion	374	374	4	2	23.80%
Twitter	490	123	292	2	44.7%
USPS	7291	2007	256	10	7.61%
Vehicle	423	423	18	4	32.65 %

As per terms of services of Twitter, the original labeled tweets are not included

in the dataset used by [151] which is publicly available at [1] so, the original tweets were crawled through Twitter API (using "tweepy" python package) by the tweet ids that are available online at [1]. By the time we crawled the data, some of the tweets were no longer available probably due to deletion by their owners (obtained only 1092 out of 1762 original labeled tweets). After the non-English tweets were discarded (resulting in 613 tweets), the tweets were case-folded, English stopwords and excessive punctuations were removed, and any user mentions preceded by a "@" were replaced by the anonymized user name @USER. Any URLs starting with "http" were replaced by the token "URL". Hashtags were also replaced by "#HASHTAG". Moreover, based on our experiments, all the single-character words were removed since they may not carry any meaning and are used differently by users. Some of such words are used as a substitute for English stopwords such as "and" that appears as "n" or "the" that shows up as "d".

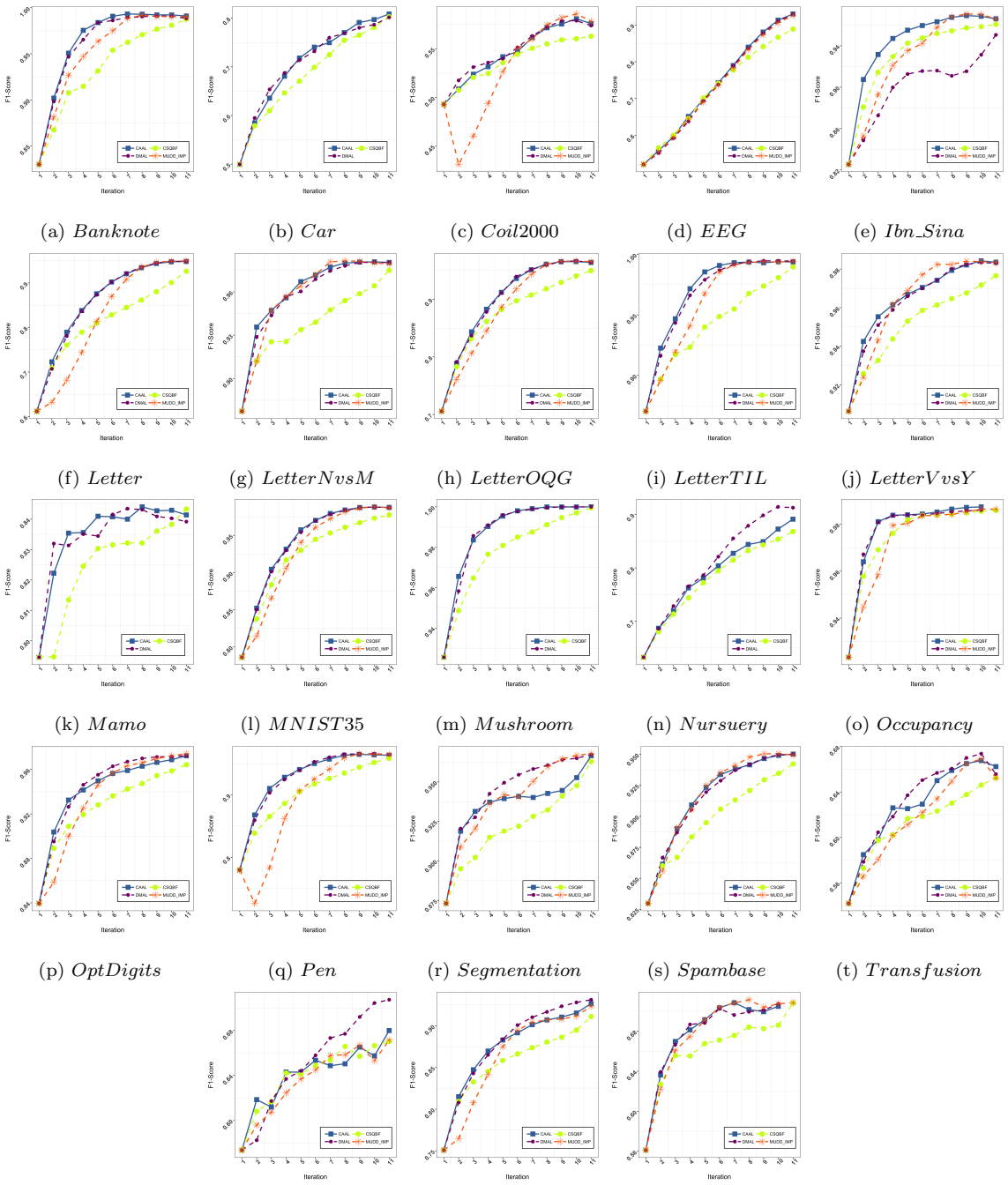
We used unigrams and bigrams (which would automatically include the number of hashtags as well as referred users and websites since we treated them as tokens), number of words, and length of the tweet as the textual features. This process created over 6000 features. We used the frequency threshold of 5 to choose the final set of bigrams and unigrams and discarded those tokens that did not have enough occurrences among all the tweets. This reduced the number of features to 291. The rest of the test datasets were obtained from the UCI Machine Learning Repository [33]. Note that LetterOQG, LetterTIL, Letter NvsM and Letter VvsY are subsets of the Letter dataset that include the corresponding letters only. Figure 4.2 shows the F1-scores learning curves over 11 iterations. Each point on the curves represents the average of 15 F1-scores, each from one replicate. Since MUDD_IMP uses Euclidean distance to calculate diversity and density, it cannot be applied to the datasets with categorical features. Therefore, for those datasets, we only provided the comparison

of CAAL, DMAL and CSQBF algorithms. As Fig. 4.2 shows CAAL and DMAL outperform CSQBF and MUDD_IMP for most cases.

In terms of consistency in performance, CAAL and DMAL show very competitive performances with CSQBF, which had lower variances than SQBF for most of the datasets. Figure 4.3 illustrates the distribution of F1-scores throughout the iterations. Each box is based on 15 replicates. It is clear that CAAL and DMAL have relatively low variances from the early iterations.

We also calculated the relative Area under the Learning Curves (ALC) as used in the AL challenge [53] for comparisons which specifies the percentage of ALC described by that method compared to the best learning curve possible. ALC values were calculated based on a $\log 2$ scaling for the x-axis (number of labeled instances) to favor good performances at early iterations [53]. Tukey's pair-wise test at 95% level of significance was conducted to compare the average ALC values of CAAL and DMAL against CSQBF and MUDD_IMP. As Table 4.2 shows, CAAL and DMAL produce significantly better results in terms of average ALC and average rank compared to CSQBF and MUDD_IMP. CAAL achieved the best average results with 0.8423 average ALC and 1.5652 average rank. The ALC values for MUDD_IMP have left empty in case of data with categorical features.

It is worth mentioning that one of the main drawback of CM-based methods in multi-label classification problems is that they might create a large number of distinct label combinations while each combination is only associated with a few instances, i.e. the can create a class-imbalanced problem. Although, DMAL resembles a CM approach, the number of possible labels does not scale exponentially with the number of class labels because of two reasons. First, unlike a typical CM problem, during DMAL only two labels are assigned to each instance: class and cluster. Second, in many semi-supervised studies it is a fair assumption that the number of clusters is



(a) *Banknote* (b) *Car* (c) *Coil*2000 (d) *EEG* (e) *Ibn_Sina*

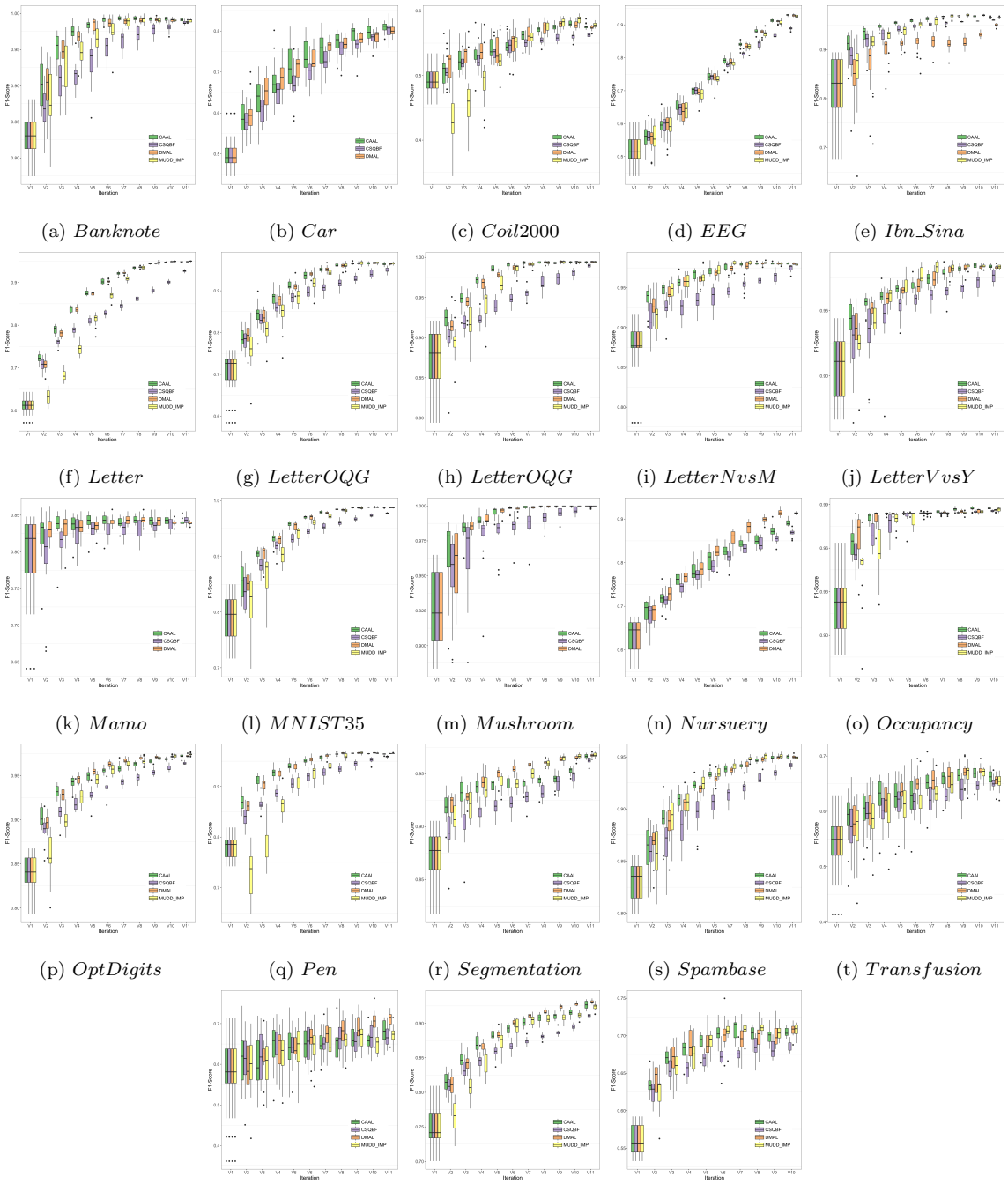
(f) *Letter* (g) *LetterNvsM* (h) *LetterOQG* (i) *LetterTIL* (j) *LetterVvsY*

(k) *Mamo* (l) *MNIST*35 (m) *Mushroom* (n) *Nursuery* (o) *Occupancy*

(p) *OptDigits* (q) *Pen* (r) *Segmentation* (s) *Spambase* (t) *Transfusion*

(u) *Twitter* (v) *USPS* (w) *Vehicle*

Figure 4.2: F1-Score Learning Curves for CAAL($\beta = 1/2$), DMAL, CSQBF and MUDD_IMP. Color Legend: CAAL (Solid Blue), DMAL (Dashed Purple), CSQBF (Dashed Yellow) and MUDD_IMP (Dashed Orange). See the Color Version for the Best View.



(a) *Banknote* (b) *Car* (c) *Coil*2000 (d) *EEG* (e) *Ibn_Sina*

(f) *Letter* (g) *LetterOQG* (h) *LetterOQG* (i) *LetterNvsM* (j) *LetterVvsY*

(k) *Mamo* (l) *MNIST*35 (m) *Mushroom* (n) *Nursuery* (o) *Occupancy*

(p) *OptDigits* (q) *Pen* (r) *Segmentation* (s) *Spambase* (t) *Transfusion*

(u) *Twitter* (v) *USPS* (w) *Vehicle*

Figure 4.3: Distributions of F1 Scores for CAAL, DMAL, CSQBF and MUDD_IMP Algorithms. ($\beta = 1/2$ for Both CAAL and DMAL). Color Legend: CAAL (Green), DMAL (Orange), CSQBF (Purple) and MUDD_IMP (Yellow). See Color Version for the Best View.

not extremely high.

In order to understand the effect of β parameters, we further calculated the ALC for both CAAL and DMAL algorithms under multiple settings of β. Since $M_c(x), M_k(x) \leq 1$, we chose β and $1-\beta$ to control the relative importance of uncertainty and density. Table 4.3 and Table 4.4 illustrate the average ALC for several values of β. For each datasets, numbers in parentheses are the ranking of each β among all the β values for that datasets. Based on Table 4.3, although $\beta = 1/3$ has the best average rank followed by $\beta = 2/3$ and $\beta = 1$, the best average ALC is achieved by $\beta = 1/3$. Results for the DMAL presented in Table 4.4 indicate that $\beta = 1/2$ and $\beta = 1$ have the best average ranks, but $\beta = 1/2$ wins in terms of average ALC. One-factor ANOVA and Tukey's tests at a 95% significance level on ALC values showed that only ALC values corresponding to $\beta = 0$ are significantly lower than the other ones.

4.6 Conclusion

Uncertainty and density-based approaches have been widely adopted individually and simultaneously for active learning problems. However, understanding the high dimensional input data structure is associated with uncertainty, and only a few studies have developed efficient non-parametric methods to incorporate this uncertainty into label query process explicitly. In this study, we proposed two probabilistic query-by-committee frameworks that take both classification and clustering uncertainty into account at the same time that they benefit from the efficiency of random forest. Moreover, our methodologies are naturally capable of handling multi-class problems without using one-vs-all strategies that are used in other active learning approaches. We tested the proposed algorithms on several real-world datasets of different features dimension sizes and class ratios. Our test results on both algorithms show that

considering the clustering uncertainty can improve the learning process significantly.

