Bounded Expectation of Label Assignment: Dataset Annotation by Supervised Splitting with Bias-Reduction Techniques

Alyssa K. Herbst

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
Computer Science & Application

Bert Huang, Chair
Noah D Barnette
Sharath Raghvendra

December 2, 2019
Blacksburg, Virginia

Keywords: Machine Learning, Active Learning, Dataset Annotation

Bounded Expectation of Label Assignment: Dataset Annotation by Supervised Splitting with Bias-Reduction Techniques

Alyssa K. Herbst

(ABSTRACT)

Annotating large unlabeled datasets can be a major bottleneck for machine learning applications. We introduce a scheme for inferring labels of unlabeled data at a fraction of the cost of labeling the entire dataset. We refer to the scheme as Bounded Expectation of Label Assignment (BELA). BELA greedily queries an oracle (or human labeler) and partitions a dataset to find data subsets that have mostly the same label. BELA can then infer labels by majority vote of the known labels in each subset. BELA makes the decision to split or label from a subset by maximizing a lower bound on the expected number of correctly labeled examples. BELA improves upon existing hierarchical labeling schemes by using supervised models to partition the data, therefore avoiding reliance on unsupervised clustering methods that may not accurately group data by label. We design BELA with strategies to avoid bias that could be introduced through this adaptive partitioning. We evaluate BELA on labeling of four datasets and find that it outperforms existing strategies for adaptive labeling.

Bounded Expectation of Label Assignment: Dataset Annotation by Supervised Splitting with Bias-Reduction Techniques

Alyssa K. Herbst

(GENERAL AUDIENCE ABSTRACT)

Most machine learning classifiers require data with both features and labels. The features of the data may be the pixel values for an image, the words in a text sample, the audio of a voice clip, and more. The labels of a dataset define the data. They place the data into one of several categories, such as determining whether a image is of a cat or dog, or adding subtitles to Youtube videos. The labeling of a dataset can be expensive, and usually requires a human to annotate. Human labeled data can be moreso expensive if the data requires an expert labeler, as in the labeling of medical images, or when labeling data is particularly time consuming. We introduce a scheme for labeling data that aims to lessen the cost of human labeled data by labeling a subset of an entire dataset and making an educated guess on the labels of the remaining unlabeled data. The labeled data generated from our approach may be then used towards the training of a classifier, or an algorithm that maps the features of data to a guessed label. This is based off of the intuition that data with similar features will also have similar labels. Our approach uses a game-like process of, at any point, choosing between one of two possible actions: we may either label a new data point, thus learning more about the dataset, or we may split apart the dataset into multiple subsets of data. We will eventually guess the labels of the unlabeled data by assigning each unlabeled data point the majority label of the data subset that it belongs to. The novelty in our approach is that we use supervised classifiers, or splitting techniques that use both the features and the labels of data, to split a dataset into new subsets. We use bias reduction techniques that enable us to use supervised splitting.

Contents

List of Figures

List of Abbreviations

F Lower bound of correct labels for a tree T

$\tilde{f}$ Lower bound of correct labels for a node V

B The set of labeled data to be used in the calculation of bound $\tilde{f}$

X A set of unlabeled data points $\{x_1, x_2...\}$

Y The set of ground truth labels for X

L The set of all labeled data for a node

$\ell(x)$ Label assignment function that aims to imitate ground truth labels Y

V A leaf node in tree T

`split`(Γ, V) The splitting function that takes in training data Γ as input. Returns k new nodes, $\{V_1, ...V_k\}$ as output such that $\bigcup_{i=1}^{k} = V$

Γ The set of labeled data to be used in the calculation of `split`

ρ The probability of a data point x being added to Γ after ground truth label y is obtained from the oracle. The probability of a data point being added to B is $(1 - \rho)$

Adaptive Hoeffding (AH) A concentration bound used in the estimation of correctly labeled data points (definition Proposition 2.1)

T A set of nodes $\{V_1, V_2...\}$ that comprise a tree

Chapter 1

Introduction

One of the key bottlenecks in modern machine learning is the annotation of datasets. While advances in technology have significantly increased the ability of computers to collect large amounts of unlabeled data, supervised learning requires annotation of this data. For example, in classification tasks, this annotation typically requires human experts to provide labels for the true class of each example. The effort and cost of this labeling process is often prohibitive for many applications. In this paper, we introduce a scheme for acquisition of labeled examples that is able to infer high-quality labels with limited labeling budgets. Our approach builds a hierarchy of data subsets via Bounded Expectation of Label Assignment (BELA).

Schemes that acquire high-quality labels with lower cost can therefore have tremendous impact on the applicability of machine learning. In settings where there is a limited budget for annotation, tools are needed to identify which examples would be most informative to label. Such tools need to balance introduction of bias and coverage of the example space. For example, in datasets where different classes have nonuniform proportions, it can be important to ensure coverage of underrepresented classes while avoiding wasteful oversampling of overrepresented classes.

Random sampling introduces no bias, but it can provide a poor representation of the data space when the budget is low. Random sampling has difficulty finding examples in rare classes or in sparse regions of the input space. On the other hand, active learning approaches [7]

aim to acquire labels for data most useful for training specific model families. This goal can introduce significant bias [5]. For example, methods such as uncertainty sampling prefer labeling points close to a model's decision boundary, so the distribution of labeled points will be highly dependent on the model family being trained. The acquired labels may not be as useful for training other model families.

