Active Learning with Combinatorial Coverage

Sai Prathyush Katragadda

Thesis submitted to the Faculty of the
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

Master of Science
in
Industrial and Systems Engineering

Peter A Beling, Chair
Laura J Freeman
Manish Bansal

07/15/2022
Blacksburg, Virginia

Keywords: Combinatorial Coverage, Active Learning, Combinatorial Interaction Testing, Classification

Copyright 2022, Sai Prathyush Katragadda

Active Learning with Combinatorial Coverage

Sai Prathyush Katragadda

ABSTRACT

Active learning is a practical field of machine learning as labeling data or determining which data to label can be a time consuming and inefficient task. Active learning automates the process of selecting which data to label, but current methods are heavily model reliant. This has led to the inability of sampled data to be transferred to new models as well as issues with sampling bias. Both issues are of crucial concern in machine learning deployment. We propose active learning methods utilizing Combinatorial Coverage to overcome these issues. The proposed methods are data-centric, and through our experiments we show that the inclusion of coverage in active learning leads to sampling data that tends to be the best in transferring to different models and has a competitive sampling bias compared to benchmark methods.

Active Learning with Combinatorial Coverage

Sai Prathyush Katragadda

GENERAL AUDIENCE ABSTRACT

Machine learning (ML) models are being used frequently in a variety of applications. For the model to be able to learn, data is required. Processing this data is often one of the most, if not the most, time consuming aspects of utilizing ML. One especially burdensome aspect of data processing is data labeling, or determining what each data point corresponds to in terms of real world class. For example, determining if a data point that is an image contains a plane or bird. This way the ML model can learn from the data.

Active learning is a sub-field of machine learning which aims to ease this burden by allowing the model to select which data would be most beneficial to label, so that the entirety of the dataset does not need to be labeled. The issue with current active learning methods is that they are highly model dependent. In machine learning deployment the model being used may change while data stays the same, so this model dependency can cause for data points we label with respect to one model to not be ideal for another model. This model dependency has led to sampling bias issues as well; points which are chosen to be labeled may all be similar or not representative of all data resulting in the ML model not being as knowledgeable as possible.

Relevant work has focused on the sampling bias issue, and several methods have been proposed to combat this issue. Few of the methods are applicable to any type of ML model though. The issue of sampled points not generalizing to different models has been studied but no solutions have been proposed.

In this work we present active learning methods using Combinatorial Coverage. Combinatorial Coverage is a statistical technique from the field of Design of Experiments, and has

commonly been used to design test sets. The extension of Combinatorial Coverage to ML is newer, and provides a way to focus on the data. We show that this data focused approach to active learning achieves a better performance when the sampled data is used for a different model and that it achieves a competitive sampling bias compared to benchmark methods.

Acknowledgments

I am immensely appreciative of Dr. Tyler Cody and Dr. Peter Beling for their patience, support, and guidance throughout this research. Without them this work would not have been possible. I would like to extend my sincerest gratitude to Dr. Laura Freeman, who gave me the opportunity, support, and valuable input necessary to work on this research. I would also like to thank Dr. Manish Bansal for the constructive feedback and support in both the classroom and research work. Without the aid of all the professors in my masters degree committee I would not have finished this work. Last but not least, I would like to thank my parents and brother; Aruna, Shanker, and Aathish, for their support in education and life in general.

Contents

List of Figures

List of Tables

List of Abbreviations

CC Combinatorial Coverage

CIT Combinatorial Interaction Testing

QBC Query By Committee

SDCC Set Difference Combinatorial Coverage

USWCD Uncertainty Sampling Weighted by Coverage Density

Chapter 1

Introduction

This chapter provides an introduction and outline for this thesis, with section 1.1 discussing the motivation behind the research and section 1.2 highlighting the objectives and expected outcomes. Section 1.3 provides a road map for the rest of this paper.

1.1 Motivations

Data preprocessing, which involves data labeling, is often the most expensive aspect of machine learning deployment[2]. Relevant to this is active learning, which is a sub-field of machine learning concerned with models selecting which data points would be most beneficial to train on[3]. This is especially applicable when the model has access to a large pool of unlabeled data, but there is a restriction on the number of data points that can be labeled due to budget or time constraints. Therefore, active learning proposes a solution to part of the most expensive aspect of machine learning deployment.

Active learning has found success in many applications including image segmentation [4], sequence labeling [5], medical image classification [6], and cybersecurity [7]. Yet, active learning methods are heavily model dependent, thus data points sampled using one model may not be effective for the training of other models [8][9]. As pointed out by Paleyes and Urma [10], machine learning deployment is often an iterative process in deciding which model to utilize. So, active learning can provide a great benefit in the data management or

processing phase, but for the use of active learning to be practical, the sampled data points should also be effective in training other models.

Current active learning methods are successful in particular applications, in other words, a specific method will fare well for some combination of machine learning model and dataset[8][9]. The issue is that finding which active learning method samples the most effective data for a model may defeat the purpose of applying active learning due to the required investment of resources. This is especially true in deployment scenarios where model type is being constantly updated.

The cause of this issue is the model dependency of active learning methods; points to be labeled are regarded as beneficial to a specific model. However, these labeled points may not be the ideal datapoints for training a different model, and moreover these points may be from a particular area of feature space which can cause for sampling bias. Several methods have been proposed in the literature to combat the sampling bias issue, few of which are generalizable to any model and none of which are model independent. To our knowledge, no data-centric active learning methods have been proposed to sample data so that other models are just as applicable without resampling.

Active learning methods optimize data labeling for a given model, but struggle to sample data which is effective for training other models. This issue is closely related to the sampling bias issue, which results from the model dependency of active learning methods. In this work we propose an active learning approach utilizing Combinatorial Coverage (CC) which is data-centric, can generalize to any model, samples data which can be transferred to new models, and achieves a lower sampling bias than that of random sampling. We contribute three active learning methods:

- coverage density sampling,

- informative coverage density sampling, and
- uncertainty sampling weighted by coverage density

. While Combinatorial Interaction Testing(CIT) and CC are not widespread in machine learning, several applications have proven successful. We leverage these ideas to develop the proposed methods, and present their competitive performance in terms of accuracy of the trained model and different models as well as the advantages in sampling bias.

1.2 Research Objectives

As current active learning methods do not sample data with the required flexibility for machine learning deployment, this research aims to develop active learning methods with the following properties:

1. competitive with current methods when sampled data is used to train the model in the active leanring loop
2. highly data dependent so that different models perform well with the sampled data
3. reduced sampling bias, as current generalizable methods tend to suffer from sampling bias

1.3 Outline

This thesis adheres to the following format. The next section is a literature review on active learning and relevant methods, as well as on CIT. The section after provides intuitive and information theoretic reasoning as to why CC should be applicable to active learning.

The section afterwards provides the proposed method and defines several variations on the implementation of CC in active learning, then results are shown in the experimental results section. The final section discusses the methods and results of this thesis as well as future areas of research.

Chapter 2

Literature Review

The literature review is split into two sections; the first focuses on active learning, whereas the second focuses on CIT as well as its application in machine learning.

2.1 Active Learning Literature Review

Active learning is a field of machine learning that is applicable when there exists a large pool of data which is not labeled. The active learning portion of the machine learning model will select a datapoint or group of data points to send to an all knowing oracle which can provide labels for the specified data. The model can then be trained on this labeled data.

Active learning methods can be divided into three groups; membership query synthesis, stream based selective sampling, and pool based sampling [3]. In membership query synthesis the model can arbitrarily select data points to label. Stream based selective sampling involves the model receiving a stream of data points and deciding whether each should be labeled one at a time. Pool based sampling, which is the focus of this work, involves the learner drawing a set of unlabeled samples to label from the entire pool of unlabeled samples available. Several popular and generalizable query strategies exist for active learning. Uncertainty sampling selects the data point the model is currently most uncertain about [11]. Query by Committee (QBC) selects points to label as those which a committee of classifiers most disagree on or are on average most uncertain about [12].