Table 4.2: ALC for Average F1-Score Curves by CAAL, DMAL, CSQBF, and MUDD_IMP Algorithms. Numbers in Parentheses Are the Rankings Based on the Other ALC Scores for the Corresponding Dataset.

Data	**CAAL**	**CSQBF**	**DMAL**	**MUDD_IMP**
Banknote	0.9518 (1)	0.9199 (4)	0.9483 (2)	0.9381 (3)
Car	0.6741 (2)	0.6533 (3)	0.677 (1)	-
Coil2000	0.5425 (2)	0.5343 (3)	0.5455 (1)	0.5193 (4)
EEG	0.7177 (1)	0.7063 (4)	0.7129 (2)	0.7124 (3)
Ibn Sina	0.9413 (1)	0.927 (2)	0.8955 (4)	0.9226 (3)
Letter	0.8305 (1)	0.7877 (3)	0.8262 (2)	0.7818 (4)
Letter NvsM	0.9542 (1)	0.931 (4)	0.9513 (2)	0.9494 (3)
Letter OQG	0.8764 (1)	0.8545 (4)	0.8761 (2)	0.8585 (3)
Letter TIL	0.9602 (1)	0.9318 (4)	0.957 (2)	0.9464 (3)
Letter VvsY	0.9596 (1)	0.9454 (4)	0.9576 (2)	0.9561 (3)
Mamo	0.8312 (2)	0.8176 (3)	0.8325 (1)	-
MNIST35	0.9325 (1)	0.9141 (4)	0.9312 (2)	0.9165 (3)
Mushroom	0.9857 (1)	0.9746 (3)	0.9846 (2)	-
Nursery	0.7746 (2)	0.7644 (3)	0.7915 (1)	-
Occupancy	0.9768 (2)	0.9738 (3)	0.9769 (1)	0.971 (4)
Optdigits	0.9334 (2)	0.9178 (4)	0.9342 (1)	0.9197 (3)
Pen	0.9187 (1)	0.8906 (3)	0.9167 (2)	0.8649 (4)
Segmentation	0.9289 (3)	0.9123 (4)	0.9358 (1)	0.9294 (2)
Spambase	0.909 (2)	0.8922 (4)	0.9085 (3)	0.909 (1)
Transfusion	0.6155 (2)	0.6011 (4)	0.6175 (1)	0.6028 (3)
Twitter	0.6331 (2)	0.6319 (3)	0.6344 (1)	0.625 (4)
USPS	0.8649 (2)	0.8471 (4)	0.8671 (1)	0.8474 (3)
Vehicle	0.6605 (2)	0.6464 (4)	0.6606 (1)	0.658 (3)
Average ALC	0.8423	0.825	0.8408	0.8331
Average Rank	1.5652	3.5217	1.6522	3.2609
Group	A	B	A	B

* Average ALC values of algorithms from the same group are not significantly different based on Tukey's pair-wised at 95% level of significance.

Table 4.3: ALC for Average F1-Scores Curve by Several Weight Parameters for the CAAL Algorithm. Numbers in Parentheses Are the Rankings Based on the Other ALC Scores for the Corresponding Dataset.

Data	β				
	1	2/3	1/2	1/3	0
Banknote	0.9538 (1)	0.9528 (2)	0.9517 (3)	0.9485 (4)	0.8971 (5)
Car	0.6848 (1)	0.6816 (2)	0.6742 (4)	0.6756 (3)	0.6527 (5)
Coil2000	0.5469 (1)	0.5441 (2)	0.5427 (3)	0.5413 (4)	0.5386 (5)
EEG	0.7173 (3)	0.72 (1)	0.7174 (2)	0.7167 (4)	0.6926 (5)
Ibn Sina	0.9437 (1)	0.9431 (2)	0.9412 (4)	0.9413 (3)	0.91 (5)
Letter	0.8328 (3)	0.8335 (2)	0.8304 (4)	0.8837 (1)	0.7865 (5)
Letter NvsM	0.9551 (1)	0.9524 (3)	0.9545 (2)	0.9519 (4)	0.9259 (5)
Letter OQG	0.8784 (1)	0.8773 (2)	0.877 (3)	0.8689 (4)	0.8296 (5)
Letter TIL	0.9628 (1)	0.9616 (2)	0.9601 (3)	0.9581 (4)	0.9288 (5)
Letter VvsY	0.9612 (1)	0.9607 (2)	0.9596 (3)	0.9577 (4)	0.9392 (5)
Mamo	0.8304 (2)	0.8292 (4)	0.8312 (1)	0.8293 (3)	0.828 (5)
MNIST35	0.9331 (1)	0.9314 (3)	0.9326 (2)	0.9286 (4)	0.9 (5)
Mushroom	0.9864 (2)	0.9864 (1)	0.9858 (3)	0.981 (4)	0.9671 (5)
Nursery	0.7709 (4)	0.7709 (3)	0.7739 (2)	0.774 (1)	0.7682 (5)
Occupancy	0.9777 (1)	0.9774 (2)	0.9773 (3)	0.9769 (4)	0.9656 (5)
Optdigits	0.9345 (1)	0.9338 (2)	0.9334 (3)	0.9254 (4)	0.89 (5)
Pen	0.9161 (3)	0.9173 (2)	0.918 (1)	0.9156 (4)	0.8467 (5)
Segmentation	0.9281 (4)	0.9284 (2)	0.9286 (1)	0.9284 (3)	0.8978 (5)
Spambase	0.909 (2)	0.9118 (1)	0.9085 (3)	0.9054 (4)	0.8807 (5)
Transfusion	0.6161 (2)	0.6116 (4)	0.6153 (3)	0.6172 (1)	0.601 (5)
Twitter	0.6195 (5)	0.6358 (3)	0.6329 (4)	0.6417 (2)	0.645 (1)
Vehicle	0.6605 (1)	0.6548 (4)	0.6583 (2)	0.6556 (3)	0.6314 (5)
USPS	0.8605 (4)	0.8632 (3)	0.865 (2)	0.8665 (1)	0.8246 (5)
Average ALC	0.8426	0.8426	0.8422	0.843	0.8151
Average Rank	2	2.3478	2.6522	3.1739	4.8261

* Tukey's pair-wised test only shows a significance difference between average ALC values of $\beta = 0$ and the rest of combinations at 95% level of significance.

Table 4.4: ALC for Average F1-Scores Curve by Several Weight Parameters for the DMAL Algorithm. Numbers in Parentheses Are the Rankings Based on the Other ALC Scores for the Corresponding Dataset.

Data	$\boldsymbol{\beta}$				
	1	2/3	1/2	1/3	0
Banknote	0.9521 (1)	0.9516 (2)	0.95 (3)	0.942 (4)	0.9122 (5)
Car	0.6632 (5)	0.6681 (4)	0.6766 (2)	0.6757 (3)	0.6983 (1)
Coil2000	0.5437 (3)	0.5452 (2)	0.5464 (1)	0.5397 (4)	0.536 (5)
EEG	0.7136 (1)	0.7133 (3)	0.7107 (4)	0.7133 (2)	0.6962 (5)
Ibn Sina	0.8946 (4)	0.8916 (5)	0.8961 (3)	0.9057 (2)	0.9219 (1)
Letter	0.8324 (2)	0.8301 (3)	0.8266 (4)	0.8743 (1)	0.7515 (5)
Letter NvsM	0.9525 (1)	0.9522 (2)	0.9505 (3)	0.9452 (4)	0.9238 (5)
Letter OQG	0.8798 (1)	0.8744 (3)	0.8755 (2)	0.8625 (4)	0.8281 (5)
Letter TIL	0.9568 (2)	0.9564 (3)	0.958 (1)	0.9493 (4)	0.9194 (5)
Letter VvsY	0.9617 (1)	0.9584 (2)	0.9559 (3)	0.9525 (4)	0.9407 (5)
Mamo	0.8323 (2)	0.8309 (3)	0.833 (1)	0.8289 (4)	0.8271 (5)
MNIST35	0.9308 (1)	0.9297 (2)	0.9288 (4)	0.9294 (3)	0.9078 (5)
Mushroom	0.9852 (3)	0.9863 (1)	0.9855 (2)	0.9813 (4)	0.966 (5)
Nursery	0.7913 (2)	0.7898 (3)	0.7921 (1)	0.7891 (4)	0.7733 (5)
Occupancy	0.9779 (1)	0.9766 (3)	0.9766 (2)	0.9722 (4)	0.962 (5)
Optdigits	0.9349 (2)	0.9371 (1)	0.9349 (3)	0.9316 (4)	0.9105 (5)
Pen	0.9191 (1)	0.9178 (3)	0.9179 (2)	0.9156 (4)	0.8762 (5)
Segmentation	0.9357 (2)	0.9334 (3)	0.9358 (1)	0.9334 (4)	0.9158 (5)
Spambase	0.9106 (1)	0.9091 (3)	0.9091 (2)	0.9082 (4)	0.8901 (5)
Transfusion	0.6167 (3)	0.6196 (1)	0.6181 (2)	0.6114 (4)	0.5955 (5)
Twitter	0.6335 (4)	0.6388 (3)	0.6429 (1)	0.6412 (2)	0.6315 (5)
Vehicle	0.6541 (3)	0.6565 (2)	0.657 (1)	0.6455 (4)	0.6272 (5)
USPS	0.8675 (3)	0.8678 (2)	0.8692 (1)	0.8669 (4)	0.8383 (5)
Average ALC	0.8409	0.8406	0.8412	0.8398	0.8195
Average Rank	2.1304	2.5652	2.1304	3.5217	4.6522

* Tukey's pair-wised test only shows a significance difference between average ALC values of $\beta = 0$ and the rest of combinations at 95% level of significance.

Chapter5

ADAPTIVE ACTIVE LEARNING

5.1 Abstract

Most state-of-the-art active learning methodologies are designed to take both classification uncertainty and data density into account. However, they usually ignore the fact that the balance between these two factors needs to be dynamically adapted as the active learning proceeds. We address this challenge by presenting two novel methods to adaptively make an equilibrium between uncertainty and density in active learning. Our first algorithm, KF-RB, assumes the relationship between the uncertainty-density trade-off and active learner's feedback follows a time-varying model. KF-RB uses a restless multi-armed bandit framework to learn this relationship. Our second algorithm, RAFO, extends KF-RB by assuming a smooth function over a continuous action space instead of discretizing it in the form of bandit arms. RAFO benefits from a Gaussian process Bayesian optimization framework with a smooth forgetting property that allows for modeling a time-varying relationship between the learner's performance and the uncertainty-density trade-off. We illustrate the performance of both proposed algorithms on real data, and argue that the dynamic balance provided by both algorithms makes them perform favorably compared to the alternatives.

5.2 Introduction

Machine learning models often require a considerable amount of annotated (labeled) data to reach their full potential. However, obtaining annotation can be time-consuming and expensive, notably in specialized domains where only experts can provide reliable labels [122]. Active learning (AL) aims to facilitate the data annotation process by automatically selecting data instances that seem to benefit the learning model the most. This is usually accomplished with a sequential process. Starting with an initial set of labeled instances a model is trained and then, sequentially, additional instances are queried, labeled, and the model is updated.

The majority of AL strategies are exploitative (uncertainty-based), explorative (density-based), or a combination of both. Exploitation in this context refers to the utilization of a trained classifier using the currently annotated data, and involves querying instances whose label is expected to maximally reduce model's uncertainty of the label of other instances. Exploration, on the other hand, focuses on the selection of instances that better represent the predictive data. Integration of the two criteria has also been studied extensively in the AL literature. This integration is done in various ways that are either implicit through variance reduction strategies, or explicit by involving a clustering technique. Although many of the combined strategies are shown to outperform the single-criterion ones, they often fail to adaptively make a balance between exploitation and exploration criteria throughout the sequential AL process. In this paper, we show how this adaptation benefits the AL process. Specifically, we propose two new approaches that learn to adjust the trade-off between uncertainty and density factors in the sequential process.

One of the tools that capture the essence of the conflict between exploitation and exploration and the dilemma to make a balance between them is Multi-Armed Bandit

(MAB), originally proposed by [138], where at each round of play, a gambler chooses among K slot machines and receives a reward. The goal in MAB is to maximize the expected reward over a time horizon. In the first proposed approach, we construct an MAB learning agent, separate from the active learner, to learn how to make a balance between uncertainty and density which are the exploitation and exploration modules in an AL schema. In this framework, each arm corresponds to a different value for the uncertainty-density trade-off parameter. We utilize a member of the MAB family, Restless Multi-Armed Bandits (RMAB) [149], where temporal changes of the relationship between uncertainty-density and learner's feedback are captured. This choice, as opposed to a traditional MAB is more principled since a specific trade-off value affects the active learner differently when used at different time points of the learning process. Our developed algorithm, KF-RB, learns this RMAB using a linear dynamical mechanism in the state-space format and follows the stochastic procedure of Thompson sampling (TS) [138] to choose arms according to the probability that they provide the maximum reward.

The second proposed algorithm extends KF-RB by assuming a smooth reward function defined over a continuous action space as opposed to the discrete bandit setting in KF-RB. We develop the Reinforced Active Forest (RAFO) by constructing a sequential Bayesian optimization (BO) problem with bandit feedback, adopting a formulation that allows for the reward function to vary with time. BO is a global optimization technique used to find the maximum of an unknown function that is costly to evaluate, or lacks gradient information [93]. To predict this function, BO assumes a prior, usually in a form of a Gaussian process (GP), over the space of functions and sequentially combines it with action-reward observations. GP is a flexible and powerful approach to Bayesian optimization problems since it benefits from a closed-form posterior in the inference process [76]. To be congruous with

the time-varying reward analogy of KF-RB, we model the reward function using a variant of GP that captures the temporal changes of the reward functions by forgetting older action-reward observations in a smooth fashion. Our experiments illustrate the effectiveness of the two adaptive approaches on accelerating the learning process using real-world datasets.

Note that in AL language, the exploitation and exploration terms are sometimes used interchangeably with uncertainty and density (or informativeness and representativeness), respectively. In this paper, we use these terms in the context of dynamic modeling as well.