Instead, a better approach is to use the structure of the data to determine which examples to label. Dasgupta and Hsu [9] introduced such an approach called hierarchical sampling for active learning (HSAL). They construct a hierarchical clustering and adaptively determine how to prune the clustering. The idea behind HSAL is that if examples sampled from a cluster exhibit high label uniformity, i.e., are mostly the same label, then it can be inferred that the rest of the cluster is likely to have that majority label. The effectiveness of HSAL thus relies on the quality of the clustering and how well it aligns with the true labels of the classification task. In many settings, a feature-based clustering can have low label uniformity, resulting in negligible gains when using HSAL.

Dasgupta and Hsu [9] use an unsupervised clustering approach to avoid introducing bias into their data partitioning. However, using some acquired data to guide the partitioning can drastically improve the uniformity of the partitions. Our approach is therefore to design a scheme that can use supervised splitting to partition the data but takes actions to remove the bias induced by such splitting. Supervised splitting allows the algorithm to adapt to information it obtains during labeling, leading to higher quality label inferences and data efficiency.

To the best of our knowledge, BELA is the first work that partitions data guided by a supervised model. Using supervision is ideal for settings in which clustering does not partition the data well by label. One extreme example of this scenario is when data is uniformly distributed throughout most dimensions of the feature space, in which case even the best

clustering methods will fail. Data is often collected with many measurements irrelevent to the target concept. Approaches that interactively learn the structure of data can be restricted by the clustering scheme used. Our approach breaks this restriction by partitioning data with supervised models that can find relevant partitionings.

Our contributions are that we introduce our BELA strategy for actively choosing examples to label. We derive BELA using probabilistic analysis to ensure that the actions chosen by BELA greedily improve a lower bound on the expected number of correct labels by the current hierarchy. We define the overall method to use either unsupervised or supervised splits and a procedure to ensure that supervised splits do not introduce harmful bias to our labeling scheme. We evaluate BELA on labeling of three datasets and test the ability of the inferred labels to train models. Through these experiments, we demonstrate that BELA is able to acquire good estimates of labels with high data efficiency.

Chapter 2

Review of Literature

Active learning is a popular approach for training a learner in the setting where obtaining labeled data is expensive and unlabeled data is abundant. The learner selects data sequentially to be labeled by an oracle, or human labeler. The learner chooses this data by predicting which examples could be most informative. There are several approaches for deciding the "most informative" data to label, including least-confidence [18], least-margin [15], and least-entropy [24]. Other approaches learn by querying informative and representative examples [13]. Unlike traditional active learning in which individual data is chosen to be the most informative, we seek to choose the most informative subset of data, then randomly choose a sample from that subset to label. We also perform an additional step where we infer labels of unlabeled data if we believe that most data in a subset has the same label, based on the true labels obtained by the oracle. While the typical goal in active learning is to label the data most useful for training a specific model, our goal is to output a labeling of the data that could be useful for training any downstream classifier.

The work of Dasgupta and Hsu [9] on HSAL is foundational to ours. The key limitation that has prevented this innovative work from having massive impact on the data-hungry state of applied machine learning is that the fixed, unsupervised clusterings that HSAL is restricted to often do not fit the label patterns of data. And when they do not fit, the statistical assumptions that are needed for the correctness of their approach prevent any adjustment. Our goal in this work is to build a workaround for this issue, allowing the data partitioning

scheme to adapt to the labels it observes, catching data inefficiencies in the label inference scheme. Recent work by Tosh and Dasgupta [25] aims to mitigate the effects of an incorrect clustering by presenting the oracle with a snapshot of a clustering and obtaining a corrected clustering back from the oracle. However, our approach avoids introducing more complex tasks for the oracle, only asking it to classify individual examples.

The output of our BELA method is not only a set of inferred labels, but also bounded confidences in those inferences. This type of information makes the labels our method generates amenable to use in weakly supervised learning. Recent methods for weakly supervised learning include approaches that allow annotators to design noisy labeling functions or weak signals and methods that use confidence values to reason about dependencies among the weak signals [2, 23].

Our method for choosing the most informative subset to sample from relies on classical concentration bounds on random variables [19]. These bounds follow the intuition that if we have seen more labeled samples from a subset of data, we have a more confident estimate of the subset's label distribution. These bounds guarantee limits on the deviation between empirical measurements of statistics and their true expectations. Our analysis mainly uses Hoeffding's Inequality [12], which is a distribution-free concentration bound—meaning it holds for any underlying distribution. More specifically, we use a variation of Hoeffding Bounds called the Adaptive Hoeffding Inequality (AH) from the work of Zhao et al. [28]:

Lemma 2.1. Adaptive Hoeffding Inequality: Let X_i be zero-mean 1/2-subgaussian variables. Let $S_t = \sum_{i=1}^{t} x_i, t \geq 1$. Let $f : \mathbb{N} \to \mathbb{R}^+$ such that $f(t) = \sqrt{at \log(\log_c t + 1) + bt}$. Let $c \geq 1$, $a \geq c/2$, $b \geq 0$, and ζ is the Riemann-ζ function. Then,

$$Pr\left[\exists t, S_t \geq f(t)\right] \leq \zeta(2a/c) \exp(-2b/c) \tag{2.1}$$

The Adaptive Hoeffding Bound allows a practitioner to decide when to end an experiment as the experiment is occurring, while still maintaining a valid bound. Traditional Hoeffding bounds require the practitioner to decide on a fixed sample size before samples are obtained. We derive our bound for the expected number of mislabeled points for a node, based off of the labeled data L and bounds data B for the given node V.