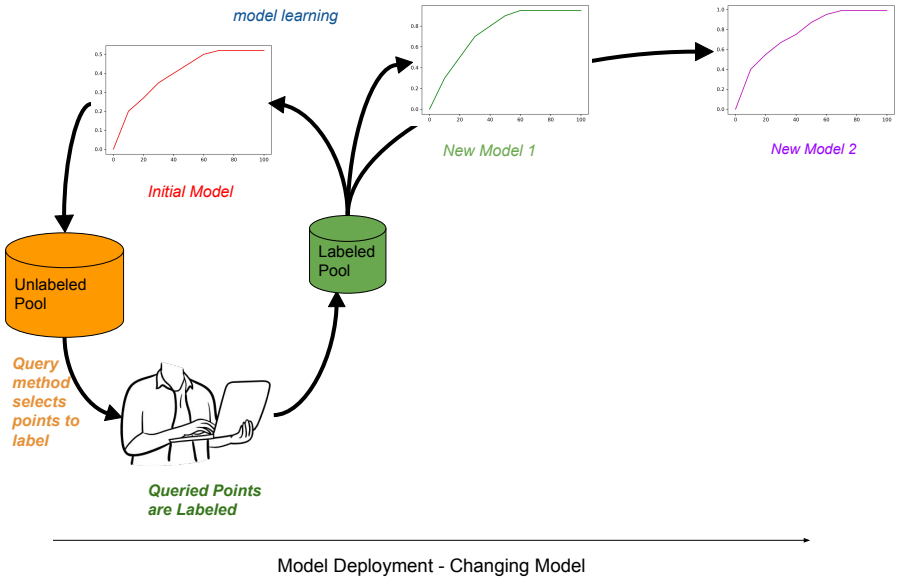

Figure 2.1: Ideal Active Learning in ML Deployment

The dependency of active learning on the model being used has lead to issues in data transferability and sampling bias. Solutions to transferability have not been proposed but there has been work done to look into the specifics of how well the data selected by popular methods transfers to other models. Lowell et al. find that, in general classification problems, transferability of data sampled using uncertainty sampling is not guaranteed [8]. Tomanek and Morik draw a similar conclusion for general classification, but they do find that data sampled using uncertainty sampling is transferable to other models for named entity recognition tasks [9]. Baldridge and Miles share a similar finding which is that data sampled using uncertainty methods and a particular model does not out-perform data sampled randomly when a different model is trained with the sampled data [13]. Pardakhti et al. also reference the inability of active learning methods to sample data which is effective for different models, but their work focuses on finding the optimal hyper parameters for a model given a data set and active learning method [14]. The process of sampling data with respect to one model and using it for other models is shown in figure 2.1.

Sampling bias is also due to the sampled data points being dependent on the model being used. For example, if the uncertainty sampling method is utilized the model will tend to draw those points nearest the decision boundary, regardless of how this affects the distribution of classes that are being sampled. There are several approaches to the problem in the literature. To improve the sampling bias of any baseline sampling method Settles and Craven propose an Information Density method which is computationally expensive for large pools of unlabeled data [5]. Dasgupta and Hsu propose a hierarchical sampling structure for active learning with the intention of mitigating this sampling bias by treating the entire unlabeled dataset as a single cluster and breaking down the dataset into smaller clusters to sample from, as more labels are acquired [15]. The drawback to this method is that as the data has a higher dimensionality the hierarchical clustering will take more samples to determine the lower level clusters where the mitigation of sampling bias will occur, and for data that is not structured for clustering it would be ineffective. Krishnan et al propose an active learning method to combat sampling bias and improve robustness utilzing contrastive learning and clustering, but again determining the clusters may require the labeling of many samples [16].

Beygelzimer et al propose a method using importance weighting which also proves effective but has been designed specifically for binary classifiers [17]. Elhamifar et al propose a batch mode model agnostic sampling method which utilizes both uncertainty and diversity, this method is still prone to sampling outliers since outliers may have high entropy and be distant from the rest of the data [18]. Sener and Savarese propose a core set approach to active learning, where they claim that the difference in average loss over all samples and average loss over selected subset of samples depends only on the radius [19]. But their use of the Euclidean metric does not transfer to high dimensional spaces. Agarwal et al extend this by proposing a core set method based on KL divergence [20]. Both of these methods are only applicable to convolutional neural networks. Liu et al propose a influence based

selection method, but the data point that has most influence on the gradient of the models' loss function may not necessarily improve the performance of the model the most, and could be an outlier [21].

The benchmark methods used in this study are Random Sampling, Uncertainty Sampling, QBC, and Information Density. Random Sampling assumes a uniform distribution over all data points, and thus selects any datapoint to query from the query set with equal probability. We use the entropy [22] based Uncertainty Sampling formulation, which may be considered the most popular approach [3]. Entropy is defined as $-\sum_{y \in Y} p(y)log(p(y))$, where y is a class and Y is all classes. The model trained on currently labeled data is tested on the query set to determine the probability of each query point belonging to each class. These probabilities are then used for the entropy calculation, and those points which the learner has the highest entropy on can be considered as those which the model is most uncertain about, and should provide the greatest benefit when labeled and used for training.

The QBC formulation we use also uses entropy to determine which data point the committee of classifiers is on average most uncertain about, with the committee being comprised of three classifiers: Random Forest , K-Nearest Neighbors, and Logistic Regression. Information Density sampling, as presented by Settles and Craven [5], weighs an original informativeness measure by a similarity metric. In this experiment we weigh uncertainty sampling as previously defined by the inverse of the cardinality of the unlabeled set multiplied by the cosine similarity of the query point in question to all other points. The equation for cosine similarity is presented in equation 2.1 where $\boldsymbol{x}$ represents all data points in both training and query sets.

$$sim(\boldsymbol{x}, x_i) = \sum_{x \in \boldsymbol{x}} \frac{x \cdot x_i}{|x|\,|x_i|} \tag{2.1}$$

2.2 CIT and extension to Machine Learning

CIT stems from covering arrays, ultimately derived from the statistical field of design of experiments. CIT has been applied to several fields but has found a plethora of success in software testing. The application of CIT to software testing has proven capable of fault detection while minimizing the test set size requirements.

CIT requires an interaction level to be defined, this interaction level dictates the the number of input parameters for which interactions of possible input parameter values is treated as a test set requirement. For example if the interaction level, or strength of test set, is defined as 2 the method is referred to as Pairwise Interaction Testing. The test set created for Pairwise Interaction Testing is a t-way covering array where t=2. For example, suppose we have a dataset with three binary input variables, with names A,B, and C. The 2-way covering array for this example is presented in table 2.1. All possible interactions for this dataset between any two input parameters are included in this covering array, and would therefore be a complete test set at the t=2 interaction level. Infeasible interactions can also be included as constraints in the generation of the test set.

Table 2.1: Covering Array for Pairwise Interaction Testing example.

A	B	C
0	0	0
1	0	1
0	1	1
1	1	0

Kuhn et al provide evidence that a vast majority of software failures occur between interaction levels of 1 and 6 [1]. Thich means creating covering arrays for upto a 6 way interaction level should be sufficient to thoroughly test a software program. Figure 2.2 displays the performance of CIT in failure detection across a variety of software applications. For a complete

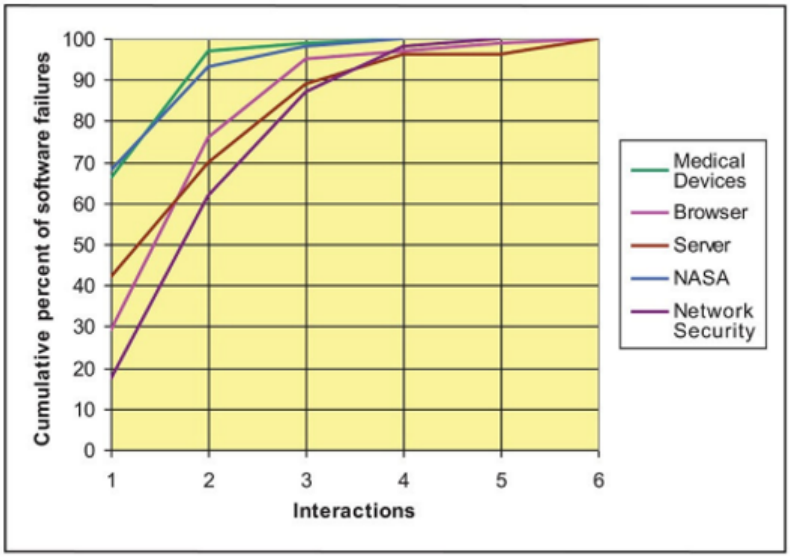

Figure 2.2: Performance of CIT across a variety of software applications versus interaction level utilized. Retrieved from: Kuhn et al[1]

review of CIT for software testing please refer to Nie and Leung's survey [23].

Machine learning can be viewed as the combination of software and data. CIT has proven to be the gold standard of software testing so if its application to data is just as beneficial then it should be of great benefit to the machine learning community. CIT has been applied to machine learning for several purposes, some notable applications include that of explainable artificial intelligence and failure analysis. In CIT for machine learning, the features of the input space and output space are factors. Values are then the specific values each factor can take. A t-way interaction is the same as a t-way value combination. This is defined as a t-tuple of (factor, value) pairs.