The layout of the rest of the paper is as follows. Section 5.3 provides a background on the literature of AL and the current approaches for adaptive AL. We also provide a background on RMAB and GP bandit optimization. Section 5.4 fully explains the two proposed methodologies. Section 5.5 explains the experiments in detail and illustrate the outperformance of the proposed methodologies over alternatives. Finally, conclusions are presented in Section 5.6.

5.3 Background

The importance of integrating uncertainty and density in AL has been extensively studied [17, 36, 57, 62, 65, 100, 154]. This integration can be done in various ways. Variance reduction approaches, such as work by [118], explored the input data space implicitly while querying uncertain instances, whereas [154] combined the two criteria explicitly by involving a clustering technique. Among the explicit approaches, a weighted combination of the two criteria has been used by a few studies [17, 74, 82]. Utilization of most clustering methods requires selection of a distance measure, and it is commonly assumed that data can be clustered into several groups, where instances share a common label. Although studies have shown the effectiveness of this cluster

assumption in AL [100, 113], the ambiguity of semi-supervised methods (where we need to learn from partially labeled data sets) persists due to the lack of ground truth about data labels and incapability of evaluating clustering quality with high certainty. Furthermore, the notion of distance in three-dimensional world does not necessarily apply in high-dimensional spaces [35]. Points in higher dimensions tend to be equally-distanced from each other, since beyond some dimensionality most of them are closer to the surface of their hypercube other than being scattered throughout a whole volume [35]. This ambiguates our understanding of these spaces and makes the distinction between high-dimensional data points almost meaningless, especially if we define dissimilarity based on three-dimensional measures such as Euclidean distance. Work by [45] proposed a new strategy to tackle this problem. Their algorithm, Cluster Agnostic Active Learning (CAAL), characterizes density by assuming a fuzzy clustering structure using unsupervised Random Forest distance [14]. They then define cluster uncertainties to represent density, which are then combined with classification uncertainties to score unlabeled instances for label query. More specifically, at each time step t, a supervised model ($\Phi^{(t)}$) is trained using the available labeled dataset $\mathcal{L}^{(t)}$. For each instance in the current unlabeled pool, $x_u \in \mathcal{U}^{(t)}$, classification and clustering scores are defined using Eq. (5.1) and Eq. (5.2), respectively. That is,

$$M_c(x_u) = \underset{c}{Max}\{P(c|\Phi^{(t)}, x_u)\} - \underset{y}{SecMax}\{P(c|\Phi^{(t)}, x_u)\} \tag{5.1}$$

$$M_k(x_u) = \underset{k}{Max}\{P(k|\Psi, x_u)\} - \underset{y}{SecMax}\{P(k|\Psi, x_u)\} \tag{5.2}$$

where $SecMax(.)$ function finds the second greatest value, and Ψ represents a separate supervised model used to learn the association between features and cluster assignments as defined in [45]. $M_c(x)$ and $1 - M_k(x)$ represent the classification and clustering uncertainties, respectively. These scores are then combined to form a

utilization score to evaluate unlabeled instances:

$$U(x) = \beta\Big(M_c(x)\Big) + (1-\beta)\Big(1 - M_k(x)\Big) \tag{5.3}$$

Parameter $0 \leq \beta \leq 1$ in Eq. (5.3) controls the relative importance of the two factors which is held constant throughout the AL process in [45]. Therefore, it does not reflect the dynamic nature of the uncertainty-density trade-off as more instances get labeled. We use the AL procedure of CAAL, but instead, explore feedback-driven approaches to select the trade-off parameter β adaptively.

The conflict between exploitation (choosing best actions based on the current knowledge) and exploration (trying other actions to obtain more information on the environment) and the dilemma to make a balance between them is a well-known problem in multi-armed bandit, reinforcement learning and other areas of artificial intelligence [48]. In AL, this dilemma is translated to balancing uncertainty and density, since the former is concerned with the exploitation of the classifier predictions, whereas the latter explores different areas of the feature space.

As a resource allocation model, MAB lends itself naturally to the AL problems, and several prior works have drawn different analogies between AL and MAB. One analogy is that MAB can help the AL algorithm to combine uncertainty and density in a dynamic fashion. Among AL query strategies that propose ways to make a trade-off between uncertainty and density, very few have focused on the fact that relative importance of uncertainty and density evolves as AL progresses. Work by [82] proposed an adaptive AL framework that dynamically reweighs uncertainty and density by selecting the trade-off parameter from a predefined set such that the expected generalization error of the classifier is minimized. Other than being computationally expensive for multi-class scenarios, the drawback of this approach is its short-sightedness as it is affected by model fluctuations, and the fact that it does

not learn any stable long-run strategy. Work by [36] monitors the average expected error and increases the weight of density once changes in uncertainty start to decrease. The main drawback of this approach is that the cross-over point for switching from uncertainty to density is held constant throughout their AL process. Studies by [23, 38, 102] used the feedback from the classifier to guide the selection of the trade-off parameter in the subsequent rounds. Although their method benefits the current model, it is myopic in a sense that it does not adopt a long-run strategy.

Another analogy between AL and MAB (not discussed in this work) is that AL algorithms can show different performances across different problems. Therefore, similar to an MAB framework, one can create a master strategy where each arm corresponds to a different AL strategy. Although this does not immediately provide a solution for creating the ensemble of active learners, works by [6, 43, 60] developed AL master algorithms using known MAB approaches to blend multiple AL strategies. The idea behind their work is that some AL strategies perform better early on, taking better advantage of the data unsupervised knowledge while some other excel at later iterations. Therefore, one can switch between these strategies to perform well in the long-run. For example, [6] used the adversarial MAB framework, originally proposed by [5], to combine active learners. Each learner (arm) is assigned a weight that is updated iteratively based on its performance. Instances are then queried randomly according to the weighted combination of scores provided by the learners. In a similar work, [38] proposed a Markov decision process (MDP) where AL strategies are treated as states, and actions are discrete values which determine the probabilities of staying in or switching between the states. Although they use a model-free method (Q-learning) to learn their MDP, the interpretation of their approach is not straightforward as several actions result in the same state. Moreover, their limited experimental results (both in terms of the number of states and tested datasets) make

it hard to draw a conclusion on their approach.

The first contribution of this paper is an adaptive strategy to select the trade-off parameter between uncertainty and density in a timely manner using an extension of the traditional MAB, namely the RMAB. Introduced by [149], RMAB assumes a decision-maker chooses an action $a^{(t)} \in \mathcal{A}$ and receives a reward $r^{(t)}$ that follows a time-varying distribution. Similar to other learning processes, reward can only be observed by taking an action, and the goal is to learn an optimal strategy to maximize the cumulative reward. This framework, due to its non-stationary nature, is able to model more complex systems, and has gained more attention recently in various fields [67, 96, 134] including AL [104]. RMAB, as opposed to traditional MAB, provides a more realistic modeling approach for balancing uncertainty and density in AL, since a specific trade-off value can make the active learner behave differently when used at different iterations of the learning process. For instance, giving more weight to density can be more advantageous at the beginning when the classifier is trained on a limited amount of data, whereas in later iterations, uncertainty can be more beneficial to fine-tune the classification model.

To learn the dynamics of the proposed RMAB model, we use Kalman filter [68] which dynamically updates the rewards associated with each bandit arm. Kalman filter has been previously used as a learning module for MAB problems in domains other than AL [49, 54, 146]. For example, [49] proposed the Kalman Filter-Multi Armed Normal Bandit algorithm (KF-MANB), which builds a Bayesian framework to track better actions as changes occur over time and to update hyper-parameters of the reward distribution in a tractable manner. At each time step, one arm is chosen according to the probability that it provides the maximum reward. This is known as probability matching or Thompson sampling (TS) which is often easy to implement and is known to perform well in practice [18, 120]. We take advantage

of the good empirical performance of KF-MANB algorithm as a subroutine in the CAAL algorithm to introduce our first adaptive AL algorithm, namely the KF-RB.

A natural generalization to the MAB problem is to embed the arms into a continuous domain, and define a maximization problem over this entire domain. This immediately lifts the need for defining arms that correspond to discrete values. Mathematically, the continuous MAB framework translates to maximizing the unobserved complex function $z = f(\theta)$ with respect to parameter θ, i.e.

$$\theta^{\star} = \underset{\theta \in \Theta}{argmax}\, z = f(\theta) \tag{5.4}$$

This problem is extensively studied in the Bayesian optimization literature [126]. We take advantage of this framework and assume the relationship between the trade-off parameter β in utility score function Eq. (5.3) and active learner's feedback is characterized by an unknown function. In other words, β and learner's feedback (which we formulate later) correspond to θ and z in Eq. (5.4), respectively.

As mentioned earlier, specific values of β have impact active learner differently at different stages of the AL process. Therefore, $f(.)$ may vary with time. Optimization of a black-box function with temporal changes has also been studied in the context of sequential Bayesian optimization. Work by [7] proposed the TV-GP-UCB algorithm which introduces an Upper Confidence Bound (UCB) GP-based model that seeks sequential optimization of an unknown time-variant function $f^{(t)}(.)$ over the compact and convex set Θ. Mathematically, these frameworks seek $\theta^{(t)}$ such that

$$\theta^{(t)} = \underset{\theta \in \Theta}{argmax}\, z^{(t)} = f^{(t)}(\theta) \tag{5.5}$$

There is a rich body of work on using GP models in sequential Bayesian optimization with bandit feedback [7, 76]. These flexible non-parametric models not only provide information on the location of the optima, but more importantly, they provide

the uncertainty associated with it. To manage exploration versus exploitation within this Bayesian optimization problem, several heuristics such as expected improvement (EI) [98], Thompson sampling (TS) [138] upper confidence bound (UCB) [5] and their variations have been proposed and shown to be successful in practice.

In UCB sampling, the estimated reward is decomposed into two parts: the empirical average gain as an estimate for the true mean and the uncertainty associated with this estimate [95]. A rich theory also exists on UCB performance guarantees when GP priors are assumed. Regret bounds on GP-UCB was first provided in [135], but we use the TV-GP-UCB algorithm by [7] who extended the GP-UCB algorithm for a time-varying problem, benefiting from a smooth forgetting kernel function. For details of the regret bounds on TV-GP-UCB, refer to [7, 135].

5.4 Methodology

In this section we introduce our proposed methodologies, KF-RB and RAFO, for selecting trade-off parameter $\beta^{(t)}$ in Eq. (5.3) in a dynamic fashion.

5.4.1 Active Learning Framework

Consider a pool-based AL scenario [122] with the goal of sequentially learning function $\Phi : X \mapsto Y$, where $X \in \mathbb{R}^{n \times m}$ represents the m-dimensional features set, and $Y \in \mathbb{R}^{n \times 1}$ is the response. Initially the learner is presented with an empty labeled set, $\mathcal{L}^{(0)}$, and an unlabeled pool of data, $\mathcal{U}^{(0)}$. Data labels are provided by an oracle in a sequential manner. At each round t, the learner estimates Φ (denoted by $\hat{\Phi}^{(t)}$) using the available labeled set $\mathcal{L}^{(t)}$, and evaluates the unlabeled pool $\mathcal{U}^{(t)}$ for label query. We utilize the CAAL algorithm proposed by [45] where unlabeled instances are evaluated according on their utility score defined in Eq. (5.3). This score is composed of uncertainty and density factors that are represented by the classification and

clustering uncertainties, defined by Eq. (5.1) and Eq. (5.2), respectively. Parameter $\beta^{(t)}$ is a trade-off parameter to control the relative effect of these factors. Moreover, superscript (t) is a time index indicating the time-varying nature of this parameter.

After the query is made for a selected batch of instances, the labeled set is augmented and learner's belief on Φ is updated ($\hat{\Phi}^{(t)} \mapsto \hat{\Phi}^{(t+1)}$). Given a limited query budget T, the goal is to fit the best classifier.

Our proposed methodologies tackle the problem of choosing $\beta^{(t)}$ as AL progresses. Note that the proposed algorithms are only a subroutine in the whole AL process. The focus of the following sections is on formulating this subroutine.

5.4.2 Choice of Reward Function

Before we elaborate on our proposed methodologies, we need to choose a reward function to assess our choices of $\beta^{(t)}$ based on active learner's feedback.

Consider this AL procedure described in Section 5.4.1 as a process to be optimized (with respect to parameter $\beta^{(t)}$) by learning a strategy from the classifier's feedback. The goal of this strategy is to guide the active learner to choose $\beta^{(t)}$ more strategically rather than fixing it throughout the AL process. Our choice of $\beta^{(t)}$ at each time step t is evaluated (rewarded or penalized) after a query is made.

Various reward functions have been proposed in similar dynamic learning frameworks. In the AL context, work by [23] defined reward as the scaled Kullback–Leibler divergence between the posterior distribution of a classifier before and after a query is made. Studies by [38, 102] utilized the change in the overall classification entropy. Also, [6] proposed Classification Entropy Maximization method (CEM), which despite practical evidence of usefulness has never formally been connected to prediction performance required in a dynamic AL scenario. Work by [60] proposed the importance-weighted accuracy (IW-ACC), which weighs labeled instances with the

inverse of the probability that the instance is queried. They then calculate reward based on the weighted accuracy. Although they show that IW-ACC is an unbiased estimator of the prediction accuracy on the held-out data, their method allows labeled instances to be queried again, which may not seem logical in practice.

Consider the probabilistic model of $\Phi : X \mapsto Y$. Classification entropy for this model is defined as

$$H = -\sum_{i}^{c} P(y_i|X, \Phi) \log P(y_i|X, \Phi) \tag{5.6}$$

where $P(y_i|x, \Phi)$ values are class membership probabilities, X is the features set, and Y is the response. Inspired by [6] and [38], we define reward for choosing a specific $\beta^{(t)}$ based on the reduction in model uncertainty, i.e. difference in the overall classification entropy before and after a query is made according to the selected $\beta^{(t)}$. The idea is that the classification entropy decreases as more data gets labeled, so it is expected that better samples (chosen based on better $\beta^{(t)}$ values) result in higher entropy reductions. We use this reward, formulated in Eq. (5.7), for both of our proposed methodologies. Note that $H^{(\mathrm{t})}$ and $H^{(\mathrm{t+1})}$ indicate the classification entropy before and after a query is made by the active learner.

$$r(\beta^{(t)}) = r^{(t)} = \sum_{i=1}^{|\mathcal{U}^{(t)}|} H^{(t)}(x_i) - \sum_{i=1}^{|\mathcal{U}^{(t+1)}|} H^{(t+1)}(x_i) \tag{5.7}$$

The research into MAB algorithms is closely tied to theoretical performance guarantees and there is little interest in dynamic problems where the distribution of $r^{(t)}$ is unbounded. In practice, however, in cases where the reward bounds is not known, all the rewards are scaled to a bounded interval as learning progresses. We scale reward $r^{(t)}$ ($s : \mathbb{R} \mapsto I = [-1, 1]$) using all the observed rewards until iteration t.

$$s^{(t)} = s(r^{(t)}) = (r^{(t)} - Min(I))\frac{Max(I) - Min(I)}{Max_i r^{(i)} - Min_i r^{(i)}} - Min(I) \tag{5.8}$$

where $2 \leq i \leq t$. This reward has been used previously by [38]. However, they use this reward as a tool to switch between different AL query strategies as opposed to our

method, which is focused on learning the uncertainty-density trade-off for specific AL strategy. One difference between the scaling process in [38] and ours is that we update all previous scaled rewards after $r^{(t)}$ is observed if $r^{(t)} = Max_i r^{(i)}$ or $r^{(t)} = Min_i r^{(i)}$. This update is used to assess rewards relative to each other.