Chapter 3

Bounded Expectation of Label Assignment

The algorithm we introduce aims to assign labels to an unlabeled dataset using a limited number of queries to an oracle, or human-labeler, which returns the ground-truth label to a single data point. We provide pseudocode for BELA in Algorithm 2. Due to the limited budget of oracle queries, we cannot obtain labels for the entire dataset and must guess the true labels for some data. The algorithm searches for uniform subsets of data, that is, subsets for which it is believed that most of the ground truth labels are the same. The data subsets are referred to as nodes V_i of tree T. At any given point in time, we will take one of two actions that will lead us to finding uniform nodes:

1. Query the oracle for the label of a data point,
2. Split a node into child nodes if we are confident that the child nodes are more uniform than the parent

Our algorithm decides to split a node or to query the oracle for the label of a data point based on whichever action has a higher potential for finding uniform datasets. This decision is based off of the bound for labeling a new data point Eq. (4.12) or for splitting a dataset Eq. (4.13). The chosen action has the highest bound for expected number of correctly labeled points. When our algorithm terminates, we will assign the majority label of each node to the unlabeled data points within the node. This label assignment can then be used as input data toward a classification task or as weak supervision.

Algorithm 1 BELA

Require: Dataset $X = \{x_1, x_2, ...x_N\}$, $Y = \varnothing$, Dataset Tree T, and oracle labeler $\ell(x)$

Initialize a root node with all data; $V_r \leftarrow \{x_1, x_2, ...x_N\}$
while $B > 0$ do
    Choose best node V and action a corresponding to

$$\text{Decision}(V, a) = \underset{V \in T, a \in \{\text{split,label}\}}{\arg\max} \tilde{f}_a(V, B, L) + \sum_{V' \in (T-V)} \tilde{f}(V', B_{V'}, L_{V'})$$

    if $a =$ label then
        `obtain_label_from`(V)
    else
        $\{V_1, ...V_k\} =$ `split`(V),
        Remove parent node from tree ; $T \leftarrow T - V$
        Add child nodes to tree; $T \leftarrow T \cup \{V_1, ...V_k\}$
    end if
end while
for $V \in T$ do Set $y_i = \hat{y}_V$ for i in $|V - L|$ and add label to return set $Y \leftarrow Y \cup \{y_i\}$ end for
return Y

3.1 Algorithm Description

Let $T = \{V_1, ..., V_{|T|}\}$ be the set of leaf nodes of our tree, where each node contains some data points $V = \{x_1, ...x_{|V|}\}$. Let Y be the set of ground truth labels of all examples. At all times during the algorithm, we assume that each data example is assigned the majority label of its leaf node, i.e., $\hat{y}_V$.

We maintain a bound, described later in Eq. (4.11) on the expected number of correctly labeled data points according to the current tree and majority labels for each node. The bound is used to decide for which node to perform an action and which action to perform. At each time step, the node V and action a are chosen as follows:

$$\text{Decision}(V, a) = \underset{V \in T, a \in \{\text{split,label}\}}{\arg\max} \tilde{f}_a(V, B, L) + \sum_{V' \in (T-V)} \tilde{f}(V', B_{V'}, L_{V'}) \tag{3.1}$$

We describe the derivation of this estimate formula in Section 4.1. Since the formula depends on quantities that are either known or can be estimated as the result of split and label actions, we can use the formula to anticipate the resulting estimate after these actions.

3.1.1 Labeling Procedure

When performing an oracle query, we anticipate that the oracle will return the leaf node's majority label for the data point queried, thus increasing the count of the majority label and the label count. Therefore, the number of estimated correct labels for the query operation is

$$\tilde{f}_{\text{label}}(V, B, L) = \tilde{f}(V, B + x, L + x). \tag{3.2}$$

The full version of this bound is given by Eq. (4.12).

3.1.2 Splitting Procedure

For the split operation, a subroutine $\texttt{split}(\Gamma, V)$ will divide the examples in V into a set $\{V_1, ...V_k\}$ of child nodes. The split is guided by the training data in Γ, such that Γ is a subset of labeled data within the node: $\Gamma \subseteq L \subseteq V$, $\Gamma \cap B = \emptyset$. The split operation will also distribute the labeled examples among the child nodes. That is, $\{V_1, ...V_k\} = \texttt{split}(\Gamma, V)$, $\{B_1, ...B_k\} = \texttt{split}(\Gamma, B)$, and $\{L_1, ...L_k\} = \texttt{split}(\Gamma, L)$.

We further analyze the effect of unsupervised and supervised splitting on our ability to estimate the number of correctly labeled examples in ??.