An extension of CIT is CC, which is the proportion of possible t-way interactions which appear in a set [24]. CC is concerned with the universe of all possible interactions, so Lanus et al. extend CC to Set Difference Combinatorial Coverage (SDCC) [25]. SDCC is the proportion of interactions contained in one dataset but not in another, this is formally

defined in Definition 2.1.

Definition 2.1 (t-way Set Difference Combinatorial Coverage)**.** Let D_L and D_U be sets of data, and ${D_L}^t$ and ${D_U}^t$ be the corresponding t-way sets of data. The set difference ${D_U}^t \setminus {D_L}^t$ gives the value combinations that are in ${D_U}^t$ but that are not in ${D_L}^t$. The t-way set difference combinatorial coverage is

$$SDCC^t(D_U, D_L) = \frac{|{D_U}^t \setminus {D_L}^t|}{|{D_U}^t|}.$$

2.2.1 Failure Analysis

Lanus et al [25] present a method for analyzing model failure. The authors utilize the CC metric for this analysis, this metric is central to the rest of this paper. CC is the number of t-way interactions appearing in a set over the total number of possible t-way interactions. Referring back to a prior example, if table 1 were the set in question for the relevant dataset then this set would have a coverage metric of 1 as it contains all possible 2 way interactions. This metric is used to reason why a model trained on images from southern California does not perform well on images from northern California but one trained on images from northern California does perform well in southern California. It is found that at t=2, a set containing images from southern California has coverage of .83 and a set containing images from northern California has coverage .93. So, it is found that the set of northern images covers a greater amount of feature space, resulting in a model with better transferability.

2.2.2 Explainable Artificial Intelligence

As artificial intelligence becomes more widespread, the need for end users understanding how the model functions to deliver an output is a necessity. Not only does this instill confidence

in the results the model produces but also lends to understanding why the model may fail in certain settings, ultimately leading to a better understanding of the inner workings of the model. Combinatorial interactions can provide an intuitive method of understanding how artificial intelligence models function by allowing the inner workings of the model to boil down to certain interactions of features resulting in a classification or prediction.

Kuhn et al. show how combinatorial interactions can be used to understand the intuition behind a machine learning image classification model [26]. An initial example is that for the classification of a cat. Interactions that are unique for a cat such as brown & furry, whiskers, claws, and not aquatic are easily descriminable from aquatic, venomous, and 6 legs which are not associated with a cat.

They also utilize the prototype ComXAI tool on the AWA database of images. The dataset contains images of animals, and the ComXAI tool can look at interactions of different levels to determine which interactions are significant to the classification of an image.

Chapter 3

Reasoning

There is a natural extension of the ideas behind CC to active learning. The following sections highlight this from intuitive and mathematical standpoints.

3.1 Intuitive Reasoning for CC in Active Learning

For a machine learning model to achieve optimal performance the model can be tuned, and the dataset the model is trained on can be altered. Active learning is not concerned with the tuning of the model, rather it only focuses on which data points would be most beneficial for a model to learn on. Current model agnostic active learning methods struggle with sampling points which are diverse yet informative. Therefore imposing a constraint on current methods or creating a new method which aims to increase coverage should create for a sampling method which grasps a larger proportion of feature space than current generalizable methods. To ensure that there is an importance metric associated with the coverage as well, data points containing lower-level missing interactions will carry a greater importance, the reason for this being twofold; the software testing literature shows the ability of lower level interactions to cover a majority of the software's function over feature space, and an information theoretic viewpoint which will be described in the following section.

3.2 Information Theoretic Approach to Combinatorial Coverage for Active Learning

The following presents an information theoretic viewpoint of why interactions of values that variables can take are important in determining the true posterior between input variables and the models prediction.

Let T_i be set of all interactions at interaction level i

Let Y be the possible classes the interaction can belong to

Mutual information is then defined as:

$I(T_i : Y) = \sum_{t \in T_i} \sum_{y \in Y} p(t, y) log \frac{p(y|t)}{p(y)}$

Given that T_i is some incomplete representation of a particular datapoint, we can use Rate Distortion Theory, as presented in [27], to show that the Rate Distortion Function converges to an optimal value for $p(y|t)$. Rate Distortion Theory was originally proposed by Shannon to quantify the information in distorted signals, where one signal is the compressed or distorted version of the complete signal. In this case the interaction is being treated as the compressed version of the entire data point, and the class it belongs to as the true signal. It is reasonable to assume that the interactions correspond to an entire datapoint, and because this is the discrete case, that datapoint should often be associated with a single class.

The rate distortion function is defined as

$$\min_{p(y|t):(d(t,y) \leq D)} \quad I(T_i : Y) \tag{3.1}$$

The mutual information function is being minimized with respect to $p(y|t)$ and the constraint

being imposed is that the distortion, $d(t, y)$, must be less than or equal to some specified level D. Here we are assuming the distortion function to be the discrete metric, it takes a value of 0 if the interaction is included in the proper class and 1 otherwise. The other common option would be squared error. The optimization problem can be solved using variational calculus and with the introduction of a Lagrange multiplier, defined here as β. The problem can now be written as:

$$F[p(y|t)] = I(T_i : Y) + \beta d(t, y) \tag{3.2}$$

Then, taking the derivative and setting equal to zero and introducing a normalization function $Z(t, \beta)$, we can find a solution for the conditional probability of y on t:

$$p(y|t) = \frac{p(y)}{Z(t, \beta)} e^{-\beta d(t,y)} \tag{3.3}$$

The Shannon lower bound also guarantees that in the discrete space the Rate Distortion function is bounded below, as described by the following equation:

$$\lim_{D \to 0} R(D) - R_L(D) = 0 \tag{3.4}$$

This means that though interactions may only contain partial amounts of information, a conditional distribution on the classes that they are associated with can be found. So if there are interactions that are not included in the training set, these interactions can provide valuable information to the model, especially those interactions which allow for the lowest amount of information transfer.

Also, since there are a greater number of classes that lower level interactions belong to, we know there is a greater entropy associated with the conditional probabilities $p(Y|T_i)$, at

lower level interactions. Therefore the intuition of missing lower level interactions having a greater effect on the determination of the posterior is also justified.

Chapter 4

Method

4.1 Problem Statement

Current active learning methods struggle with sampling points that are representative of the entire data set or feature space, resulting in sampled points being effective for the model at hand but not necessarily beneficial for different models. The ability for sampled data to be useful for different models is crucial to the machine learning deployment process as the model in use often changes. The model dependency of current active learning methods has also resulted in sampling bias. Several methods have been proposed in the literature to combat the sampling bias issue, but many are not model agnostic and those that are suffer from sampling outliers or data points which do not effectively contribute to training the model. There have not been any proposed solutions to the lack of generalizability of sampled data points to different models. The proposed method chiefly aims to solve the issue of sampled data points not contributing effectively to the training of new models and as it is a data-centric method, the effect on sampling bias is also studied.

4.2 Proposed Method

The proposed query criterion relies heavily on SDCC, where the labeled dataset is considered D_L and the unlabeled dataset D_U. The query strategy involves finding those datapoints

in the unlabeled pool which contain interactions not included in the labeled set. Those datapoints which contain a greater number of missing interactions are to have a higher priority for labeling. Once the hierarchy of datapoints to label has been determined, selection according to this hierarchy is done in three ways; coverage density sampling, informative coverage density sampling, and uncertainty sampling weighted by coverage density. These data-centric methods should aid in sampling points which allow for data transferability to new models as illustrated in figure 2.1.

Algorithm 1 presents a method to determine coverage density given unlabeled and labeled datasets. As a data point from the query set can contain several missing interactions, the sum of the number of missing interactions it contains could be considered as the density of coverage at that point. Lower level interactions are expected to be associated with a greater number of classes than higher level interactions, so they should hold a greater weight. The weighing scheme that is proposed utilizes the decreasing function $\frac{1}{t}$ for $t = 1, ..., 6$ where each t is the t interaction level. The coverage density of some point is then the weighted sum of all interactions contained in that data point. This density is used in determining which datapoints to query in the proposed methods.