5.4.3 Baseline Algorithm: A Feedback-driven Approach

Before introducing our proposed methodologies, we briefly overview a feedback-driven approach for updating $\beta^{(t)}$, used by [23, 38, 102], as a competing algorithm with our proposed ones. As mentioned earlier, feedback-driven approaches have been previously used in AL to make a balance between uncertainty and density dynamically. Works by [23, 102] propose an update schema to obtain $\beta^{(t+1)}$ from $\beta^{(t)}$ according to the classifier's feedback:

$$\beta^{(t+1)} = Max\Big(Min\big(\beta^{(t)}\lambda exp(s^{(t)}), 1-\beta_{Min}\big), \beta_{Min}\Big) \tag{5.9}$$

where β_{Min} ensures a minimum level of exploration, λ is the learning rate to control the effect of reward on how fast $\beta^{(t)}$ changes, and $s^{(t)}$ is the scaled reward from Eq. (5.8). Although this approach is myopic, and does not have a long-run strategy, it has been shown to perform relatively well in practice [23, 38, 102]. Therefore, we use it as a baseline for our evaluations.

5.4.4 KF-RB: A Kalman Filter Restless Bandit Approach

We introduce the KF-RB algorithm which solves the problems of choosing $\beta^{(t)}$ using a Kalman filter RMAB approach. Consider the AL framework described in Section 5.4.1. Each time after a query is made, the learner updates its estimate of model Φ from $\hat{\Phi}^{(t)}$ to $\hat{\Phi}^{(t+1)}$. At this stage the learner has the chance to evaluate the query and adjust the selection of $\beta^{(t)}$. To formulate this evaluation we use scaled

reward $s^{(t)}$) defined in Eq. (5.7). As mentioned earlier, $H^{(t)}$ and $H^{(t+1)}$ in this equation are based on classification probabilities according to $\hat{\Phi}^{(t)}$ and $\hat{\Phi}^{(t+1)}$ respectively. Now the question is how to select $\beta^{(t)}$?

In order to answer this question, KF-RB constructs a dynamic learning framework to learn the relationship between $\beta^{(t)}$ and $s^{(t)}$. We consider an RMAB framework where a set of arms $\mathcal{B}$ each corresponding to a value for $\beta^{(t)}$ is given, i.e. $\beta^{(t)}$ chooses from $\mathcal{B}$. At each time step t, and before a query is made, one of the arms is chosen to provide $\beta^{(t)}$, and utility score are calculated according to Eq. (5.3). Upon obtaining $\hat{\Phi}^{(t+1)}$, reward $s^{(t)}$ is observed which corresponds to the selected $\beta^{(t)}$. As AL progresses, more $(\beta^{(t)}, s^{(t)})$ pairs are observed. Therefore, we can learn the reward distribution for each arm.

At this stage, we take advantage of the Bayesian reasoning to define a framework for stochastic optimal decision making. Following [49], we use a Kalman filter-based learning process for the RMAB problem under normally distributed rewards. More specifically, we assume the reward for each arm follows a Gaussian distribution with a time-varying mean $\mu_i^{(t)}$ according to:

$$s_i^{(t)} = \mu_i^{(t)} + \epsilon_i^{(t)} \qquad \epsilon_i^{(t)} \sim \mathcal{N}(0, \sigma_\epsilon^2) \tag{5.10a}$$

$$\mu_i^{(t)} = \mu_i^{(t-1)} + \xi_i^{(t)} \qquad \xi_i^{(t)} \sim \mathcal{N}(0, \sigma_\xi^2) \tag{5.10b}$$

where $1 \leq i \leq |\mathcal{B}|$. Here, σ_ϵ and σ_ξ are two sources of variability introduced to the process which represent observation and transition variances respectively [49]. Moreover, we assume that arms' rewards are independent of the selection of the other arms.

Equations (5.10a)–(5.10b) are also known as the general form of local level models that are suited to dynamic time series models that involve unobserved components [55]. It also represents a wide range of autoregressive integrated moving average

(ARIMA) models that are widely used in timeseries analysis and forecasting [10, 55].

To consider the similarities between the state-space model in Equations (5.10a)–(5.10b) and ARIMA models, subtract Eq. (5.10a) from itself lagged one period. Substituting for $\mu_i^{(t)}$ from Eq. (5.10b) yields:

$$\Delta s_i^{(t)} = s_i^{(t)} - s_i^{(t-1)} = \mu_i^{(t)} - \mu_i^{(t-1)} + \epsilon_i^{(t)} - \epsilon_i^{(t-1)} \tag{5.11}$$

Since Eq. (5.10b) implies that,

$$\mu_i^{(t)} - \mu_i^{(t-1)} = \xi_i^{(t)} \tag{5.12}$$

we can rewrite Eq. (5.11) as,

$$\Delta s_i^{(t)} = s_i^{(t)} - s_i^{(t-1)} = \xi_i^{(t)} + \Delta\epsilon_i^{(t)} \tag{5.13}$$

which is stationary and has the same autocorrelation plot as an MA(1) process. This implies that the local level model in Equations (5.10a)–(5.10b) is an ARIMA(0,1,1). A comprehensive overview of the equivalences between state-space and ARIMA models is provided in [55].

The transition variance is basically the time-dependent variation in the mean rewards. The Kalman filter procedure proposed in [49] provides Bayesian updating rules as the AL proceeds and action/reward pairs $(\beta^{(t)}, s_i^{(t)})$ are observed. Next, our hypothesis on the expected reward for the pulled arm is updated using Kalman equations. This means that the posterior of one time step will be the prior of next one $\Big(\mathcal{N}\big(\tilde{\mu}^{(t)}, \tilde{\sigma}^{2(t)}\big) \rightarrow \mathcal{N}\big(\tilde{\mu}^{(t+1)}, \tilde{\sigma}^{2(t+1)}\big)\Big)$. The mean and variance of posterior reward distribution for all arms are obtained from

$$\begin{aligned}
\tilde{\mu}_{b_i}^{(t+1)} &= \begin{cases} \tilde{\mu}_{b_i}^{(t)} + \omega_i^{(t)}(r_{b_i}^{(t)} - \tilde{\mu}_{b_i}^{(t)}) & \text{if } \beta^{(t)} = b_i \\ \tilde{\mu}_{b_i}^{(t)} & \text{if } \beta^{(t)} \neq b_i \end{cases} \\
\tilde{\sigma}_{b_i}^{2\,(t+1)} &= (1 - \omega_i^{(t)})(\tilde{\sigma}_{b_i}^{2\,(t)} + \sigma_\xi^2)
\end{aligned} \tag{5.14}$$

where $\omega_i^{(t)} \in [0,1]$ is the Kalman gain defined as

$$\omega_i^{(t)} = \begin{cases} \frac{\tilde{\sigma}_i^{2(t)} + \sigma_\xi^2}{\tilde{\sigma}_i^{2(t)} + \sigma_\xi^2 + \sigma_\epsilon^2} & \text{if } \beta^{(t)} = i \\ 0 & \text{if } \beta^{(t)} \neq i \end{cases} \tag{5.15}$$

Since we only obtain information about the pulled arms, it makes sense to set $\omega_i^{(t)} = 0$ for all the unchosen arms. However, since this is an RMAB setting, the expected reward for all the arms are equally likely to change with time.

The variance update rule in Eq. (5.14) takes care of this through increasing the variance of expected reward so that the unchosen arms have a higher chance to be selected in the next time step. Although [37] has shown that convergence of $\omega_i^{(t)}$ to the steady state $\omega_s = 1/2\sqrt{\frac{\sigma_\xi^4}{\sigma_\epsilon^4} + 4\frac{\sigma_\xi^2}{\sigma_\epsilon^2}} - \frac{\sigma_\xi^2}{\sigma_\epsilon^2}$ can happen quickly, the benefit of utilizing Kalman filter is still valid since the quick convergence happens if reward for a particular bandit arm is observed at all time points while in practice it takes the algorithm some time to gather information about each arm.

Following [49], we use the probability of maximum reward, i.e., TS, to select next $\beta^{(t)}$. The probability of achieving the maximum reward by an arm is naturally derived from combining the expected reward and the uncertainty associated with it. While TS might seem myopic as it is merely based on the probability of obtaining the optimum reward for the immediate decision, it actually allows for a good balance between exploitation and exploration in the dynamic environment. The longer an arm has not been chosen, the higher the chance that it provides a higher reward [134, 147]. Moreover, TS has been shown to outperform other similar heuristic strategies [49, 51, 147].

$$\beta^{(t)} = \underset{b_i \in \mathcal{B}}{argmax}\, \mathbb{E}_{\mathcal{D}^{(t)}}[s^{(t)} | \beta^{(t)} = b_i] \tag{5.16}$$

where $\mathcal{D}^{(t)}$ is the prior distribution of reward at time step t (in our case $\mathcal{D}^{(t)} = \mathcal{N}\big(\tilde{\mu}^{(t)}, \tilde{\sigma}^{2(t)}\big)$).

5.4.5 RAFO: Reinforced Active Forest

One of the drawbacks of using an MAB setup, like that of KF-RB, or a Markov decision process, similar to the work in [38], is that these approaches discretize the action space $\mathcal{B}$. Instead, inspired by [7], we propose a Bayesian optimization framework to adaptively select $\beta^{(t)}$ from a continuous interval based on a time-varying GP.

In Bayesian optimization framework we seek to sequentially optimize the unknown function f of the $z = f(x) + e$ form over a compact set $\mathcal{X}$ where $e \sim \mathcal{N}(0, \sigma^2)$. The BO agent assumes a prior belief over f, and samples f based on this belief where it thinks the optimum occurs. The prior is then updated to obtain the posterior (which acts as the prior for the next round) based on the new observation.

Let $f^{(t)}$ be the BO agent estimate of function f at time step t. Assuming a GP prior over $f^{(t)}$, the posterior of $f^{(t)}$ (or the prior for $f^{(t+1)}$) is a GP distribution with mean and covariance structure of

$$\begin{aligned} \mu^{(t+1)}(x) &= \mathbf{k}^{(t)}(x)^T (\mathbf{K}^{(t)} + \sigma^2 I^{T'})^{-1} z^{(t)} \\ \sigma^{2(t+1)}(x, x') &= k(x, x) - \mathbf{k}^{(t)}(x)^T (\mathbf{K}^{(t)} + \sigma^2 I^{T'})^{-1} k^{(t)}(x) \end{aligned} \tag{5.17}$$

where $\mathbf{K}^{(t)} = [k(x, x')]_{x,x' \in \mathbb{X}^{(t)}}$ and $\mathbf{k}^{(t)}(x) = [k(x^{(i)}, x)]_{i=1}^{t}$ with $k(.)$ a kernel function, and $I^{T'}$ is the $T' \times T'$ identity matrix where T' is the number of evaluations made on function f until time point t. Also, $\mathbb{X}^{(t)}$ denotes points where function f has been evaluated on until time point t. Examples of kernel function $k(.)$ are

$$\begin{aligned} k_{SE}(x, x') &= \exp\big(-\frac{||x - x'||^2}{2l^2}\big) \\ k_{Matérn}(x, x') &= \frac{2^{1-\nu}}{\Gamma(\nu)} \big(\frac{\sqrt{2\nu}||x - x'||}{l}\big)^{\nu} \times B_{\nu}\big(\frac{\sqrt{2\nu}||x - x'||}{l}\big) \end{aligned} \tag{5.18}$$

with $l, \nu > 0$ are hyperparameters, and B_{ν} denotes the modified Bessel function [7, 135].

Although we learn function f sequentially, Bayesian updating rules of Eq. (5.17) are based on the assumption that function f does not change with time. Instead,

assume f follows a time-varying GP. Therefore, the estimation of f between time point t and $t+1$ (i.e. $f^{(t)}$ and $f^{(t+1)}$) should consider that f has changed between these time points.

In order to adjust the GP posterior updating rules in Eq. (5.17) under the assumption that f varies with time, we follow the time-varying GP model, TV-GP-UCB, proposed by [7] where the early observations receive less weight as time proceeds. Updating rules for the posterior reward distribution are provided as

$$\begin{aligned} \widetilde{\mu}^{(t+1)}(x) &= \tilde{\mathbf{k}}^{(t)}(x)^T(\widetilde{\mathbf{K}}^{(t)} + \sigma^2 I^{T'})^{-1} s^{(t)} \\ \widetilde{\sigma^2}^{(t+1)}(x,x') &= k(x,x) - \widetilde{\mathbf{k}}^{(t)}(x)^T(\widetilde{\mathbf{K}}^{(t)} + \sigma^2 I^{T'})^{-1}\widetilde{k}^{(t)}(x) \end{aligned} \tag{5.19}$$

where $\widetilde{\mathbf{K}}^{(t)} = \mathbf{K}^{(t)} \circ \mathbf{D}^{(t)}$ with $\mathbf{D}^{(t)} = [1-\epsilon^{|i-j|/2}]_{i,j=1}^{T'}$, $\widetilde{\mathbf{k}}^{(t)} = \mathbf{k}^{(t)} \circ \mathbf{d}^{(t)}$ with $\mathbf{d}^{(t)} = [1-\epsilon^{T'+1-i/2}]_{i=1}^{T'}$, $\circ$ is the Hadamard product. The older a sample is, the smaller the corresponding entries of $\mathbf{d}^{(t)}$ and $\mathbf{D}^{(t)}$ would be making the contribution of the sample smaller in Eq. (5.19).