Algorithm 2 BELA

Require: Dataset $X = \{x_1, x_2, ...x_N\}$, $Y = \varnothing$, budget B, training ratio ρ, splitting function `split`, and oracle labeler $\ell(x)$

Initialize a root node with all data; $V_r \leftarrow \{x_1, x_2, ...x_N\}$
Create a tree with root V_r: $T \leftarrow \{V_r\}$
while $B > 0$ do
  Calculate bound on expected correct labels for all leaves; $S \leftarrow \left[\tilde{f}(V_i) \text{ for leaf } V_i \in T\right]$
  Calculate bound on expected correct labels for all leaves if label: $S_{\text{label}} \leftarrow \left[\tilde{f}_{\text{label}}(V_i) \text{ for leaf } V_i \in T\right]$
  Calculate bound on expected correct labels for all leaves if split: $S_{\text{split}} \leftarrow \left[\tilde{f}_{\text{split}}(V_i) \text{ for leaf } V_i \in T\right]$
  Choose best node V and action a corresponding to

$$\text{Decision}(V, a) = \underset{V \in T, a \in \{\text{split,label}\}}{\arg\max} \tilde{f}_a(V, B, L) + \sum_{V' \in (T-V)} \tilde{f}(V', B_{V'}, L_{V'})$$

  if $a =$ label then
    Select x_i randomly from V
    Obtain label from oracle: $y_i \leftarrow \ell(x_i)$
    Add new label to set of labels: $Y \leftarrow Y \cup y_i$
    Add new label to set of labeled data for selected node: $L \leftarrow L \cup (x_i, y_i)$
    Choose random number $0 \leq r \leq 1$.
    if r $< \rho$ then
      Add x_i, y_i to V's isolated training set: $\Gamma \leftarrow (x_i, y_i)$
    else
      Add x_i, y_i to V's isolated set for computing node bound: $B \leftarrow (x_i, y_i)$
      Update majority label for node V: $\hat{y} = \max_{\hat{y}} \sum_{y_j \in B} I(y_j = \hat{y})$
      $B \leftarrow B - 1$ if x_i has never been labeled
    end if
  else
    Split parent node into set of child nodes; $\{V_1, ...V_k\} =$ `split`(Γ, V), $\{B_1, ...B_k\} =$ `split`(Γ, B), and $\{L_1, ...L_k\} =$ `split`(Γ, L)
    For each $\{V_1, ...V_k\}$, empty isolated training set $\{\Gamma_1, ...\Gamma_k\} = \{\varnothing, ...\varnothing\}$
    Remove parent node from tree ; $T \leftarrow T - V$
    Add child nodes to tree; $T \leftarrow T \cup \{V_1, ...V_k\}$
  end if
end while
for $V \in T$ do Set $y_i = \hat{y}_V$ for i in $|V - L|$ and add label to return set $Y \leftarrow Y \cup \{y_i\}$ end for
return Y

Chapter 4

Theoretical Analysis

First, we define the probability of violating a bound on the ratio of a particular label in a data subset. This will be useful later in bounding the expected value of correct label assignments.

Theorem 4.1. Let the labeled data from V be L, and the bounds data from V be B such that $B \subseteq L \subseteq V$ at time $t = |B|$. Let

$$S_t = \frac{|V - L|}{|B|} \sum_{y_j \in B} \big[I(y_j = \hat{y}) \big] - \sum_{y_i \in (V-L)} \big[I(y_i = \hat{y}) \big] \tag{4.1}$$

be the result of a random sampling from node V, and let $f(t) = qt$. Let $c \geq 1$, $a \geq \frac{c}{2}$, and $0 \leq q \leq 1$. Then,

$$Pr\big[\exists t, S_t \geq f(t) \big] \leq \delta(q) \tag{4.2}$$

where

$$\delta(q) = \zeta(a/c) \exp\Big\{ \frac{-2}{c} \big(q^2 t - a \log(\log_c t + 1) \big) \Big\} \tag{4.3}$$

Proof. Let

$$\delta = \zeta(2a/c) \exp(-2b/c) \tag{4.4}$$

be the probability of encountering a bound violation from AH Eq. (2.1). We obtain Eq. (4.1) by centering the count of majority label $\hat{y}$ about the expected frequency of $\hat{y}$ in the bounds data. We choose $f(t) = qt$, such that q represents a "buffer" frequency from the expected frequency of $\hat{y}$. That is:

$$Pr\left[\frac{1}{|B|}\sum_{y_j \in B} I(y_j = \hat{y}) - \frac{1}{|V-L|}\sum_{y_i \in (V-L)} I(y_i = \hat{y}) \geq q\right] \leq \delta \tag{4.5}$$

Given that $f(t) = qt = \sqrt{at\log(\log_c t + 1) + bt}$, we can solve for b:

$$b = q^2 t - a\log(\log_c t + 1) \tag{4.6}$$

Substituting Eq. (4.6) for b in Eq. (4.4) gives us our final bound for δ in Eq. (4.3). □

Lemma 4.2. Adaptive Hoeffding Bounds hold assuming samples are drawn with replacement. Bardenet et al. [3] state that bounds that hold with replacement will also hold for samples drawn without replacement. That is:

$$\mathbb{E}(S_{t,nr}) \leq \mathbb{E}(S_{t,r}) \tag{4.7}$$

Where $S_{t,r}$ is S_t (Eq. (4.1)) drawn with replacement and $S_{t,nr}$ is S_t drawn without replacement.