Algorithm 1 Coverage Density Algorithm

function Coverage Density (L, U)
Input: labeled Set L and Unlabeled Set U
Output: Coverage Density, c
for t *in* 1 *to* T **do**
    $\mu_t \leftarrow$ LIST of interactions in L at level t
    $\beta_t \leftarrow$ LIST of interactions in U at level t
    $indices \leftarrow EMPTYLIST$
    for i *in* 1 *to* length of β_t **do**
        if i^{th} *element of* β_t *not in* μ_t **then**
            APPEND index value from β_t to indices
    for j *in indices* **do**
        j^{th} *element of* $c \leftarrow j^{th}$ *element of* $c + 1/t$

The proposed method utilizes a single coverage computation, that is, the density initally determined will dictate the sampling throughout all iterations. The reasoning for this is two-fold:

1. Initial Sampling is the most important, as sampling progresses into further iterations most methods will tend to randomly selecting points

2. In the inital stages of selecting points it is likely that points chosen with a high density will not always effect the density of other points

An example for the second reason is show in figure 4.2.

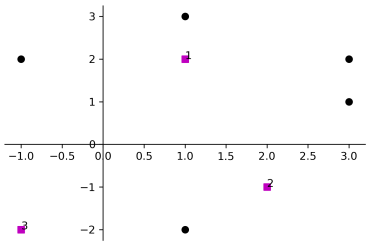

Figure 4.1: Single Coverage Computation Reasoning

In this figure let the circles represent labeled data points and the squares be unlabeled points that are available to query and label. Also suppose we can only query a single point in each iteration, and that the point to be queried will be the one with the highest coverage density as previously defined. Query point 1 has density of .5, point 2 has density 2.5, and point 3 has density .5. Point 2 will be chosen to label, but the density of point's 1 and 3 does not change with the labeling of point 2.

The following are the three proposed methods to select which data points to send to the oracle. All methods use the same definition of variables. x_i is a binary variable valued 1 if data point i will be sent to the oracle for labeling and 0 otherwise. c_i is the coverage density

of data point i as previously defined, and b is the budget or number of points we are allowed to select. The three proposed methods to sample points are presented in Definitions 4.1-4.3.

Definition 4.1 (Coverage Density Sampling)**.** The points selected are those with the highest coverage density:

$$\begin{aligned} \arg\max_i \quad & \sum_i c_i x_i \\ \text{s.t.} \quad & \sum_i x_i \leq b \end{aligned}$$

The second method weighs coverage density by similarity, which should protect against outliers. Cosine similarity is used as the measure of similarity between each query point and all other points, this is defined in Equation 4.2 where $\boldsymbol{x}$ represents all data points in both training and query sets.

Definition 4.2 (Informative Coverage Density Sampling)**.** The informativeness of a datapoint is coverage density by similarity, where U is the cardinality of the unlabeled set:

$$I(x_i) = c_i \frac{1}{U} sim(\boldsymbol{x}, x_i) \tag{4.1}$$

Where similarity is cosine similarity:

$$sim(\boldsymbol{x}, x_i) = \sum_{x \in \boldsymbol{x}} \frac{x \cdot x_i}{|x|\,|x_i|} \tag{4.2}$$

The datapoints selected should maximize the sum of informativeness:

$$\begin{aligned} \arg\max_i \quad & \sum_i I(x_i) \\ \text{s.t.} \quad & \sum_i x_i \leq b \end{aligned}$$

The final method involves weighing the common uncertainty sampling with entropy formu-

lation by the coverage density of the data point as defined previously.

Definition 4.3 (Uncertainty Sampling Weighted by Coverage Density(USWCD))**.** The informativeness of data point is defined as the following:

$$I(x_i) = H(x_i)c_i$$

Where $H(x_i)$ is the entropy of the model at prediction at point i:

$$H(i) = -\sum_{y \in Y} p(y_i) log(p(y_i))$$

That is, the entropy over all the classes that a specific data point may belong too. The datapoints selected should maximize the sum of informativeness:

$$\begin{aligned} \arg\max_i \quad & \sum_i I(x_i) \\ \text{s.t.} \quad & \sum_i x_i \leq b \end{aligned}$$

These three methods are reliant on the data, with USWCD being the only method which takes some model input. Data-centric methods should allow for the data to be better transferred between models; this is illustrated in the experiments.

Chapter 5

Experimental Results

This section contains information about the current experimental set-up, and preliminary results.

5.1 Experiment Setup

All experiments are conducted on data sets from the UCI Machine Learning repository [28], and the benchmark methods used are the ones previously defined. Table 5.1 displays general information about each data set. Batch size is the number of data points queried at each Active Learning iteration, for larger data sets a batch size of 100 is used while 25 points per sample is used for smaller data sets. All data sets, other than the Monk data set, are randomly split so that there are 10% to test on. Of the remaining 90% of data, 2.5% is used as initial training data and 97.5% is used as the query set. The Monk data set is pre-partitioned into training and testing sets, so the training set (which is the size listed in Table 1) is split into 97.5% query points and 2.5% initial training set.

Data Set	Data Points	Features	Batch Size	Number of Batches
Tic-Tac-Toe	957	9	100	8
Balance Scale	624	4	25	21
Car Evaluation	1727	6	100	15
Chess	28066	6	100	246
Nursery	12959	8	100	113
Monk	414	6	25	13

Table 5.1: Data Set Information

F1 is used as the measure of performance for each of the classifiers, as F1 will take into account class imbalance as well as model performance unlike model accuracy which only looks at model performance. F1 is defined as

$$F1 = 2 * \frac{Precision * Recall}{Precision + Recall} \tag{5.1}$$

Precision and Recall are defined as the following, where tp is true positive and fn is false negative:

$$\begin{aligned} Precision &= \frac{tp}{tp + fp} \\ Recall &= \frac{tp}{tp + fn} \end{aligned} \tag{5.2}$$

Experiments on each data set are conducted three times, each time using the same random partition of data, and taking the average F1 of the three runs to determine performance.

To quantify the performance over all iterations of sampling we take the are under the learning curve for which the x-axis is number of datapoints queried and the y-axis is F1. The area is determined using the Trapezoidal rule.

To determine the effectiveness of active learning methods in sampling points which are transferable to different models, all data sets are sampled from and tested utilizing a Random Forest Classifier with max depth constrained to 5. These sampled data points are then also used to train a Decision Tree classifier and Support Vector Machine (SVM); for the SVM a Support Vector Classifier (SVC) implementation is used. Both the Decision Tree and SVC do not have any hyperparameter tuning and are utilized as is from Scikit-learn [29]. After each iteration of sampling, all models are tested.

5.2 Experiment Results

Learning curves for each of the six datasets are presented in figures 5.1 through 5.6.

Figure 5.1: Monk Data Set Performance Plots

On the monk dataset regular uncertainty sampling does the best with the original model, when the data is transferred to new models QBC and coverage density sampling achieve the best performance. Something of note though is that none of the models achieve a stellar performance on the dataset, that is, Random Forest acheives the highest F1 at roughly .66. All three models achieve a much better performance on the balance scale data set. USWCD

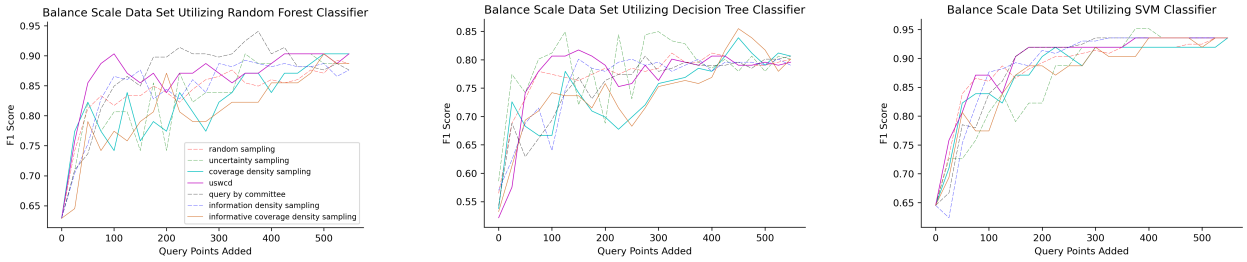

Figure 5.2: Balance Scale Data Set Performance Plots

stands out in this dataset, with both Random Forest and SVM this method achieves great increases in performance initially. Uncertainty sampling achieves the greatest performance with a Decision Tree model, but the performance experiences sharp increases and decreases whereas USWCD achieves a similar performance and is more constant. Similar results are seen on the Car Evaluation and Tic-tac-toe datasets. On the Car Evaluation dataset USWCD achieves the best performance with a Random Forest classifer and Decision Tree classifier,