Each time we have a new prior with updated parameters obtained from updating rules in Eq. (5.19), the TV-GP-UCB algorithm constructs an upper confidence bound using a linear combination of the average expected value of $f^{(t)}$ and the variance associated with it, i.e. $\widetilde{\mu}^{(t)}$ and $\widetilde{\sigma^2}^{(t)}$. $x^{(t+1)}$ is the argument that maximizes this bound and is the next point to evaluate $f^{(t)}$.

Algorithm 4 summarizes the TV-GP-UCB method by [7]. Note that parameter $\gamma^{(t)}$ in the UCB (Line 2 of Algorithm 4) is a multiplier to adjust $\widetilde{\mu}^{(t)}(x)$ against $\sigma^{(t)}(x)$. In other words, $\gamma^{(t)}$ makes the trade-off between exploitation of the available knowledge on where the optimum happens ($\widetilde{\mu}^{(t)}$) and exploration of the action space X ($\sigma^{(t)}(x)$).

We utilize the TV-GP-UCB framework and introduce our second adaptive AL algorithm, the Reinforced Active Forest (RAFO). In order to select $\beta^{(t)}$, RAFO assumes $s^{(t)} = f^{(t)}(\beta^{(t)}) + e^{(t)}$ where $s^{(t)}$ is the active learner's feedback as defined in

Algorithm 4 Time-Varying GP-UCB (TV-GP-UCB) by [7]

Input: Domain $\mathcal{X}$, prior $\mathcal{GP} \sim (\widetilde{\mu}^{(0)}, \widetilde{\sigma}^{(0)})$, and ϵ

1: **for** t: 0 to T-1 **do**

2: Choose $x^{(t+1)} = argmax_{x \in \mathcal{X}} \widetilde{\mu}^{(t)}(x) + \sqrt{\gamma^{(t)}}\sigma^{(t)}(x)$

3: Sample $s^{(t+1)} = f^{(t+1)}(x^{(t+1)}) + e^{(t+1)}$

4: Obtain $\tilde{\mu}^{(t+1)}$ and $\tilde{\sigma}^{(t+1)}$ based on updating rules in Eq. (5.19)

5: **end for**

Eq. (5.10), and $e^{(t)} \sim \mathcal{N}(0, \sigma^2)$ is noise. Note that we drop index i in $s_i^{(t)}$ because we no longer have discrete arms, but rather a single reward for the whole model.

RAFO starts with a few warp-up iterations to accumulate initial observations based on random actions and their corresponding rewards. This is required to start the GP process. In other words, RAFO begins similar to CAAL and KF-RB with the only difference of selecting $\beta^{(t)}$ at random. After this initial phase, the learner starts the Bayesian updates. Each time a query is made, the learner updates its estimate of the classification model Φ from $\hat{\Phi}^{(t)}$ to $\hat{\Phi}^{(t+1)}$, and adjusts the previous selection of $\beta^{(t)}$ according to the observed reward Eq. (5.8) and the TV-GP-UCB module in Algorithm 4. The output of this module is $\beta^{(t+1)}$ which is used for the next label query.

As a final note, theoretical bounds on TV-GP-UCB performance (in terms of regret) as well as selection conditions for $\gamma^{(t)}$ are provided in [7]. Suffice it to say that although more conservative bounds for $\gamma^{(t)}$ have been driven by several studies, e.g. [7, 69], they indicate that $\gamma^{(t)} \in \mathcal{O}(\log(t))$ provides a good empirical performance. Therefore, we set $\gamma^{(t)} = \log(t)$ for RAFO.

5.5 Experiments and Results

5.5.1 Experiments Setups

We evaluated the performance of CAAL, the baseline algorithm (described in Section 5.4.3), KF-RB, and two versions of RAFO denoted as RAFO_0 with $\epsilon = 0$ and RAFO_+ with $\epsilon > 0$. All algorithms are evaluated on benchmark datasets used in [45] and summarized in Table 5.1. Evaluations were made in terms of the Area Under the Learning Curve (ALC) [53]. The learning curves are based on average F1-scores over 15 replicates to account for randomness at multiple stages of the algorithms including the clustering and supervised modeling steps. Moreover, for all experiments the query size was set to one.

For the baseline algorithm, we tested various values for the learning rate λ. Best results were obtained by $\lambda = 0.5$ (similar to work by [23, 38]). Since the other methods have the option to set $\beta^{(t)}$ to both 0 and 1, we set $\beta_{Min} = 0$. Therefore, all the algorithms can explore the $[0, 1]$ interval. For CAAL, we followed the setup in [45], where $\forall t \in T$, $\beta^{(t)} = \beta$ is fixed throughout the AL iterations and $\beta = 0.5$ achieved the best results in terms of average ALC. For the rest of the algorithms, KF-RB, RAFO_0, and RAFO_+ we follow the same AL procedure as CAAL. The only difference is the selection of $\beta^{(t)}$.

For KF-RB, we considered 11 equidistant points on the $[0, 1]$ interval, each corresponding to one arm. In the absence of initial state and covariance structure for the Kalman filter, it is a common practice to set the average arm rewards to 0 ($\tilde{\mu}^{(0)}$=0) and their variance $\tilde{\sigma}^{2(0)}$ to a high number in order to attenuate the influence of this initial guess [49, 85]. Therefore, we set $\widetilde{\mu}^{(0)} = 0$ and $\widetilde{\sigma^2}^{(0)} = 0.5$ for all arms given that reward $s^{(t)} \in [0, 1]$.

As mentioned earlier, the Kalman gain steady state is proportional to $\frac{\sigma_\xi}{\sigma_\epsilon}$, and not

Table 5.1: Test Datasets

Dataset	Train	Test	Features	Classes	Min %
Banknote	686	686	4	2	23.52 %
Car	864	864	62	2	44.46%
Coil2000	4911	4911	85	2	5.97 %
EEG	7490	7490	14	2	44.88%
Ibn Sina	10361	10361	92	2	37.84%
Letter	10000	10000	16	26	3.67%
Letter NvsM	788	787	16	2	49.71%
Letter OQG	1155	1154	16	3	32.61%
Letter TIL	1156	1156	16	3	32.61%
Letter VvsY	788	787	16	2	49.71%
Mamo	415	415	5	4	3.76 %
MNIST35	5776	5776	784	2	46.93%
Mushroom	4062	4062	22	2	48.20%
Nursery	6479	6479	8	2	48.55%
Occupancy	10280	10280	5	2	2.53 %
OptDigits	3823	1797	64	2	9.86%
Pen	7494	3498	16	10	9.6%
Segmentation	1155	1155	19	7	14.29%
Spambase	2300	2301	57	2	39.40%
Transfusion	374	374	4	2	23.80%
Twitter	490	123	292	2	44.7%
USPS	7291	2007	256	10	7.61%
Vehicle	423	423	18	4	32.65 %

necessarily the absolute values of σ_ξ^2 and σ_ϵ^2. For a fixed value of σ_ξ^2), we conducted a sensitivity analysis using different values of σ_ϵ^2. Results are provided in Table 5.3, which is discussed shortly. However, based on this analysis, we set the observation and transition perturbations $(\sigma_\xi^2, \sigma_\epsilon^2)$ to 0.05 and 0.2 respectively.

We used *Matérn*52 kernel for RAFO to achieve the regret bounds shown in [7] for TV-GP-UCB. In order to learn the best value for ϵ in Eq. (5.19), [7] has used a marginal likelihood maximization approach based on a considerable amount of training data. Since we are in a cold start AL setup, we could not use this approach. Instead, we did a sensitivity analysis on multiple reasonable values of $\epsilon \in \{0, 0.001, 0.01, 0.03\}$. Best results were obtained by $\epsilon = 0.001$. Note that ϵ quantifies how much the function changes at each time step. If $\epsilon = 0$ we will have $RAFO_0$ which is the time-invariant version of RAFO, whereas if $\epsilon = 1$ then the reward functions are independent between time steps. Therefore, the higher the value of ϵ, the less the dependency between the function values at two consecutive time steps.

We considered 5 warm-up iterations for RAFO where $\beta^{(t)}$ was sampled uniformly from the $[0, 1]$ interval. To make a fair comparison, all the other algorithms followed the same rule.

5.5.2 Discussion of Results

Figure 5.1 presents the F1-score learning curves for CAAL, baseline, KF-RB RAFO_0, and RAFO_+ algorithms over 300 iterations. Table 5.2 illustrates the ALC scores and ranks based on the corresponding learning curves. The best average ALC was obtained by RAFO_+ followed by KF-RB and the baseline. This means that all the adaptive algorithms performed better on average. In terms of the average rank, RAFO_+ and KF-RB came in first and second place respectively. However, the baseline algorithm was outperformed by CAAL.

Although for the majority of cases RAFO_+ performs well from the early iterations, an interesting observation is the behavior of RAFO_+ towards the last iterations such as in Fig. 5.2d, Fig. 5.2o, Fig. 5.2q, and Fig. 5.2t where the learning accelerates quickly. This might be due to the improved selection of $\beta^{(t)}$ by GP toward the end.

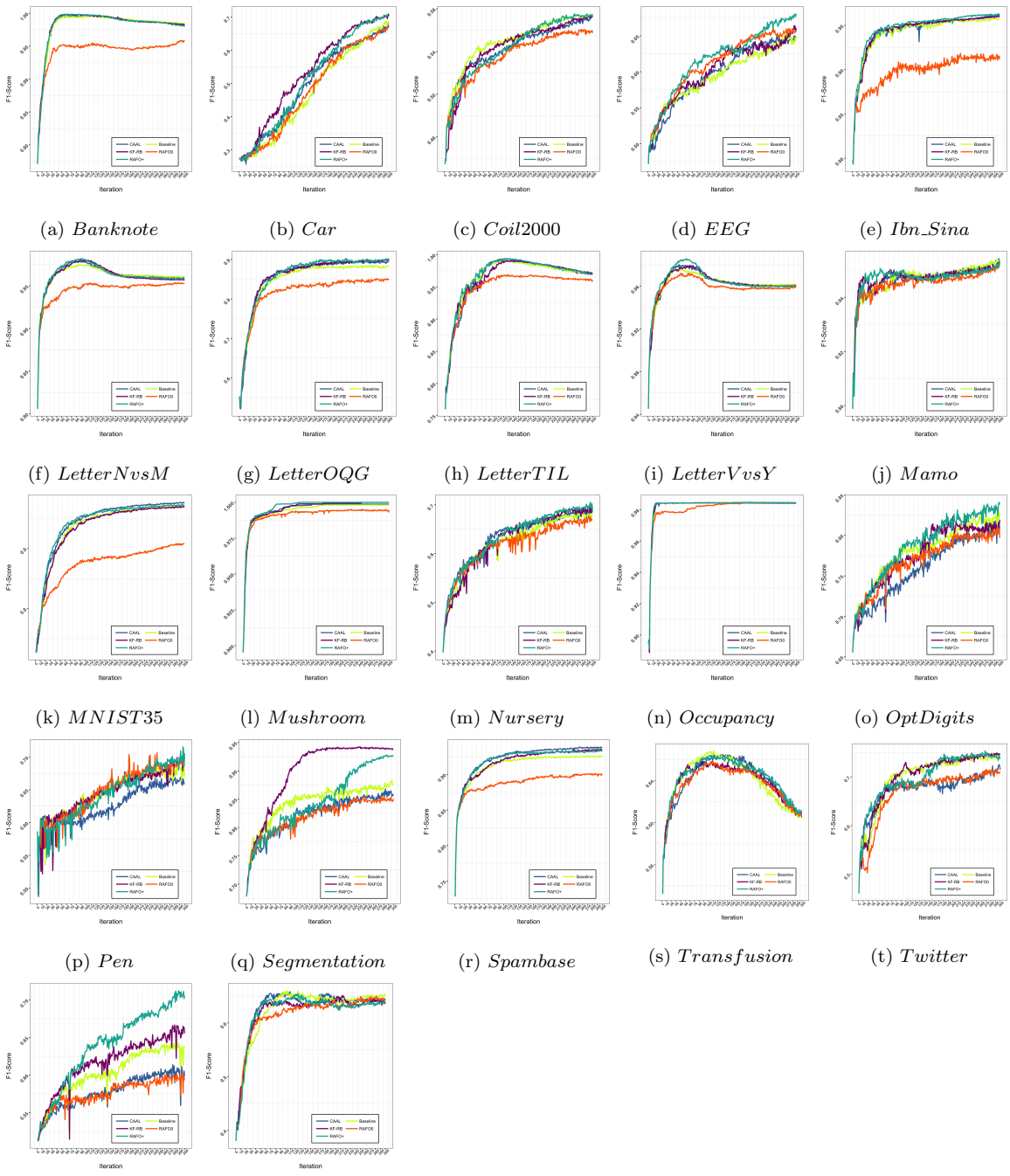
The analysis of RAFO_0 learning curves indicates the importance of capturing the underlying time-varying nature of expected reward with respect to $\beta^{(t)}$. Although in a few cases, e.g. Fig. 5.2d, Fig. 5.2o, and Fig. 5.2p, RAFO_0 performs quite well, for majority of the cases it falls behind after a few iterations.

To show the stability of the proposed approaches, we compared the variability in performance of KF-RB and RAFO_+ and those of the alternatives. Figure 5.2 shows the standard deviation of F1-scores for 15 replicates of each algorithm. Both KF-RB and RAFO_+ clearly have lower variations throughout the learning process, which indicates that the mean performance of these algorithms show consistently better results. Especially for EEG, Nursery, MNIST35, Mushroom, and Segmentation datasets where KF-RB and RAFO_+ clearly have more stable performances. The only dataset that KF-RB shows slightly higher variation is USPS.

Table 5.3 presents the average ALC results for various values of σ_ϵ^2 when $\sigma_\xi^2 = 0.05$. We see that $\sigma_\epsilon^2 = 0.5$ achieves the first rank in terms of average ALC, but the best average rank is obtained by $\sigma_\epsilon^2 = 0.2$. If we compare these results with those of Table 5.2, we see that although $\sigma_\epsilon^2 = 0.5$ obtains a slightly better average ALC, the overall ranking among other algorithms does not change.