Theorem 4.3. We present a lower bound for the expected proportion of correctly labeled points in a given node, if all unknown data $V - L$ is assigned label $\hat{y}$:

$$\mathbb{E}\left[\frac{1}{|V|}\sum_{y_i \in V} I(y_i = \hat{y})\right] \geq \tilde{f}(V, B, L) \tag{4.8}$$

such that

$$\tilde{f}(V,B,L) = \max_q \frac{1}{|V|}\left\{|L| + |V-L|\big(1-\delta(q)\big)\Big[\frac{1}{|B|}\sum_{y_i\in B} I(y_i=\hat{y}) - q\Big]\right\} \tag{4.9}$$

We are able to apply Adaptive Hoeffding Bounds to a finite dataset sampled without replacement by Lemma Proposition 4.2.

Corollary 4.4. The lower bound of the expected proportion of correct labels over the entire dataset is the weighted sum of correct labels over each node:

$$\mathbb{E}[\#\text{correct labels}] \geq F \tag{4.10}$$

where

$$F = \frac{1}{|X|}\sum_{i=1}^{i<|T|} |V_i|\ \tilde{f}(V_i,B_i,L_i) \tag{4.11}$$

Corollary 4.5. The following bound for expected correct labels if we were to label another data point is derived from Eq. (4.9):

$$\tilde{f}_{\text{label}}(V,B,L) = \max_q \frac{1}{|V|}\left\{|L|+1+\big(|V-L|-1\big)\big(1-\delta(q)\big)\Big[\frac{1}{|B|}\sum_{y_i\in B} I(y_i=\hat{y}) - q\Big]\right\} \tag{4.12}$$

This comes from the assumption that we will know the groundtruth label of one more data point. We assume that the likelihood of obtaining the majority label $L(x=\hat{y}) = \frac{1}{|B|}\sum_{y_i\in B} I(y_i=\hat{y})$ will not change.

Corollary 4.6. We bound the expected correct labels if we split the dataset into k subsets. Let Γ be a set of labeled training data such that $\Gamma \subseteq L \subseteq V$ and $\Gamma \cap B = \emptyset$. Let $\texttt{split}(\Gamma, V) = \{V_1, ...V_k\}$ be a data splitting function that splits V into k data subsets guided by Γ such

that $V = \bigcup_{i=1}^{k} V_i$. Then,

$$\tilde{f}_{\text{split}}(V, B, L) = \frac{1}{|V|} \sum_{i=1}^{k} |V_i| \; \tilde{f}(V_i, B_i, L_i) \tag{4.13}$$

Where each $\{V_1, ...V_k\} = \texttt{split}(\Gamma, V)$, $\{B_1, ...B_k\} = \texttt{split}(\Gamma, B)$, and $\{L_1, ...L_k\} = \texttt{split}(\Gamma, L)$.

Our procedure is designed to increase a pessimistic lower bound on an expected number of correctly labeled examples. As we create leaf nodes containing data subsets, we also track a set of labeled examples for each leaf that are uniformly and independently selected from the leaf population. This unbiased sample allows us to derive a bound on the expected number of correctly labeled points. To avoid introducing bias into the labeled subset, we "forget" a node's labeled points when it splits that node. The following sections discuss the bound we use and the reason this forgetting is necessary.

4.1 Derivation and Analysis of Lower Bound $\tilde{f}$

Our goal is to provide a lower bound on the expected number of correct labels obtained by BELA if we choose the majority label for each set of data. At any given point in BELA, we perform the operation that is expected to yield the largest lower bound. This bound should be pessimistic to ensure that the bound is rarely violated, but it should still provide a reasonable estimate on the expected number of correct labels. Most importantly, it should follow the same "curve" as the true count of correct labels for the algorithm to make the optimal choice of labeling a data point or splitting a set of data.

We obtain $\tilde{f}$ by taking the expected value of correctly labeled points on the success and failure of AH. The AH bound represents the probability that the true proportion of $\hat{y}$ in the unknown labels $|V - L|$ falls significantly below the empirical estimate of $\hat{y}$ in the bounds

data B. When AH holds, we expect to get at least $Pr(\text{label correct})$ correct labels out of all unknown labels with confidence $(1-\delta)$. When AH does not hold, we make the pessimistic assumption that we will not make any correct label assignments. Whether or not AH holds, we know we will make $|L|$ correct label assignments, because these are the ground truth labels that we have obtained from the oracle. The expected value of correct labels is the following:

$$\mathbb{E}(\%\ correct\ labels) \geq \frac{1}{|V|}\big[(\#\ known\ labels) + (\#\ unknown\ labels)(1-\delta)Pr(label\ correct)\big] \tag{4.14}$$

Where $Pr(\text{label correct})$ is a proportion that is some buffer size q less than the empirical proportion of $\hat{y}$:

$$Pr(label\ correct) = \frac{1}{|B|}\sum_{y_i \in B} I(y_i = \hat{y}) - q \tag{4.15}$$

The number of known data points in V is $|L|$, and the size of unknown data points is $|V-L|$. Lastly, the probability of AH failure is given by $\delta(q)$. It is important to note that AH determines both $\delta(q)$ and $Pr(label\ correct)$. Both rely on some buffer size q. A larger buffer size q will result in a larger confidence $(1-\delta)$, but a smaller $Pr(label\ correct)$. In a sense, we can perform a tradeoff of correct labels for the confidence that these labels will be correct. Since we can choose whichever q we want to perform this tradeoff, we choose the q that yields the largest $\mathbb{E}(\%\ correct\ labels)$. There is no closed-form solution for the maximum of $\tilde{f}$. However, since it is a univariate, unimodal function, ternary search [8, 17] can approximate the maximum to the desired accuracy after logarithmic executions of $\tilde{f}$. The unimodality of the bound $\tilde{f}$ can be seen because it is the sum of a function linear in

q, and a product of a concave quadratic function and a log-concave function. Since such a product is itself log-concave and therefore unimodal, and the linear function has no effect on unimodality, the overall bound is unimodal and amenable to ternary search.