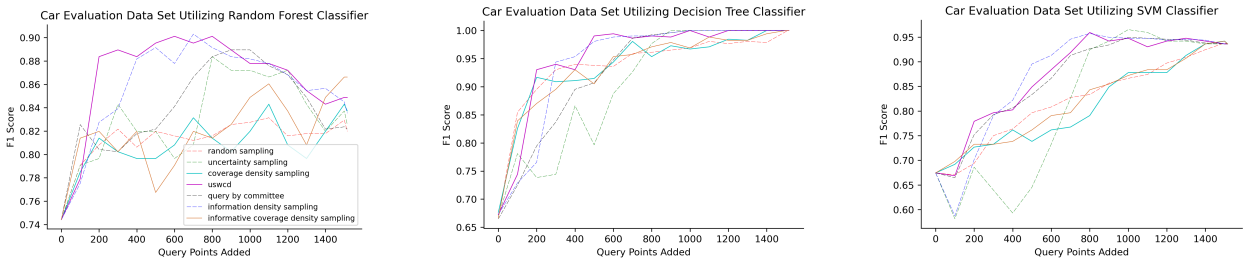

Figure 5.3: Car Evaluation Data Set Performance Plots

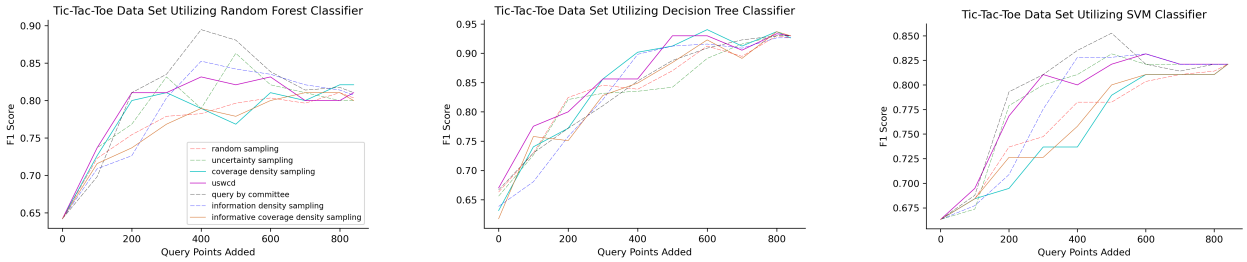

Figure 5.4: Tic-Tac-Toe Data Set Performance Plots

while achieving seemingly similar performance to information density sampling with an SVM. The difference between the proposed method and information density sampling lies in the initial iterations where USWCD far outperforms information density sampling.

QBC outperforms all other methods on the Tic-tac-toe dataset with a Random Forest classifier, but USWCD does perform similar to or better than the other benchmark methods. Once again the inital increase in performance for USWCD is greater than any other method. This is also seen with a Decision Tree and SVM, where the data sampled by USWCD is the best performer in the former and data from QBC performs best in the latter.

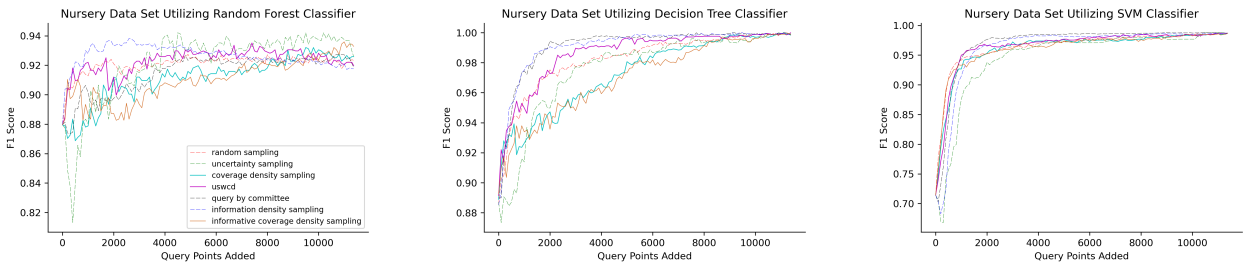

Figure 5.5: Nursery Data Set Performance Plots

A Random Forest classifier performs well on the Nursery dataset, but does so initally as well so larger performance gain are not seen. Both information density sampling and USWCD perform well initally, then information density sampling takes the lead which it soon gives up to uncertainty sampling. When the data is transferred to new models, all three of QBC, information density sampling, and USWCD perform well in different instances. An inital performance increase is seen with the proposed method once again.

Performance of the methods on the Chess dataset also supports the idea of USWCD outperforming other methods initally. This inital performance increase is seen across all three models. When the data is transferred to a Decision Tree the best performing method is intially USWCD sampling, but information density does take over in the later iterations. With an SVM model, USWCD sampling is the clear best performer both intially and overall.

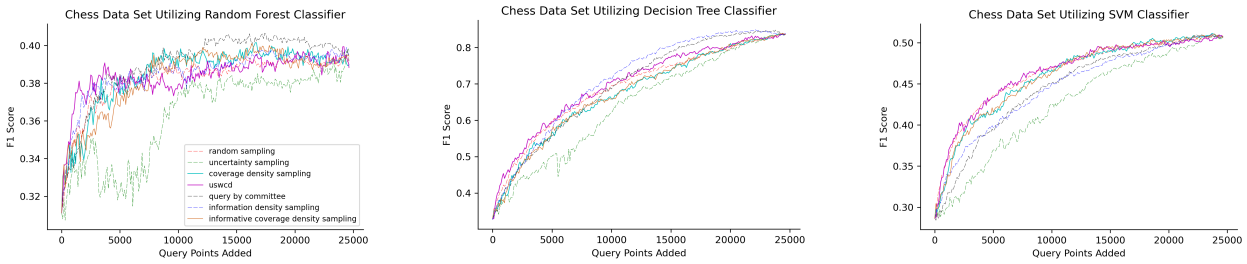

Figure 5.6: Chess Data Set Performance Plots

Initial sample performance is especially important for deployable active learning because in an application scenario where labeling is constrained the entirety of the unlabeled dataset would not be queried. From the previous plots it is clear that USWCD achieves a greater performance initially on the original model, as well as when the sampled data is transferred to new models. This confirms the hypothesis of missing interactions being inherently informative of the feature space regardless of the learner.

To quantatively compare the performance of each active learning method on the data sets, the area under the learning curve (AUC) could be utilized. The trapezoidal rule is used to

calculate the AUC, as implemented in Numpy [30]. AUC's for each of the datasets, methods, and models are presented in table 5.2.

Active Learning Method	Model	Monk	Balance Scale	Car Evaluation	Tic-Tac-Toe	Nursery	Chess
Random Sampling	Random Forest	221.60	462.63	1303.77	696.84	10499.40	9437.38
	Decision Tree	210.46	426.47	1509.68	764.03	11201.24	17051.97
	SVM	219.71	488.30	1321.41	690.35	10997.11	11464.67
Uncertainty Sampling	Random Forest	**228.47**	459.87	1332.55	714.21	10507.02	8972.98
	Decision Tree	213.20	**433.73**	1455.13	758.59	11126.62	15848.8
	SVM	221.67	477.82	1305.52	710.00	10800.42	10537.07
QBC	Random Forest	224.90	**477.55**	1346.31	**731.57**	10430.06	**9632.39**
	Decision Tree	**217.30**	415.52	1491.37	761.22	**11301.59**	17196.40
	SVM	222.03	489.51	1395.44	**717.71**	**11056.61**	11031.15
Information Density Sampling	Random Forest	220.91	466.80	1374.51	712.80	**10563.24**	9511.56
	Decision Tree	213.28	419.28	1517.34	760.35	11299.12	**17593.14**
	SVM	218.88	488.57	1403.19	703.33	10971.20	11048.85
Coverage Density Sampling	Random Forest	219.68	453.42	1297.09	705.78	10390.65	9512.64
	Decision Tree	211.20	408.87	1507.07	775.08	11090.34	16704.74
	SVM	**222.81**	484.27	1304.94	681.57	10969.15	11381.05
Informative Coverage Density Sampling	Random Forest	222.02	445.56	1314.82	693.15	10376.79	9465.12
	Decision Tree	209.71	409.81	1505.03	759.47	11065.89	16720.74
	SVM	221.73	479.83	1317.73	686.84	10970.84	11369.16
USWCD	Random Forest	220.05	477.21	**1388.95**	716.84	10522.23	9485.49
	Decision Tree	209.74	425.33	**1531.87**	**778.59**	11245.97	17300.60
	SVM	218.84	**493.54**	**1404.94**	711.05	11006.75	**11482.56**

Table 5.2: Area under Learning Curves

The proposed methods should be competitive with the benchmark methods when sampling and testing with respect to a specific model, in our case the Random Forest model restricted to a depth of 5. To examine this performance we look at the percent difference in AUC of each method from the best performing method on each dataset, this is presented in Table 5.3. USWCD is the best performer once, it has 0.00% difference in AUC from the best method, itself. Uncertainty sampling and information density also perform best only once, but QBC performs best three times. Though USWCD does not always perform the best, it achieves performance nearest the best performer in 60% of instances it is not the best performer. So, USWCD does achieve a competitive performance for the model in the active learning loop, next we study models outside the learning loop.