5.6 Conclusion

In this work, we developed two new algorithms,KF-RB and RAFO, that dynamically adjust the trade-off between uncertainty and density elements in AL using a bandit optimization setup with dynamic rewards. KF-RB and RAFO, in contrast to their alternatives, accelerate classification learning while balancing uncertainty versus density in a more strategic manner. Both KF-RB and RAFO are the first AL algorithms that take advantage of restless bandits and GP-UCB to balance uncertainty and density adaptively. Our experiments on real-world data sets show the



(a) *Banknote* (b) *Car* (c) *Coil*2000 (d) *EEG* (e) *Ibn_Sina*

(f) *LetterNvsM* (g) *LetterOQG* (h) *LetterTIL* (i) *LetterVvsY* (j) *Mamo*

(k) *MNIST*35 (l) *Mushroom* (m) *Nursery* (n) *Occupancy* (o) *OptDigits*

(p) *Pen* (q) *Segmentation* (r) *Spambase* (s) *Transfusion* (t) *Twitter*

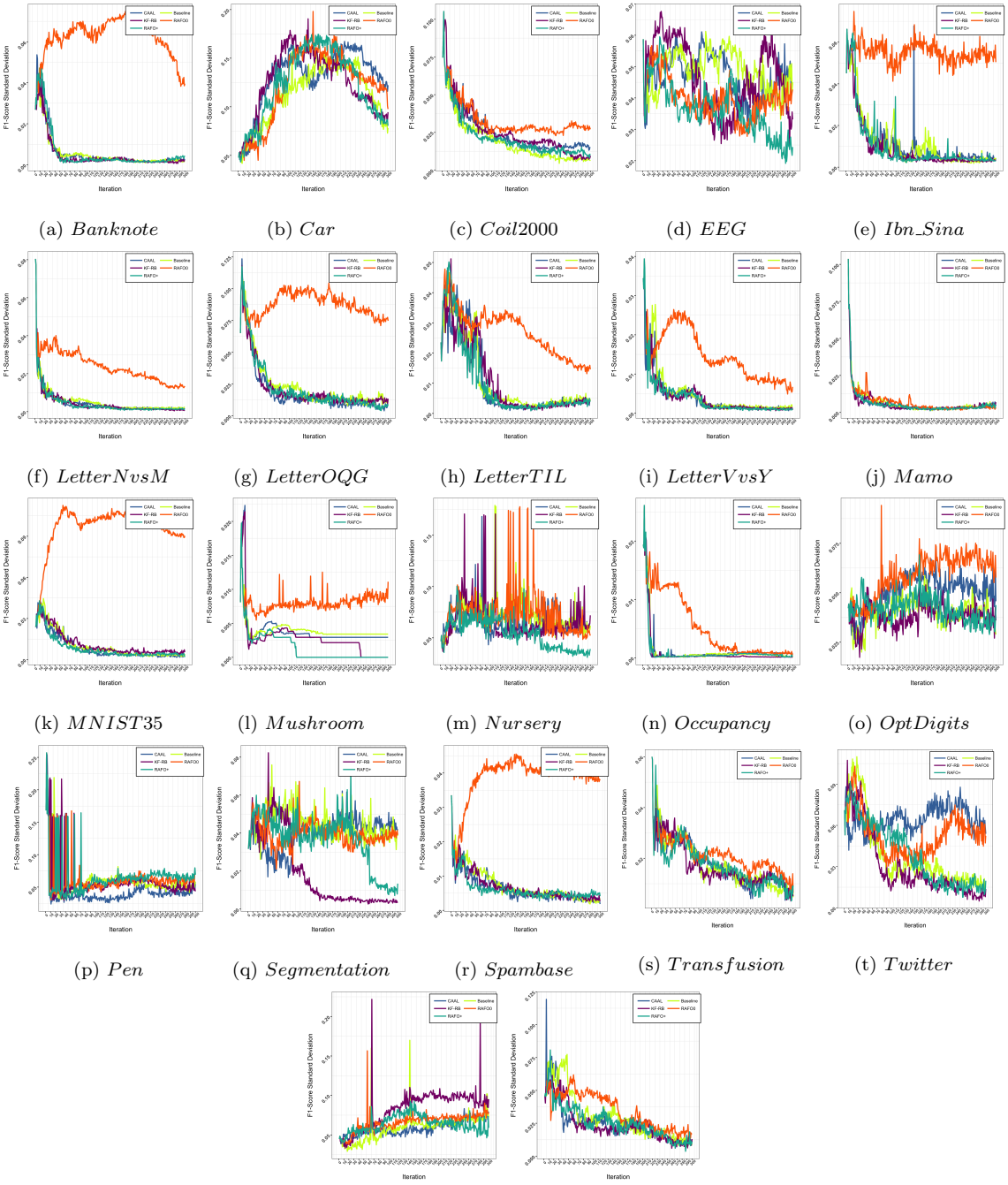
(u) *USPS* (v) *Vehicle*

Figure 5.1: F1-Score Learning Curves for CAAL($\beta = 1/2$), Baseline ($\lambda = 0.5$), KF-RB, RAFO_0 and RAFO_+. Color Legend: CAAL (Solid Blue), Baseline (Solid Yellow), KF-RB (Solid Purple), RAFO_0(solid Orange) and RAFO_+ (Solid Green). See Color Version for the Best View.

Table 5.2: ALC for Average F1-Scores Curve for CAAL, Baseline, KF-RB $RAFO_0$, and $RAFO_+$ Algorithms. Numbers in Parentheses Are the Rankings Compared to the Other ALC Scores for the Corresponding Dataset.

Dataset	**CAAL**	**Baseline**	**KF-RB**	**$RAFO_0$**	**$RAFO_+$**
Banknote	0.9824 (2)	0.9816 (4)	0.9829 (1)	0.9432 (5)	0.9824 (3)
Car	0.4971 (3)	0.4773 (5)	0.5327 (1)	0.4815 (4)	0.5141 (2)
Coil2000	0.5431 (4)	0.5492 (1)	0.5439 (2)	0.5357 (5)	0.5437 (3)
EEG	0.5972 (4)	0.5948 (5)	0.6021 (3)	0.6116 (2)	0.618 (1)
Ibn Sina	0.9448 (3)	0.943 (4)	0.9448 (2)	0.8985 (5)	0.9477 (1)
Letter OQG	0.8581 (3)	0.8488 (4)	0.8596 (2)	0.8162 (5)	0.8606 (1)
Letter TIL	0.9632 (4)	0.9641 (2)	0.9632 (3)	0.9487 (5)	0.9663 (1)
Letter VvsY	0.9624 (3)	0.9623 (4)	0.9627 (2)	0.9587 (5)	0.9641 (1)
Letter NvsM	0.9626 (2)	0.9616 (4)	0.9618 (3)	0.9456 (5)	0.9627 (1)
Mamo	0.8468 (4)	0.8469 (3)	0.8474 (2)	0.8452 (5)	0.8474 (1)
MNIST35	0.9443 (1)	0.9375 (3)	0.9359 (4)	0.8745 (5)	0.9442 (2)
Mushroom	0.9948 (3)	0.9938 (4)	0.9951 (2)	0.9909 (5)	0.9964 (1)
Nursery	0.6331 (2)	0.6216 (4)	0.6283 (3)	0.617 (5)	0.6348 (1)
Occupancy	0.9828 (1)	0.9825 (4)	0.9826 (3)	0.9806 (5)	0.9826 (2)
Optdigits	0.7596 (5)	0.7803 (2)	0.7769 (3)	0.7694 (4)	0.7877 (1)
Pen	0.6257 (5)	0.6456 (2)	0.6445 (3)	0.6477 (1)	0.6435 (4)
Segmentation	0.8216 (4)	0.8436 (3)	0.8993 (1)	0.8181 (5)	0.8464 (2)
Spambase	0.9233 (1)	0.9134 (4)	0.9183 (3)	0.8885 (5)	0.9207 (2)
Transfusion	0.6398 (2)	0.6396 (3)	0.6383 (4)	0.6359 (5)	0.6421 (1)
Twitter	0.6758 (4)	0.7005 (3)	0.7023 (1)	0.6681 (5)	0.7011 (2)
USPS	0.5775 (4)	0.6024 (3)	0.6167 (2)	0.5754 (5)	0.6389 (1)
Vehicle	0.628 (1)	0.624 (2)	0.6211 (4)	0.615 (5)	0.6217 (3)
Mean ALC	0.7649	0.7673	0.7734	0.7517	**0.7737**
Mean Rank	2.9565	3.2174	2.4348	4.6087	**1.7826**

clear improvements over multiple alternatives.



(a) *Banknote* (b) *Car* (c) *Coil*2000 (d) *EEG* (e) *Ibn_Sina*

(f) *LetterNvsM* (g) *LetterOQG* (h) *LetterTIL* (i) *LetterVvsY* (j) *Mamo*

(k) *MNIST*35 (l) *Mushroom* (m) *Nursery* (n) *Occupancy* (o) *OptDigits*

(p) *Pen* (q) *Segmentation* (r) *Spambase* (s) *Transfusion* (t) *Twitter*

(u) *USPS* (v) *Vehicle*

Figure 5.2: Standard Deviation of F1-Scores Over 15 Replicates for CAAL($\beta = 1/2$), Baseline ($\lambda = 0.5$), KF-RB, RAFO_0 and RAFO_+. Color Legend: CAAL (Solid Blue), Baseline (Solid Yellow), KF-RB (Solid Purple), RAFO_0(solid Orange) and RAFO_+ (Solid Green). See Color Version for the Best View.

Table 5.3: ALC for Average F1-Scores Curve by Several σ_ϵ^2 Values for KF-RB. Numbers in Parentheses Are the Rankings Based on the Other ALC Scores for the Corresponding Dataset.

Dataset	$\boldsymbol{\sigma_\epsilon^2}$			
	0.5	0.2	0.1	0.05
Banknote	0.9827 (1)	0.9829 (1)	0.9828 (1)	0.983 (1)
Car	0.5236 (1)	0.5327 (1)	0.5126 (3)	0.5058 (2)
Coil2000	0.5431 (3)	0.5439 (2)	0.5446 (2)	0.5491 (2)
EEG	0.6029 (3)	0.6021 (3)	0.5912 (5)	0.5988 (3)
Ibn Sina	0.9432 (3)	0.9448 (2)	0.9446 (3)	0.9439 (3)
Letter OQG	0.856 (3)	0.8596 (2)	0.8562 (3)	0.858 (3)
Letter TIL	0.962 (4)	0.9632 (3)	0.9628 (4)	0.9621 (4)
Letter VvsY	0.9631 (2)	0.9627 (2)	0.9631 (2)	0.9629 (2)
Letter NvsM	0.9615 (4)	0.9618 (3)	0.9617 (3)	0.9616 (3)
Mamo	0.8472 (2)	0.8474 (2)	0.8469 (3)	0.8462 (4)
MNIST35	0.9369 (4)	0.9359 (4)	0.9376 (3)	0.9362 (4)
Mushroom	0.9955 (2)	0.9951 (2)	0.9954 (2)	0.9953 (2)
Nursery	0.6297 (3)	0.6283 (3)	0.6298 (4)	0.6377 (1)
Occupancy	0.9825 (4)	0.9826 (3)	0.9825 (3)	0.9825 (3)
Optdigits	0.7747 (3)	0.7769 (3)	0.782 (3)	0.7791 (3)
Pen	0.6455 (3)	0.6445 (3)	0.6468 (2)	0.6403 (4)
Segmentation	0.8999 (1)	0.8993 (1)	0.9016 (1)	0.8991 (1)
Spambase	0.9179 (3)	0.9183 (3)	0.9185 (3)	0.9179 (3)
Transfusion	0.6368 (4)	0.6383 (4)	0.6371 (4)	0.6366 (4)
Twitter	0.7019 (1)	0.7023 (1)	0.7053 (1)	0.7036 (1)
USPS	0.6514 (1)	0.6167 (2)	0.6335 (3)	0.6262 (2)
Vehicle	0.6263 (2)	0.6211 (4)	0.6223 (3)	0.6252 (2)
Mean ALC	0.7741	0.7734	0.7728	0.7729
Mean Rank	2.6957	2.4348	2.8261	2.6522

Chapter6

CONCLUSION

In today's information-rich digital world, supervised learning approaches are used extensively to mine credible patterns of the data and solve real-world challenges. However, these algorithms have a non-trivial limitation: they require labeled training data which is often time-consuming or expensive to obtain.

This dissertation has focused on the development of new efficient pool-based batch-mode active learning approaches that aim to reduce the overall cost of acquiring labeled data by allowing the learner to effectively choose the instances on which it is trained. In this final chapter, I summarize the specific contributions and limitations of this work.

6.1 Summary of Contributions

This thesis has made several contributions to the state of the art in AL. Specifically,

- A new approach for incorporating data density into uncertainty sampling for efficient batch-mode AL. The proposed algorithm in Chapter 3, CSQBF, provides a stochastic sampling procedure that facilitates balancing exploration and exploitation in AL. CSQBF benefits from advantages of Random forest classification model including robustness to noise and handling mixed type data. This

algorithm can be used to reduce overall annotation costs by actively querying the most uncertain instances from different areas of the data. The results have implications for many real-world learning applications.

- Two new AL frameworks with new formulations for density. Two novel algorithms, DMAL and CAAL, with new strategies for characterizing classification uncertainty and data density were introduced. Instead of defining distinct clusters, both algorithms take advantage of a soft clustering approach which enables them to define density as a separate source of uncertainty that is consistent with classification uncertainty. Both methods combine uncertainty and density into unified and coherent scoring measures for AL label query. DMAL algorithm is especially useful for multi-label data as it aggregates class and cluster labels into a single pseudo label. Uncertainty and density in DMAL are calculated based on classification probabilities provided by a supervised model that uses the pseudo label. CAAL on the other hand, formulates density by constructing a separate supervised model, other than the active learner, using cluster assignments as labels to obtain cluster membership probabilities. These probabilities are then used to define a density score which provides an improved overall score for label query. To show the outperformance of DMAL and CAAL, we compared them with CSQBF and other alternatives. Future work in this area includes exploring multi-label datasets. Details of the DMAL and CAAL are discussed in Chapter 4.

- Two new feedback-driven strategies to balance uncertainty and density in AL. In Chapter 5, I presented two approaches to adaptively learn how to make a trade-off between uncertainty and density sampling in AL. These approaches are extensions to the CAAL algorithm in Chapter 4 by using the overall AL

procedure of CAAL, but instead learning how to different elements of CAAL impact it differently at different stages of the AL process. The KF-RB algorithm proposes a restless multi-armed bandit approach where each arm represents a specific trade-off value for combining uncertainty and density scores. RAFO extends KF-RB by embedding separate arms into a continuous action domain. It proposes a time-varying Gaussian process model to learn the relationship between uncertainty-density trade-off and active learner's feedback. The proposed are state of the art for dynamically balancing exploration and exploitation in AL. The proposed methodologies are not suitable for the suggested settings, they can also be applied to other AL algorithms where exploration and exploitation are modeled explicitly using separate modules.