Performing these substitutions and choosing the optimal q gives us our lower bound on the expected proportion of correctly labeled points:

$$\tilde{f}(V, B, L) = \max_{q} \frac{1}{|V|} \left\{ |L| + |V - L| \big(1 - \delta(q)\big) \left[\frac{1}{|B|} \sum_{y_i \in B} I(y_i = \hat{y}) - q \right] \right\} \qquad \text{(Eq. (4.9))}$$

Importantly, since each node's estimate is a lower bound on its expected number of correctly labeled examples, the sum of all nodes' estimates also bounds the expected value of the entire tree's number of correctly labeled examples. This fact follows the linearity of expectation. Thus, when BELA chooses the node-action pair that most increases the estimated bound, it is heuristically choosing to improve the overall estimate of expected total correct labels.

4.2 Supervised Splitting Procedure and Preserving Independence

Equation (4.9) uses Adaptive Hoeffding, which depends on the fact that the random variables are drawn independently from a fixed distribution. Since the random variable in question is whether an example belongs to the majority class, the observed labels satisfy this requirement if they are randomly sampled from any subset v. However, if the random points are sampled from a parent set v, which is later split into child nodes $\{u | u \in U\}$, special care is needed to ensure that the independence and uniform sampling probability hold.

When the partitioning U is determined by an unsupervised split, it is not affected by whether points are labeled or what their labels are. Instead, it can be thought of as a pre-determined partitioning. Therefore, a random sample from a parent node v that happens to be part of

a child node u is also a random sample from u.

In contrast, when partitioning U is determined by a supervised model, the data used to train the model is no longer randomly sampled from the resulting child nodes. Therefore, BELA tracks an isolated set of labeled examples for each node used to train splitting models. Since this isolated training set is independent of the data used to calculate $\tilde{f}_{\text{split}}$, it preserves the fact that the data in the partitioned subsets is randomly sampled.

Regardless of the method of splitting, the usage of the $\tilde{f}_{\text{split}}$ score to decide whether to split introduces a dependency. In practice, this dependency can be slight and may not have serious effects on the bounds. However, since the bound $\tilde{f}_{\text{split}}$ may be evaluated many times on a node—and using multiple possible partitionings—it is safer to have BELA reset its counts after the split operation. Doing so ensures that the data considered for calculating future bounds is uniformly randomly sampled from the new child nodes. This assurance is the reason for the "forgetting" in line 20 of ??. The impact of forgetting on data efficiency is somewhat mitigated by the fact that, if an example sampled for labeling has previously been labeled and forgotten, we can simply look up the known label and use it, with no need to invoke the costly oracle.

Chapter 5

Experiments

5.1 Setup

We compare BELA to a baseline of the HSAL algorithm proposed by Dasgupta and Hsu [9]. HSAL uses unsupervised splitting of data and a tree that does not adapt to new label queries from the oracle. We show the quality and the quantity of the labels obtained by our algorithm, and empirically show that our bound F is rarely violated. We apply HSAL and BELA to the 20 Newsgroups [11], MNIST [10], Fashion-MNIST [27], and a synthetic dataset of isotropic gaussian blobs. We start each method with the training sets of completely unlabeled examples, and we simulate oracle calls by revealing the true label of the requested examples. For each method, we have a budget of the full dataset size. That is, we continue to reveal labels until the entire dataset has been labeled by the oracle. In this way, we can see how "soon" correct labels are found by each of the algorithms. Each algorithm is similar in that they each provide a lower bound on the number of correct labeled points. The true number of correct labels found is also provided in the plots. This is found by counting the correct labels if we were to assign the majority label to each node of the tree.

In addition to comparing BELA to baselines of PLAL and HSAL, we try variations of our algorithm with different `split` functions. In comparing our method with HSAL we use support-vector machines (SVM) to partition the dataset. We also use multi-layered perceptrons (MLP), decision trees (DT), and naive bayes (NB). For the 20 Newsgroups

dataset, we use multinomial naive bayes, the preferred naive bayes for text data. For all other datasets, we use gaussian naive bayes.

5.1.1 Datasets

The MNIST dataset is a set of $60,000$ images of handwritten digits. Each image is 20x20 pixels, is in black and white, and contains 10 classes of digits 0-9. This dataset is popular among the machine learning community for image classification tasks.

The Fashion MNIST dataset was created to challenge machine learning researchers after classification error on digit MNIST neared 0%. Fashion MNIST also contains $60,000$ images, but instead contains 10 classes of different clothing items: T-shirt, trouser, pullover, dress, coat, sandal, shirt, sneaker, bag, and ankle boot. Each image is slightly larger than digit MNIST at 28x28 pixels, and is greyscaled.