Active Learning Method	Monk	Balance Scale	Car Evaluation	Tic-Tac-Toe	Nursery	Chess
Random Sampling	-3.007%	-3.124%	-6.133%	-4.747%	-0.604%	-2.025%
Uncertainty Sampling	**0.000%**	-3.702%	-4.061%	-2.373%	-0.532%	-6.846%
QBC	-1.563%	**0.000%**	-3.070%	**0.000%**	-1.261%	**0.000%**
Information Density Sampling	-3.309%	-2.251%	-1.040%	-2.566%	**0.000%**	-1.254%
Coverage Density Sampling	-3.847%	-5.053%	-6.614%	-3.525%	-1.634%	-1.243%
Informative Coverage Density Sampling	-2.823%	-6.699%	-5.337%	-5.252%	-1.765%	-1.737%
USWCD	-3.685%	-0.071%	**0.000%**	-2.013%	-0.388%	-1.525%

Table 5.3: Percent Difference from AUC of Best Performing Method on Original Model

Active Learning Method	Model	Balance Scale	Car Evaluation	Tic-Tac-Toe	Nursery	Chess
Uncertainty Sampling	Decision Tree	-	-3.61%	-.71%	-.67%	-7.06%
	SVM	-2.15%	-1.20%	-	-1.79%	-8.09%
QBC	Decision Tree	-	-1.21%	-.37%	**.9%**	.85%
	SVM	.25%	5.60%	-	**.54%**	-3.78%
Information Density Sampling	Decision Tree	-	.51%	-.48%	.87%	**3.17%**
	SVM	.06%	6.19%	-	-.24	-3.63%
Coverage Density Sampling	Decision Tree	-	-.17%	1.45%	-.99%	-2.04%
	SVM	-.83%	-1.25%	-	-.25%	-.73%
Informative Coverage Density Sampling	Decision Tree	-	-.31%	-.6%	-1.21%	-1.94%
	SVM	-1.74%	-.28%	-	-.24%	-.83%
USWCD	Decision Tree	-	**1.469%**	**1.906%**	.40%	1.46%
	SVM	**1.073%**	**6.321%**	-	.09%	**.16%**

Table 5.4: Percent Difference from Random Sampling when Model Performance Increases

In machine learning deployment, the model in use would likely not change unless the new model provides some benefit such as computational efficiency or performance. For this study we pay special attention to performance, and look at instances in which the use of a Decision Tree or SVM increases final model performance by 5% or more. As Lowell et al. [8] point out, active learning methods often do not outperform random sampling when the sampled data is transferred to a new model. Therefore, we treat random sampling as a baseline for data transfer comparison. Table 5.4 shows the percent difference in area under F1 curve between each method and random sampling when model performance increases by 5% or more. In a majority of the presented scenarios USWCD outperforms the other methods. This is also the only method which does not perform worse than random sampling in any instance of model improvement. So, the proposed method is effective in sampling data which is transferable to new and more effective models.

Taking the median of these performance values for each of the methods helps to draw the

overall conclusion that the proposed USWCD is the best performing method when transferring actively sampled data to a better performing model. This is presented in Table 5.5.

Active Learning Method	Median Percent Difference from Random Sampling
Uncertainty Sampling	-1.97%
QBC	.39%
Information Density Sampling	.28%
Coverage Density Sampling	-.78%
Informative Coverage Density Sampling	-.71%
USWCD	**1.27%**

Table 5.5: Median Percent Difference From Random Sampling when Model Performance Increases

Another hypothesis of this thesis is that a more data-centric approach to actively labeling data points should lead to a reduction in sampling bias. To quantify sampling bias a measure presented by Krishan et al. [16] is used. This is presented in equation 5.7.

$$\text{Sampling Bias} = 1 - \frac{H_{D_L}}{H_{Balanced}} \tag{5.3}$$

$H_{Balanced}$ is the entropy of a set with an equal number of datapoints from each class. H_{D_L} is defined as $-\sum_{k=1}^{K} \frac{M_k}{M} log(\frac{M_k}{M})$ where M_k is the number of datapoints belonging to class k and M is the total number of datapoints in our sample. Sampling bias plots using the afformentioned measure are presented below.

The best performing proposed method, USWCD, does achieve a lower sampling bias than random sampling for 5 out of 6 datasets. Random sampling can be considered the baseline for sampling bias, or expected to sample with the least bias because it is neither reliant on the model or data.

To further determine the effect coverage based methods may have on sampling bias, we plot

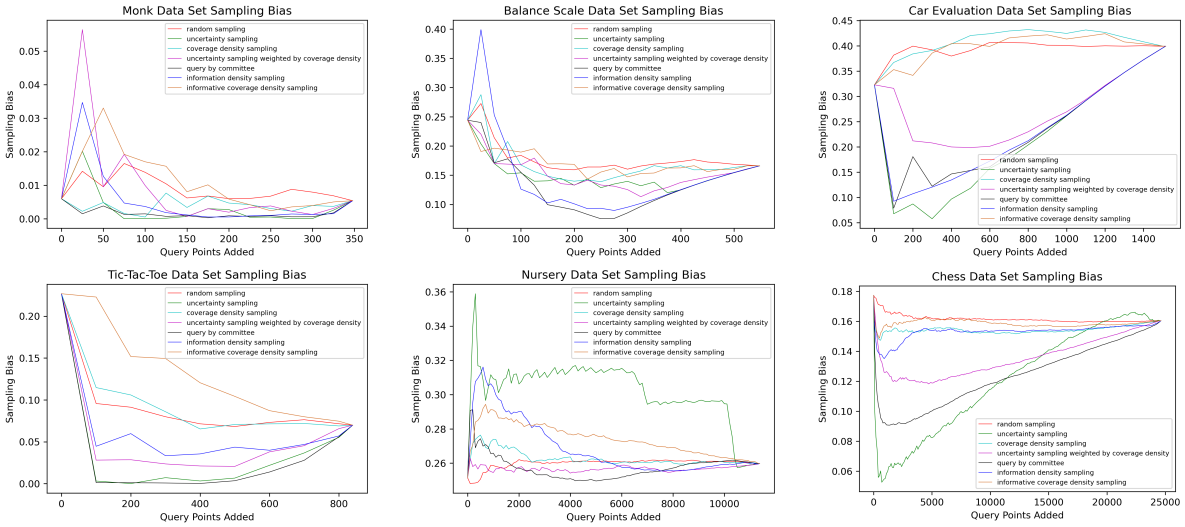

Table 5.6: Sampling Bias Plots

sampling bias versus performance with the Random Forest model used to sample the data. These plots are displayed in table 5.7, and provide an idea of how performance is effected by sampling bias for each of the methods.

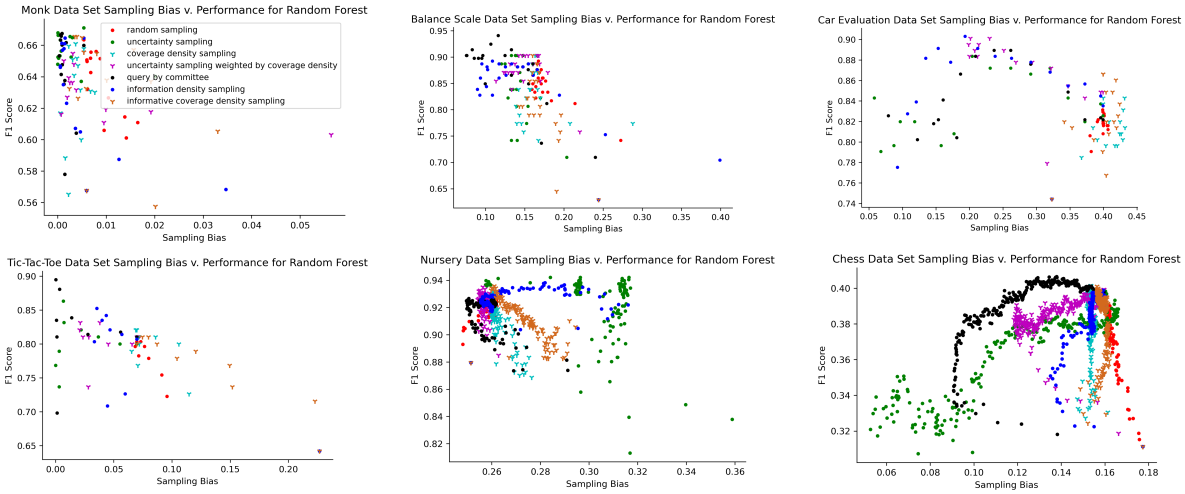

Table 5.7: Performance vs. Sampling Bias Plots

To more clearly illustrate sampling bias v. performance, table 5.8 contains the centroids for each method on each dataset. The same conclusion of outperforming random sampling can be drawn.