REFERENCES

[1] "Understanding and fighting bullying with machine learning", URL `http://research.cs.wisc.edu/bullying/data.html` (2014).

[2] AISTATS, "AISTATS 2010 active learning challenge", URL `http://www.causality.inf.ethz.ch/activelearning.php` (2010).

[3] Ali, K. M. and M. J. Pazzani, *On the link between error correlation and error reduction in decision tree ensembles* (Citeseer, 1995).

[4] Atkinson, A. C. and L. M. Haines, "14 designs for nonlinear and generalized linear models", Handbook of statistics **13**, 437–475 (1996).

[5] Auer, P., "Using confidence bounds for exploitation-exploration trade-offs", Journal of Machine Learning Research **3**, Nov, 397–422 (2002).

[6] Baram, Y., R. E. Yaniv and K. Luz, "Online choice of active learning algorithms", Journal of Machine Learning Research **5**, Mar, 255–291 (2004).

[7] Bogunovic, I., J. Scarlett and V. Cevher, "Time-varying gaussian process bandit optimization", in "Artificial Intelligence and Statistics", pp. 314–323 (2016).

[8] Borisov, A., E. Tuv and G. Runger, "Active batch learning with stochastic query-by-forest (SQBF)", in "JMLR Workshop on Active Learning and Experimental Design", pp. 59–69 (2011).

[9] Bosch, A., A. Zisserman and X. Munoz, "Image classification using random forests and ferns", in "2007 IEEE 11th international conference on computer vision", pp. 1–8 (Ieee, 2007).

[10] Box, G. E., G. M. Jenkins and G. C. Reinsel, *Time series analysis: forecasting and control*, vol. 734 (John Wiley & Sons, 2011).

[11] Boyd, S. and L. Vandenberghe, *Convex optimization* (Cambridge university press, 2004).

[12] Bradley, A. P., "The use of the area under the roc curve in the evaluation of machine learning algorithms", Pattern recognition **30**, 7, 1145–1159 (1997).

[13] Breiman, L., "Bagging predictors", Machine learning **24**, 2, 123–140 (1996).

[14] Breiman, L. and A. Cutler, "Random Forests", URL `https://www.stat.berkeley.edu/~breiman/RandomForests/` (2003).

[15] Brochu, E., V. M. Cora and N. De Freitas, "A tutorial on bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning", arXiv preprint arXiv:1012.2599 (2010).

[16] Cawley, G., "Some baseline methods for the active learning challenge", in "Thirteenth International Conference on Artificial Intelligence and Statistics. Workshop and Active Learning Competition", (2010).

[17] Cebron, N. and M. R. Berthold, "Active learning for object classification: from exploration to exploitation", Data Mining and Knowledge Discovery **18**, 2, 283–299 (2009).

[18] Chapelle, O. and L. Li, "An empirical evaluation of thompson sampling", in "Advances in neural information processing systems", pp. 2249–2257 (2011).

[19] Chapelle, O., B. Scholkopf and A. Zien, "Semi-supervised learning (chapelle, o. et al., eds.; 2006)[book reviews]", IEEE Transactions on Neural Networks **20**, 3, 542–542 (2009).

[20] Chapelle, O., J. Weston and B. Schölkopf, "Cluster kernels for semi-supervised learning", in "Advances in Neural Information Processing Systems", pp. 585–592 (2002).

[21] Chapelle, O., J. Weston and B. Schölkopf, "Cluster kernels for semi-supervised learning", in "Advances in neural information processing systems", pp. 601–608 (2003).

[22] Chattopadhyay, R., Z. Wang, W. Fan, I. Davidson, S. Panchanathan and J. Ye, "Batch mode active sampling based on marginal probability distribution matching", ACM Transactions on Knowledge Discovery from Data (TKDD) **7**, 3, 13 (2013).

[23] Cheng, Y., Z. Chen, L. Liu, J. Wang, A. Agrawal and A. Choudhary, "Feedback-driven multiclass active learning for data streams", in "Proceedings of the 22nd ACM international conference on Conference on information & knowledge management", pp. 1311–1320 (ACM, 2013).

[24] Cohn, D., L. Atlas and R. Ladner, "Improving generalization with active learning", Machine learning **15**, 2, 201–221 (1994).

[25] Cortes, C. and V. Vapnik, "Support-vector networks", Machine learning **20**, 3, 273–297 (1995).

[26] Cutler, D. R., T. C. Edwards, K. H. Beard, A. Cutler, K. T. Hess, J. Gibson and J. J. Lawler, "Random forests for classification in ecology", Ecology **88**, 11, 2783–2792 (2007).

[27] Dasgupta, S., "Two faces of active learning", Theoretical computer science **412**, 19, 1767–1781 (2011).

[28] Dasgupta, S. and D. Hsu, "Hierarchical sampling for active learning", in "Proceedings of the 25th International Conference on Machine Learning", pp. 208–215 (ACM, 2008).

[29] de Mello, C. E. R., *Active Learning: an unbiased approach*, Ph.D. thesis, Ecole Centrale Paris (2013).

[30] DeBarr, D. and H. Wechsler, "Spam detection using clustering, random forests, and active learning", in "Sixth Conference on Email and Anti-Spam.", (Citeseer, 2009).

[31] Delbos, F. and J. C. Gilbert, *Global linear convergence of an augmented Lagrangian algorithm for solving convex quadratic optimization problems*, Ph.D. thesis, INRIA (2003).

[32] Dempster, A. P., N. M. Laird and D. B. Rubin, "Maximum likelihood from incomplete data via the em algorithm", Journal of the Royal Statistical Society: Series B (Methodological) **39**, 1, 1–22 (1977).

[33] Dheeru, D. and E. Karra Taniskidou, "UCI machine learning repository", URL `http://archive.ics.uci.edu/ml` (2017).

[34] Díaz-Uriarte, R. and S. A. De Andres, "Gene selection and classification of microarray data using random forest", BMC bioinformatics **7**, 1, 3 (2006).

[35] Domingos, P., "A few useful things to know about machine learning", Communications of the ACM **55**, 10, 78–87 (2012).

[36] Donmez, P., J. G. Carbonell and P. N. Bennett, "Dual strategy active learning", in "European Conference on Machine Learning", pp. 116–127 (Springer, 2007).

[37] Durbin, J. and S. J. Koopman, *Time series analysis by state space methods* (Oxford university press, 2012).

[38] Ebert, S., M. Fritz and B. Schiele, "Ralf: A reinforced active learning formulation for object class recognition", in "2012 IEEE Conference on Computer Vision and Pattern Recognition", pp. 3626–3633 (IEEE, 2012).

[39] Elahi, M., F. Ricci and N. Rubens, "A survey of active learning in collaborative filtering recommender systems", Computer Science Review **20**, 29–50 (2016).

[40] Freund, Y. 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).

[41] Freund, Y., H. S. Seung, E. Shamir and N. Tishby, "Selective sampling using the query by committee algorithm", Machine Learning **28**, 2-3, 133–168 (1997).

[42] Friedman, J., T. Hastie and R. Tibshirani, *The elements of statistical learning*, vol. 1 (Springer series in statistics New York, 2001).

[43] Ganti, R. and A. G. Gray, "Building bridges: Viewing active learning from the multi-armed bandit lens", arXiv preprint arXiv:1309.6830 (2013).

[44] Geman, S., E. Bienenstock and R. Doursat, "Neural networks and the bias/variance dilemma", Neural computation **4**, 1, 1–58 (1992).

[45] Ghazal Shams, E. d. C. and G. Runger, "A dual model agnostic strategy to explore representativeness and informativeness in active learning", Manuscript submitted for publication (2020).

[46] Gini, C., "Variabilità e mutabilità", Reprinted in Memorie di metodologica statistica (Ed. Pizetti E, Salvemini, T). Rome: Libreria Eredi Virgilio Veschi (1912).

[47] Godbole, S. and S. Sarawagi, "Discriminative methods for multi-labeled classification", in "Pacific-Asia conference on knowledge discovery and data mining", pp. 22–30 (Springer, 2004).

[48] Granmo, O.-C., "The bayesian learning automaton—empirical evaluation with two-armed bernoulli bandit problems", in "International Conference on Innovative Techniques and Applications of Artificial Intelligence", pp. 235–248 (Springer, 2008).

[49] Granmo, O.-C. and S. Berg, "Solving non-stationary bandit problems by random sampling from sibling kalman filters", in "International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems", pp. 199–208 (Springer, 2010).

[50] Gu, Y., D. Zydek and Z. Jin, "Active learning based on random forest and its application to terrain classification", in "Progress in Systems Engineering", pp. 273–278 (Springer, 2015).

[51] Gupta, N., O.-C. Granmo and A. Agrawala, "Thompson sampling for dynamic multi-armed bandits", in "2011 10th International Conference on Machine Learning and Applications and Workshops", vol. 1, pp. 484–489 (IEEE, 2011).

[52] Guyon, I., G. Cawley, G. Dror, V. Lemaire and A. Statnikov, *Active Learning Challenge: Challenges in Machine Learning, Volumen 6* (Microtome publishing, 2012).

[53] Guyon, I., G. C. Cawley, G. Dror and V. Lemaire, "Results of the active learning challenge.", in "JMLR Workshop on Active Learning and Experimental Design", pp. 19–45 (2011).

[54] Hartland, C., N. Baskiotis, S. Gelly, M. Sebag and O. Teytaud, "Change point detection and meta-bandits for online learning in dynamic environments", (2007).

[55] Harvey, A. C., *Forecasting, structural time series models and the Kalman filter* (Cambridge university press, 1990).

[56] Hoi, S. C., R. Jin, J. Zhu and M. R. Lyu, "Batch mode active learning and its application to medical image classification", in "Proceedings of the 23rd International Conference on Machine Learning", pp. 417–424 (ACM, 2006).

[57] Hoi, S. C., R. Jin, J. Zhu and M. R. Lyu, "Semisupervised svm batch mode active learning with applications to image retrieval", ACM Transactions on Information Systems (TOIS) **27**, 3, 16 (2009).

[58] Hossain, H. M. S., N. Roy and M. A. A. H. Khan, "Active learning enabled activity recognition", in "2016 IEEE International Conference on Pervasive Computing and Communications (PerCom)", pp. 1–9 (2016).

[59] Hsu, D. J., *Algorithms for active learning*, Ph.D. thesis, UC San Diego (2010).

[60] Hsu, W.-N. and H.-T. Lin, "Active learning by learning", in "Twenty-Ninth AAAI conference on artificial intelligence", (2015).

[61] Hu, L., S. Lu and X. Wang, "A new and informative active learning approach for support vector machine", Information Sciences **244**, 142–160 (2013).

[62] Huang, S.-J., R. Jin and Z.-H. Zhou, "Active learning by querying informative and representative examples", in "Advances in neural information processing systems", pp. 892–900 (2010).

[63] Huang, S. J., R. Jin and Z. H. Zhou, "Active learning by querying informative and representative examples", IEEE Transactions on Pattern Analysis and Machine Intelligence **36**, 10, 1936–1949 (2014).

[64] Jain, P. and A. Kapoor, "Active learning for large multi-class problems", in "Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on", pp. 762–769 (IEEE, 2009).

[65] Jegelka, S., A. Kapoor and E. Horvitz, "An interactive approach to solving correspondence problems", International journal of computer vision **108**, 1-2, 49–58 (2014).

[66] Joshi, A. J., F. Porikli and N. Papanikolopoulos, "Multi-class active learning for image classification", in "Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on", pp. 2372–2379 (IEEE, 2009).

[67] Jung, Y. H., M. Abeille and A. Tewari, "Thompson sampling in non-episodic restless bandits", arXiv preprint arXiv:1910.05654 (2019).

[68] Kalman, R. E., "A new approach to linear filtering and prediction problems", Journal of basic Engineering **82**, 1, 35–45 (1960).

[69] Kandasamy, K., J. Schneider and B. Póczos, "High dimensional bayesian optimisation and bandits via additive models", in "International Conference on Machine Learning", pp. 295–304 (2015).

[70] Kapoor, A., K. Grauman, R. Urtasun and T. Darrell, "Gaussian processes for object categorization", International journal of computer vision **88**, 2, 169–188 (2010).

[71] Kaufman, L. and P. J. Rousseeuw, "Partitioning around medoids (program pam)", Finding Groups in Data: an Introduction to Cluster Analysis pp. 68–125 (1990).

[72] Kaufman, L. and P. J. Rousseeuw, *Finding groups in data: an introduction to cluster analysis*, vol. 344 (John Wiley & Sons, 2009).

[73] Kearns, M. and D. Koller, "Efficient reinforcement learning in factored mdps", in "IJCAI", vol. 16, pp. 740–747 (1999).

[74] Kee, S., E. del Castillo and G. Runger, "Query-by-committee improvement with diversity and density in batch active learning", Information Sciences **454**, 401–418 (2018).

[75] King, R. D., 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 (2004).

[76] Krause, A. and C. S. Ong, "Contextual gaussian process bandit optimization", in "Advances in neural information processing systems", pp. 2447–2455 (2011).

[77] Kullback, S., *Information theory and statistics* (Courier Corporation, 1997).

[78] Kullback, S. and R. A. Leibler, "On information and sufficiency", The annals of mathematical statistics **22**, 1, 79–86 (1951).

[79] LeCun, Y., "The mnist database of handwritten digits", http://yann. lecun. com/exdb/mnist/ (1998).

[80] Leskovec, J., A. Rajaraman and J. D. Ullman, *Mining of massive datasets* (Cambridge University Press, 2014).

[81] Lewis, D. D. and W. A. Gale, "A sequential algorithm for training text classifiers", in "Proceedings of the 17th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval", pp. 3–12 (Springer-Verlag New York, Inc., 1994).

[82] Li, X. and Y. Guo, "Adaptive active learning for image classification", in "Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition", pp. 859–866 (2013).

[83] Li, X., L. Wang and E. Sung, "Multilabel svm active learning for image classification", in "Image Processing, 2004. ICIP'04. 2004 International Conference on", vol. 4, pp. 2207–2210 (IEEE, 2004).