The 20 Newsgroups dataset is a set of $18,846$ examples of blog posts corresponding to 20 different topics provided by Scikit-learn [21]. This data is preprocessed using TFIDF to normalize text examples by commonly-used words [14]. 20 Newsgroups is intended to be a challenge to our method due to its sparse features and large number of classes.

5.1.2 HSAL Setup

HSAL was implemented according to the algorithm given from [9], adapted from the implementation from Kale [16]. For the image datasets, PCA and k-means were applied hierarchically to split the dataset into subsets. While Dasgupta and Hsu used agglomerative clustering in their experiments, we found that standard implementations of this algorithm are $O(n^3)$ [4], which would be intractable for our image datasets of size 60,000. For the non-

image datasets (20 newsgroups and synthetic dataset), PCA was not used. The text data of 20 newsgroups was too sparse to apply PCA, and the synthetic dataset was 2-dimensional and did not require PCA. Each oracle from the dataset was sampled randomly without replacement.

5.1.3 PLAL Setup

PLAL was implemented as described by Urner et al. [26]. To stay consistent with HSAL, PLAL also used hierarchical PCA and k-means to cluster the image data and hierarchical k-means to cluster the text and synthetic data.

5.2 Results

Overall, we found that BELA was able to provide better guarantees (a higher bound $\tilde{f}$) than both PLAL and HSAL at almost all timesteps. BELA is able to achieve a higher maximum bound than HSAL and PLAL with respect to the bound and the true number of correctly labeled points. We attribute this to supervised splits, as it is able to provide a more refined split than unsupervised splits.

5.2.1 Image Datasets

BELA obtained a higher estimate for labels per budget than HSAL or PLAL for both image datasets. Although the clustering algorithm for HSAL was able to achieve a better split near the beginning of both trials than the SVM split for BELA, the bound for HSAL did not indicate this. It was necessary to obtain more labels from the oracle to confirm the quality of the split. At the beginning of both HSAL and BELA, HSAL is able to observe the

data features without observing the label, whereas BELA must perform a split using only the labeled examples obtained at that time period. Though HSAL and PLAL use the same `split` function, HSAL still assigns more correct labels than PLAL. It is important to note that PLAL will not assign labels within a cluster unless all the labels obtained within a trial period are the same for a single node. This trial period constitutes obtaining the labels for $40 - 150$ data points all at once. The number of labels obtained depends on the height of the tree, requiring more labels the greater the height of the node. Most datasets are noisy and will not split data perfectly when partitioned by an unsupervised learning algorithm, meaning that the occurrence of obtaining 40 data points of the same class is unlikely. For this reason, PLAL assigns labels almost linearly with respect to the budget used.

Throughout both trials, the bound for BELA was rarely violated, except for once at the beginning of the trial for the fashion mnist.

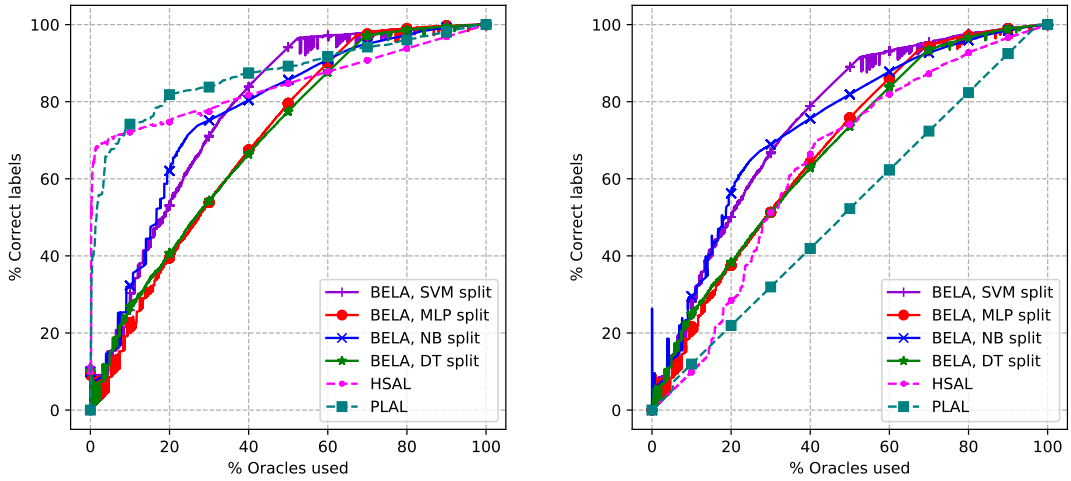
5.2.2 Text Dataset

BELA achieved a higher estimate for the number of correctly labeled data points than both HSAL and PLAL. Similarly to the image datasets, HSAL was able to achieve a better initial clustering, but was unable to exploit this clustering in the bounded estimate. Text data is notoriously difficult to cluster and proved a challenge to all methods, especially considering the dataset contained 20 classes. For this reason, all methods appear mostly linear.

5.2.3 Synthetic Dataset

HSAL and BELA had similar performance with the synthetic dataset comprised of 2-dimensional gaussian blobs, however BELA still had a higher estimated bound of correct labels. The blobs generated with this dataset were mostly distinct and non-overlapping, meaning it was easily

classified with k-means. Similarly to other datasets, HSAL performed better at guessing the true labels of each data setset near the beginning. The bound could not be verified until more budget had been used.