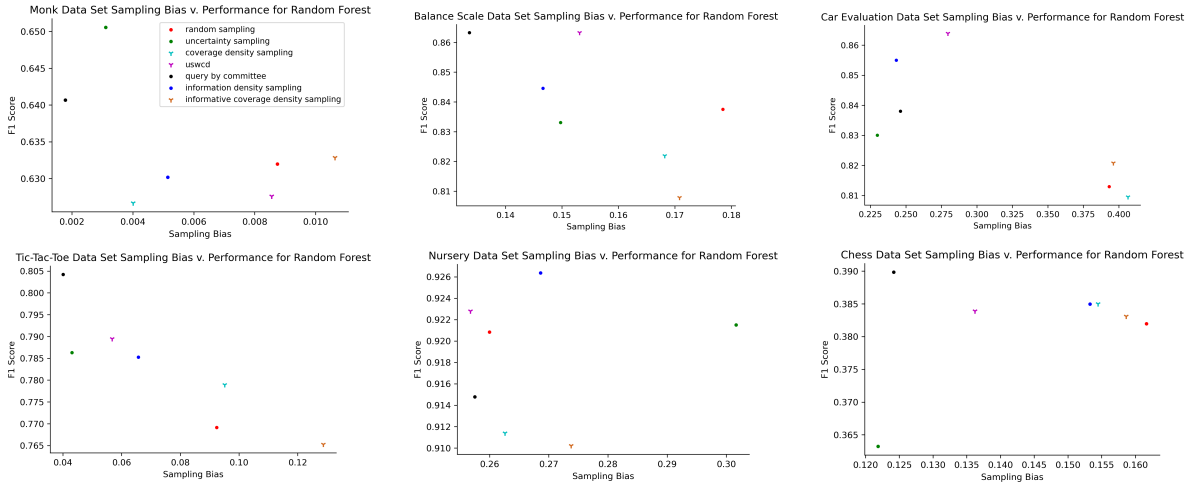

Table 5.8: Sampling Bias Centroid Plots

It seems as though all benchmark methods and USWCD outperform random sampling in a majority of instances, but to examine the effect that each method has across datasets, heatmaps are also plotted. These are displayed in table 5.9. What is more clear from these plots is that the dataset seems to have the greatest effect on sampling bias.

To determine sampling bias across all datasets a violin plot for each method is presented in figure 5.7 The violin plot is used as implemented in seaborn[31], and is created using kernel density estimation for fitting a probability distribution. The white dot in each distribution of the violin plot marks the median. Median sampling bias of USWCD is lower than that of all other methods. However, the distributions do look very similar, so testing the methods on a greater number of datasets, and datasets with a greater class diversity may help to paint a better picture of sampling bias.

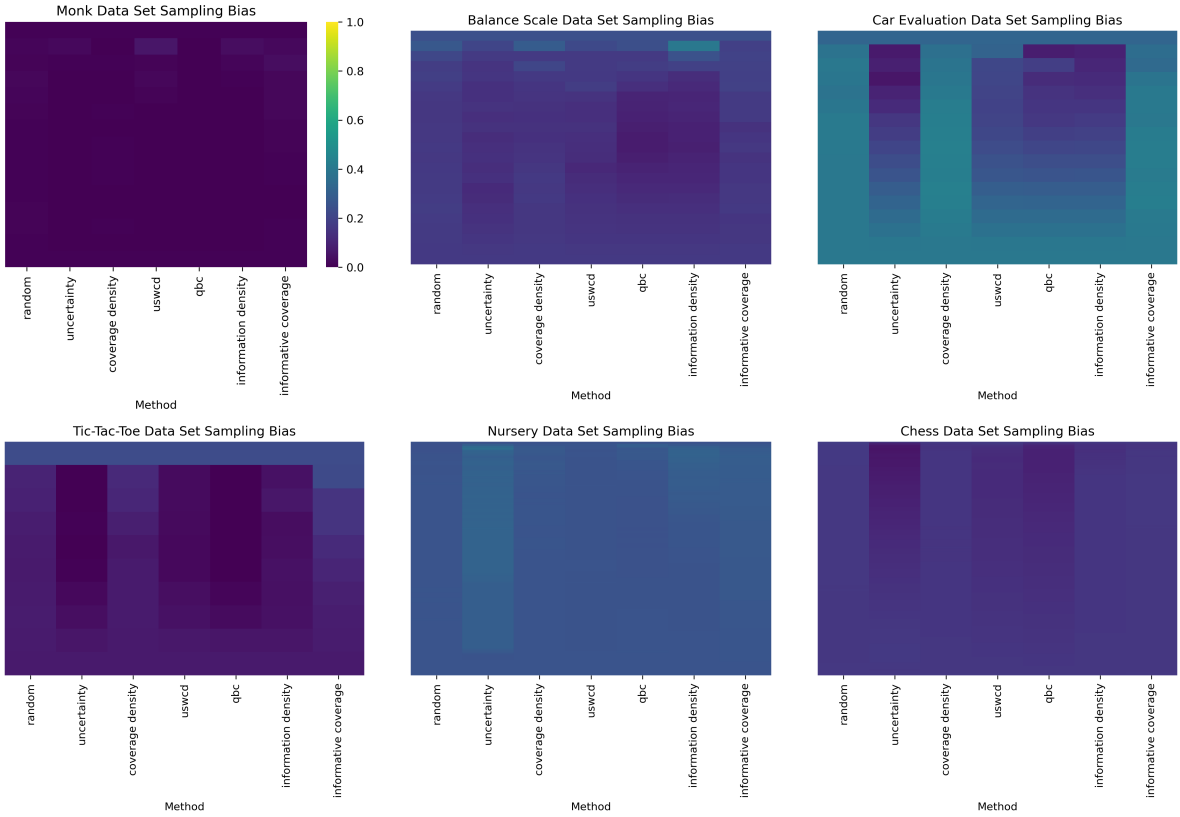

Table 5.9: Sampling Bias Heat Maps

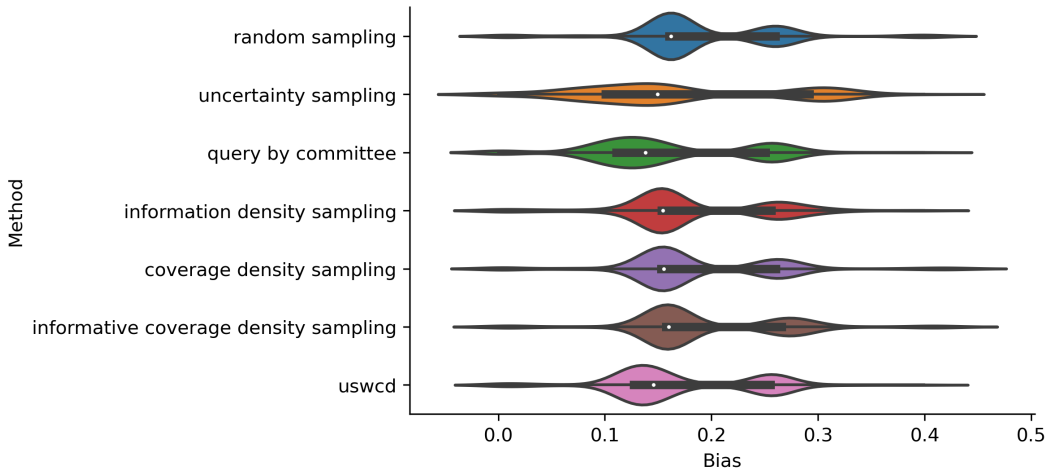

Figure 5.7: Violin Plot for Sampling Bias

Chapter 6

Conclusions

Motivated by the inability of current model generalizable active learning methods to play an effective role in the machine learning deployment life-cycle, we propose three methods utilizing combinatorial coverage to actively query points. The acquired points are highly data dependent, and are therefore more beneficial to subsequently trained models.