[84] Lichman, M., "UCI machine learning repository", URL `http://archive.ics.uci.edu/ml` (2013).

[85] Linderoth, M., K. Soltesz, A. Robertsson and R. Johansson, "Initialization of the kalman filter without assumptions on the initial state", in "2011 IEEE International Conference on Robotics and Automation", pp. 4992–4997 (IEEE, 2011).

[86] Ling, C. K., K. H. Low and P. Jaillet, "Gaussian process planning with lipschitz continuous reward functions: Towards unifying bayesian optimization, active learning, and beyond", in "Thirtieth AAAI Conference on Artificial Intelligence", (2016).

[87] Liu, R., Y. Wang, T. Baba, D. Masumoto and S. Nagata, "Svm-based active feedback in image retrieval using clustering and unlabeled data", Pattern Recognition **41**, 8, 2645–2655 (2008).

[88] Luaces, O., J. Díez, J. Barranquero, J. J. del Coz and A. Bahamonde, "Binary relevance efficacy for multilabel classification", Progress in Artificial Intelligence **1**, 4, 303–313 (2012).

[89] Mac Aodha, O., N. D. Campbell, J. Kautz and G. J. Brostow, "Hierarchical subquery evaluation for active learning on a graph", in "Proceedings of the IEEE conference on computer vision and pattern recognition", pp. 564–571 (2014).

[90] MacKay, D. J., "Information-based objective functions for active data selection", Neural computation **4**, 4, 590–604 (1992).

[91] MacQueen, J. *et al.*, "Some methods for classification and analysis of multivariate observations", in "Proceedings of the fifth Berkeley symposium on mathematical statistics and probability", vol. 1, pp. 281–297 (Oakland, CA, USA, 1967).

[92] Mantripragada, K., J. A. Quintanilha and M. A. Giannotti, "Active learning classification and change detection on multispectral images", in "2014 12th IEEE International Conference on Industrial Informatics (INDIN)", pp. 26–30 (2014).

[93] Marchant, R., F. Ramos, S. Sanner *et al.*, "Sequential bayesian optimisation for spatial-temporal monitoring.", in "UAI", pp. 553–562 (2014).

[94] McCallumzy, A. K. and K. Nigamy, "Employing em and pool-based active learning for text classification", in "Proc. International Conference on Machine Learning (ICML)", pp. 359–367 (Citeseer, 1998).

[95] Mellor, J., *Decision Making Using Thompson Sampling*, Ph.D. thesis, The University of Manchester (United Kingdom) (2014).

[96] Meshram, R., A. Gopalan and D. Manjunath, "Restless bandits that hide their hand and recommendation systems", in "2017 9th International Conference on Communication Systems and Networks (COMSNETS)", pp. 206–213 (IEEE, 2017).

[97] Mockus, J., *Bayesian approach to global optimization: theory and applications*, vol. 37 (Springer Science & Business Media, 2012).

[98] Mockus, J., V. Tiesis and A. Zilinskas, "Toward global optimization, volume 2, chapter bayesian methods for seeking the extremum", (1978).

[99] Montgomery, D. C., E. A. Peck and G. G. Vining, *Introduction to linear regression analysis*, vol. 821 (John Wiley & Sons, 2012).

[100] Nguyen, H. T. and A. Smeulders, "Active learning using pre-clustering", in "Proceedings of the twenty-first international conference on Machine learning", p. 79 (ACM, 2004).

[101] Opitz, D. and R. Maclin, "Popular ensemble methods: An empirical study", Journal of artificial intelligence research **11**, 169–198 (1999).

[102] Osugi, T., D. Kim and S. Scott, "Balancing exploration and exploitation: A new algorithm for active machine learning", in "Fifth IEEE International Conference on Data Mining (ICDM'05)", pp. 8–pp (IEEE, 2005).

[103] Özgür, A., L. Özgür and T. Güngör, "Text categorization with class-based and corpus-based keyword selection", in "International Symposium on Computer and Information Sciences", pp. 606–615 (Springer, 2005).

[104] Pang, K., M. Dong, Y. Wu and T. M. Hospedales, "Dynamic ensemble active learning: A non-stationary bandit with expert advice", in "2018 24th International Conference on Pattern Recognition (ICPR)", pp. 2269–2276 (IEEE, 2018).

[105] Parsons, L., E. Haque and H. Liu, "Subspace clustering for high dimensional data: a review", ACM SIGKDD Explorations Newsletter **6**, 1, 90–105 (2004).

[106] Patra, S. and L. Bruzzone, "A cluster-assumption based batch mode active learning technique", Pattern Recognition Letters **33**, 9, 1042–1048 (2012).

[107] Pauwels, E., C. Lajaunie and J.-P. Vert, "A bayesian active learning strategy for sequential experimental design in systems biology", BMC Systems Biology **8**, 1, 102 (2014).

[108] Pfingsten, T., "Bayesian active learning for sensitivity analysis", in "European Conference on Machine Learning", pp. 353–364 (Springer, 2006).

[109] Platt, J. *et al.*, "Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods", Advances in large margin classifiers **10**, 3, 61–74 (1999).

[110] Raghavan, H., O. Madani and R. Jones, "Active learning with feedback on features and instances", Journal of Machine Learning Research **7**, Aug, 1655–1686 (2006).

[111] Read, J., B. Pfahringer and G. Holmes, "Multi-label classification using ensembles of pruned sets", in "Data Mining, 2008. ICDM'08. Eighth IEEE International Conference on", pp. 995–1000 (IEEE, 2008).

[112] Reyes, O., A. H. Altalhi and S. Ventura, "Statistical comparisons of active learning strategies over multiple datasets", Knowledge-Based Systems **145**, 274–288 (2018).

[113] Rigollet, P., "Generalization error bounds in semi-supervised classification under the cluster assumption", Journal of Machine Learning Research **8**, Jul, 1369–1392 (2007).

[114] Rokach, L., A. Schclar and E. Itach, "Ensemble methods for multi-label classification", Expert Systems with Applications **41**, 16, 7507–7523 (2014).

[115] Roy, N. and A. McCallum, "Toward optimal active learning through monte carlo estimation of error reduction", ICML, Williamstown pp. 441–448 (2001).

[116] Sagi, O. and L. Rokach, "Ensemble learning: A survey", Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery **8**, 4, e1249 (2018).

[117] Schapire, R. E. and Y. Singer, "Boostexter: A boosting-based system for text categorization", Machine learning **39**, 2-3, 135–168 (2000).

[118] Schein, A. I. and L. H. Ungar, "Active learning for logistic regression: an evaluation", Machine Learning **68**, 3, 235–265 (2007).

[119] Schohn, G. and D. Cohn, "Less is more: Active learning with support vector machines", in "Proceedings of the 17th International Conference on Machine Learning", pp. 839–846 (2000).

[120] Scott, S. L., "A modern bayesian look at the multi-armed bandit", Applied Stochastic Models in Business and Industry **26**, 6, 639–658 (2010).

[121] Settles, B., *Curious machines: Active learning with structured instances*, Ph.D. thesis, University of Wisconsin–Madison (2008).

[122] Settles, B., "Active learning literature survey", Tech. Rep. 1648, Department of Computer Sciences, University of Wisconsin–Madison (2009).

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

[124] Settles, B. and M. Craven, "An analysis of active learning strategies for sequence labeling tasks", in "Proceedings of the conference on empirical methods in natural language processing", pp. 1070–1079 (Association for Computational Linguistics, 2008).

[125] Seung, H. S., M. Opper and H. Sompolinsky, "Query by committee", in "Proceedings of the fifth annual workshop on Computational learning theory", pp. 287–294 (ACM, 1992).

[126] Shahriari, B., K. Swersky, Z. Wang, R. P. Adams and N. De Freitas, "Taking the human out of the loop: A review of bayesian optimization", Proceedings of the IEEE **104**, 1, 148–175 (2015).

[127] Shannon, C. E., "A mathematical theory of communication", Bell system technical journal **27**, 3, 379–423 (1948).

[128] Sharma, M. and M. Bilgic, "Evidence-based uncertainty sampling for active learning", Data Mining and Knowledge Discovery **31**, 1, 164–202 (2017).

[129] Shi, T. and S. Horvath, "Unsupervised learning with random forest predictors", Journal of Computational and Graphical Statistics **15**, 1, 118–138 (2006).

[130] Shi, T. and S. Horvath, "Unsupervised learning with random forest predictors", Journal of Computational and Graphical Statistics **15**, 1, 118–138 (2006).

[131] Shuyang, Z., T. Heittola and T. Virtanen, "Active learning for sound event classification by clustering unlabeled data", in "Acoustics, Speech and Signal Processing (ICASSP), 2017 IEEE International Conference on", pp. 751–755 (IEEE, 2017).

[132] Sivaraman, S. and M. M. Trivedi, "A general active-learning framework for on-road vehicle recognition and tracking", IEEE Transactions on Intelligent Transportation Systems **11**, 2, 267–276 (2010).

[133] Sokolova, M. and G. Lapalme, "A systematic analysis of performance measures for classification tasks", Information Processing & Management **45**, 4, 427–437 (2009).

[134] Speekenbrink, M. and E. Konstantinidis, "Uncertainty and exploration in a restless bandit problem", Topics in cognitive science **7**, 2, 351–367 (2015).

[135] Srinivas, N., A. Krause, S. M. Kakade and M. W. Seeger, "Information-theoretic regret bounds for gaussian process optimization in the bandit setting", IEEE Transactions on Information Theory **58**, 5, 3250–3265 (2012).

[136] Svetnik, V., A. Liaw, C. Tong, J. C. Culberson, R. P. Sheridan and B. P. Feuston, "Random forest: a classification and regression tool for compound classification and qsar modeling", Journal of chemical information and computer sciences **43**, 6, 1947–1958 (2003).

[137] Tang, M., X. Luo and S. Roukos, "Active learning for statistical natural language parsing", in "Proceedings of the 40th Annual Meeting on Association for Computational Linguistics", pp. 120–127 (Association for Computational Linguistics, 2002).

[138] Thompson, W. R., "On the likelihood that one unknown probability exceeds another in view of the evidence of two samples", Biometrika **25**, 3/4, 285–294 (1933).

[139] Tong, S., *Active learning: theory and applications*, vol. 1 (Stanford University USA, 2001).

[140] Tong, S. and D. Koller, "Support vector machine active learning with applications to text classification", Journal of machine learning research **2**, Nov, 45–66 (2001).

[141] Tsoumakas, G. and I. Katakis, "Multi-label classification: An overview", International Journal of Data Warehousing and Mining (IJDWM) **3**, 3, 1–13 (2007).

[142] Tuia, D., F. Ratle, F. Pacifici, M. F. Kanevski and W. J. Emery, "Active learning methods for remote sensing image classification", IEEE Transactions on Geoscience and Remote Sensing **47**, 7, 2218–2232 (2009).

[143] Tumer, K. and J. Ghosh, "Error correlation and error reduction in ensemble classifiers", Connection science **8**, 3-4, 385–404 (1996).

[144] Vapnik, V. N. and V. Vapnik, *Statistical learning theory* (Wiley New York, 1998).

[145] Vasisht, D., A. Damianou, M. Varma and A. Kapoor, "Active learning for sparse bayesian multilabel classification", in "Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining", pp. 472–481 (ACM, 2014).

[146] Velentzas, G., C. Tzafestas and M. Khamassi, "Bio-inspired meta-learning for active exploration during non-stationary multi-armed bandit tasks", in "2017 Intelligent Systems Conference (IntelliSys)", pp. 661–669 (IEEE, 2017).

[147] Viappiani, P., "Thompson sampling for bayesian bandits with resets", in "International Conference on Algorithmic DecisionTheory", pp. 399–410 (Springer, 2013).

[148] Wang, J., E. Park and Y.-c. I. Chang, "Active learning procedure via sequential experimental design and uncertainty sampling", arXiv preprint arXiv:1406.4676 (2014).

[149] Whittle, P., "Restless bandits: Activity allocation in a changing world", Journal of applied probability **25**, A, 287–298 (1988).

[150] Xiong, S., Y. Pei, R. Rosales and X. Z. Fern, "Active learning from relative comparisons", IEEE Transactions on Knowledge and Data Engineering **27**, 12, 3166–3175 (2015).

[151] Xu, J.-M., K.-S. Jun, X. Zhu and A. Bellmore, "Learning from bullying traces in social media", in "Proceedings of the 2012 conference of the North American chapter of the association for computational linguistics: Human language technologies", pp. 656–666 (Association for Computational Linguistics, 2012).

[152] Xu, R. and D. Wunsch, "Survey of clustering algorithms", IEEE Transactions on neural networks **16**, 3, 645–678 (2005).

[153] Xu, Z., R. Akella and Y. Zhang, "Incorporating diversity and density in active learning for relevance feedback", in "European Conference on Information Retrieval", pp. 246–257 (Springer, 2007).

[154] Xu, Z., K. Yu, V. Tresp, X. Xu and J. Wang, "Representative sampling for text classification using support vector machines", in "European Conference on Information Retrieval", pp. 393–407 (Springer, 2003).

[155] Yang, J. *et al.*, "Automatically labeling video data using multi-class active learning", in "Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on", pp. 516–523 (IEEE, 2003).

[156] Yang, Y. and M. Loog, "A variance maximization criterion for active learning", Pattern Recognition **78**, 358–370 (2018).

[157] Yang, Y., Z. Ma, F. Nie, X. Chang and A. G. Hauptmann, "Multi-class active learning by uncertainty sampling with diversity maximization", International Journal of Computer Vision **113**, 2, 113–127 (2015).

[158] Zhu, J., H. Wang, B. K. Tsou and M. Ma, "Active learning with sampling by uncertainty and density for data annotations", IEEE Transactions on audio, speech, and language processing **18**, 6, 1323–1331 (2009).

[159] Zhu, J., H. Wang, B. K. Tsou and M. Y. Ma, "Active learning with sampling by uncertainty and density for data annotations.", IEEE Trans. Audio, Speech & Language Processing **18**, 6, 1323–1331 (2010).

[160] Zhu, X., "Semi-supervised learning literature survey", (2005).

[161] Zhu, X., J. Lafferty and Z. Ghahramani, "Combining active learning and semi-supervised learning using gaussian fields and harmonic functions", in "ICML 2003 workshop on the continuum from labeled to unlabeled data in machine learning and data mining", vol. 3 (2003).