We tested the methods on 6 UCI Machine Learning Repository Data Sets, using a Random Forest classifier to actively sample data points. The sampled data points were then used to train a Decision Tree and SVM to simulate a changing of machine learning model in deployment. We presented the learning curves for each combination of dataset,active learning method, and model. We also use the area under curve for each learning curve to compare performance of the methods.

The proposed methods, particularly USWCD, achieves similar if not better performance than the benchmark methods when sampled data points are used for the model in the active learning loop. When the sampled data points are transferred to different models USWCD outperforms all methods. Therefore, the objective of proposing a method which competes with current methods when sampled data is used to train the model we sample with respect to is met. The objective of creating a method which beats competing methods when training and testing a model with data sampled using a different model is also achieved.

These results are summarized in table 6.1. The table shows the normalized AUC for each

method and model across all datasets. Though QBC is the best performer on the original model, USWCD is nearest to QBC in performance. When the sampled data is used to train new models, USWCD outperforms all other methods.

Method	Random Forest	Decision Tree	SVM
Random Sampling	.376	.632	.455
Uncertainty Sampling	.498	.129	.003
QBC	**.797**	.622	.854
Information Density Sampling	.740	.641	.634
Coverage Density Sampling	.160	.344	.534
Informative Coverage Density Sampling	.097	.041	.406
USWCD	.779	**.798**	**.908**

Table 6.1: Median Normalized AUC for each Query Method

We explore the effect of sampling bias a data-centric method has, and find that the best performing proposed method does achieve a sampling bias lower than that of random sampling in a majority of instances. We also find that USWCD has a lower median sampling bias than all other methods. A better comparision of sampling bias would be possible if more datasets and datasets with many classes were utilized.

Compuational cost is a drawback of coverage based methods, the cost is $O(n * f * t!)$ where $t = 6$, f is the number of features, and n is the size of the dataset. This is comparable to information density sampling, which has a cost of $O(n^2)$, and in fact is lower for large datasets with not many features. Still, lowering this compuational cost would be beneficial.

Other extensions of this work include the extension of the method to the continuous case. This would involve the discretization of data only to determine which datapoints should be queried. From there, training and testing could be conducted using the continuous data. Another interesting extension would be to neural networks and deep learning. Training a deep learning model can be computationally expensive, so re-training at every iteration of active learning, as is required by the uncertainty based or QBC methods, could be burdensome. A

coverage method, such as coverage density sampling which is entirely data dependent, might be beneficial in reducing the computational expense.

Bibliography

[1] Rick Kuhn, Yu Lei, and Raghu Kacker. Practical combinatorial testing: Beyond pairwise. *It Professional*, 10(3):19–23, 2008.

[2] Jason Brownlee. Data preparation for machine learning, 2022.

[3] Burr Settles. Active learning literature survey. 2009.

[4] Lin Yang, Yizhe Zhang, Jianxu Chen, Siyuan Zhang, and Danny Z Chen. Suggestive annotation: A deep active learning framework for biomedical image segmentation. In *International conference on medical image computing and computer-assisted intervention*, pages 399–407. Springer, 2017.

[5] Burr Settles and Mark Craven. An analysis of active learning strategies for sequence labeling tasks. In *proceedings of the 2008 conference on empirical methods in natural language processing*, pages 1070–1079, 2008.

[6] Steven CH Hoi, Rong Jin, Jianke Zhu, and Michael R Lyu. Batch mode active learning and its application to medical image classification. In *Proceedings of the 23rd international conference on Machine learning*, pages 417–424, 2006.

[7] Peilin Zhao and Steven CH Hoi. Cost-sensitive online active learning with application to malicious url detection. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 919–927, 2013.

[8] David Lowell, Zachary C Lipton, and Byron C Wallace. Practical obstacles to deploying active learning. *arXiv preprint arXiv:1807.04801*, 2018.

[9] Katrin Tomanek and Katherina Morik. Inspecting sample reusability for active learning. In Isabelle Guyon, Gavin Cawley, Gideon Dror, Vincent Lemaire, and Alexander Statnikov, editors, *Active Learning and Experimental Design workshop In conjunction with AISTATS 2010*, volume 16 of *Proceedings of Machine Learning Research*, pages 169–181, Sardinia, Italy, 16 May 2011. PMLR.

[10] Andrei Paleyes, Raoul-Gabriel Urma, and Neil D Lawrence. Challenges in deploying machine learning: a survey of case studies. *arXiv preprint arXiv:2011.09926*, 2020.

[11] David D Lewis and Jason Catlett. Heterogeneous uncertainty sampling for supervised learning. In *Machine learning proceedings 1994*, pages 148–156. Elsevier, 1994.

[12] H Sebastian Seung, Manfred Opper, and Haim Sompolinsky. Query by committee. In *Proceedings of the fifth annual workshop on Computational learning theory*, pages 287–294, 1992.

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

[14] Maryam Pardakhti, Nila Mandal, Anson WK Ma, and Qian Yang. Practical active learning with model selection for small data. In *2021 20th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 1647–1653. IEEE, 2021.

[15] Sanjoy Dasgupta and Daniel Hsu. Hierarchical sampling for active learning. In *Proceedings of the 25th International Conference on Machine learning*, pages 208–215, 2008.

[16] Ranganath Krishnan, Alok Sinha, Nilesh Ahuja, Mahesh Subedar, Omesh Tickoo, and Ravi Iyer. Mitigating sampling bias and improving robustness in active learning. *arXiv preprint arXiv:2109.06321*, 2021.

[17] Alina Beygelzimer, Sanjoy Dasgupta, and John Langford. Importance weighted active learning. In *Proceedings of the 26th annual international conference on machine learning*, pages 49–56, 2009.

[18] Ehsan Elhamifar, Guillermo Sapiro, Allen Yang, and S. Shankar Sasrty. A convex optimization framework for active learning. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, December 2013.

[19] Ozan Sener and Silvio Savarese. Active learning for convolutional neural networks: A core-set approach. *arXiv preprint arXiv:1708.00489*, 2017.

[20] Sharat Agarwal, Himanshu Arora, Saket Anand, and Chetan Arora. Contextual diversity for active learning. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *Computer Vision – ECCV 2020*, pages 137–153, Cham, 2020. Springer International Publishing.

[21] Zhuoming Liu, Hao Ding, Huaping Zhong, Weijia Li, Jifeng Dai, and Conghui He. Influence selection for active learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 9274–9283, October 2021.

[22] C. E. Shannon. A mathematical theory of communication. *The Bell System Technical Journal*, 27(3):379–423, 1948.

[23] Changhai Nie and Hareton Leung. A survey of combinatorial testing. *ACM Computing Surveys (CSUR)*, 43(2):1–29, 2011.

[24] D Richard Kuhn, Itzel Dominguez Mendoza, Raghu N Kacker, and Yu Lei. Combinatorial coverage measurement concepts and applications. In *2013 IEEE Sixth International Conference on Software Testing, Verification and Validation Workshops*, pages 352–361. IEEE, 2013.

[25] Erin Lanus, Laura J Freeman, D Richard Kuhn, and Raghu N Kacker. Combinatorial testing metrics for machine learning. In *2021 IEEE International Conference on Software Testing, Verification and Validation Workshops (ICSTW)*, pages 81–84. IEEE, 2021.

[26] D Richard Kuhn, Raghu N Kacker, Yu Lei, and Dimitris E Simos. Combinatorial methods for explainable ai. In *2020 IEEE International Conference on Software Testing, Verification and Validation Workshops (ICSTW)*, pages 167–170. IEEE, 2020.

[27] Naftali Tishby, Fernando C. Pereira, and William Bialek. The information bottleneck method, 2000.

[28] Dheeru Dua and Casey Graff. UCI machine learning repository, 2017.

[29] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[30] Charles R. Harris, K. Jarrod Millman, Stéfan J. van der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel J. Smith, Robert Kern, Matti Picus, Stephan Hoyer, Marten H. van Kerkwijk, Matthew Brett, Allan Haldane, Jaime Fernández del Río, Mark Wiebe, Pearu Peterson, Pierre Gérard-Marchant, Kevin Sheppard, Tyler Reddy, Warren Weckesser, Hameer Abbasi, Christoph Gohlke, and Travis E. Oliphant. Array programming with NumPy. *Nature*, 585(7825):357–362, September 2020.

[31] Michael L. Waskom. seaborn: statistical data visualization. *Journal of Open Source Software*, 6(60):3021, 2021.