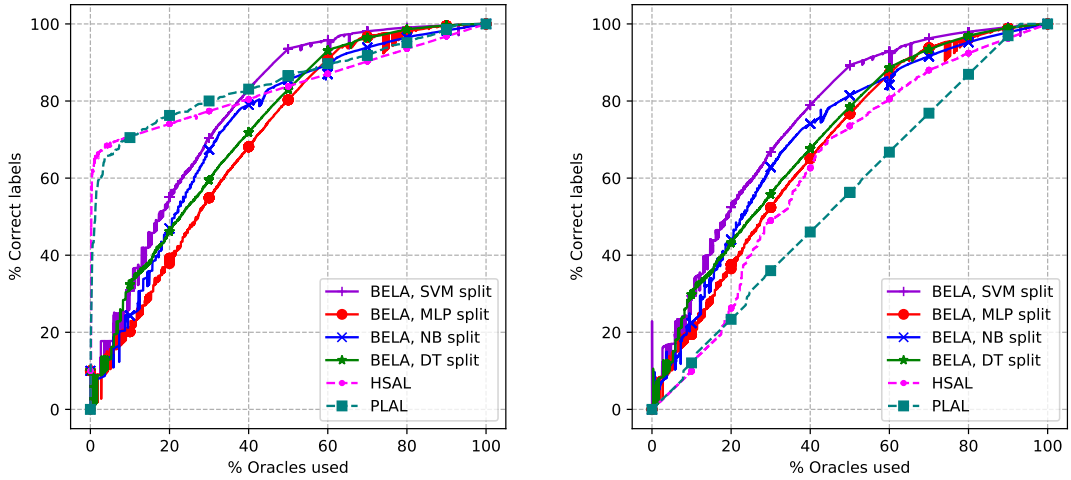
(a) True data points labeled correct (b) Bound on correct labels

Figure 5.1: Digit MNIST Results

5.3 Discussion

5.3.1 Split method evaluation

We evaluated BELA on 4 different `split` functions: support-vector machines (SVM), multi-layered perceptron (MLP), naive bayes (NB) and decision tree (decision tree). Overall, SVM and NB were found to perform the best on both image datasets and the text dataset. All methods performed comparably on the synthetic dataset, with MLP performing slightly worse than the other split methods. We speculate that SVM performed best because, as a linear classifier, it is less prone to overfitting on a small number of samples compared to

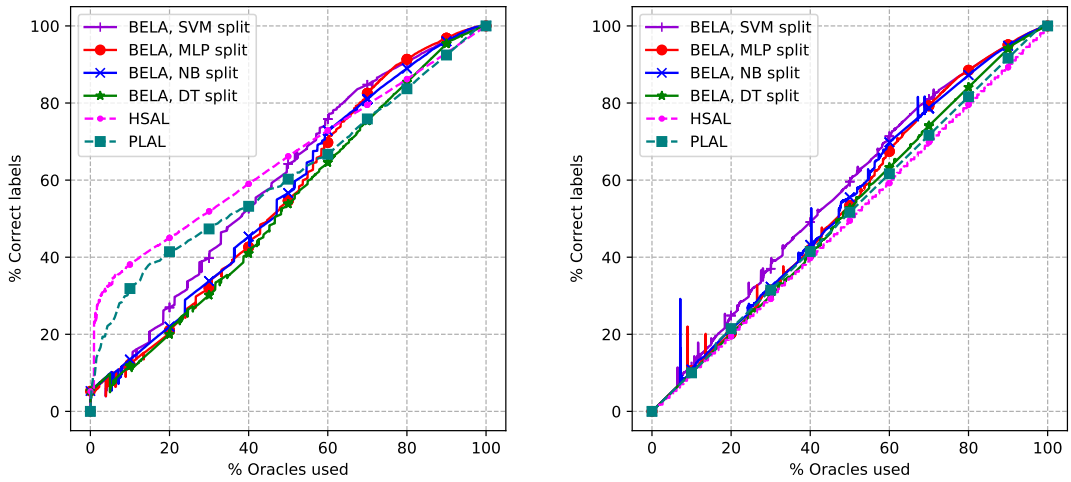


(a) True data points labeled correct (b) Bound on correct labels

Figure 5.2: Fashion MNIST Results

nonlinear methods like MLP. The success of SVM and NB could also be due to the low complexity of the models. Intuitively, when a split is performed, the distribution of the dataset becomes less complex. Models with low complexity should be able to capture more obvious relationships between dataset features without overfitting on weak relationships between features. The weaker and less obvious relationships between features may be exploited in future splits of the data.

We also examine the usage of unsupervised and supervised splits. In almost all experiments, unsupervised splits are able to perform a better initial split on the dataset. That is, the methods that use unsupervised splitting (HSAL and PLAL) have a higher percentage of true data points labeled correct than do the methods that perform supervised splits. However, while the percentage of correctly labeled data points may be higher for unsupervised splitting, the lower bound $\tilde{f}$ on the correctly labeled data still remains lower than the supervised splitting methods. The bound does not reflect the initial success of the unsupervised splits. For both supervised and unsupervised methods, we must test the quality of our splits by

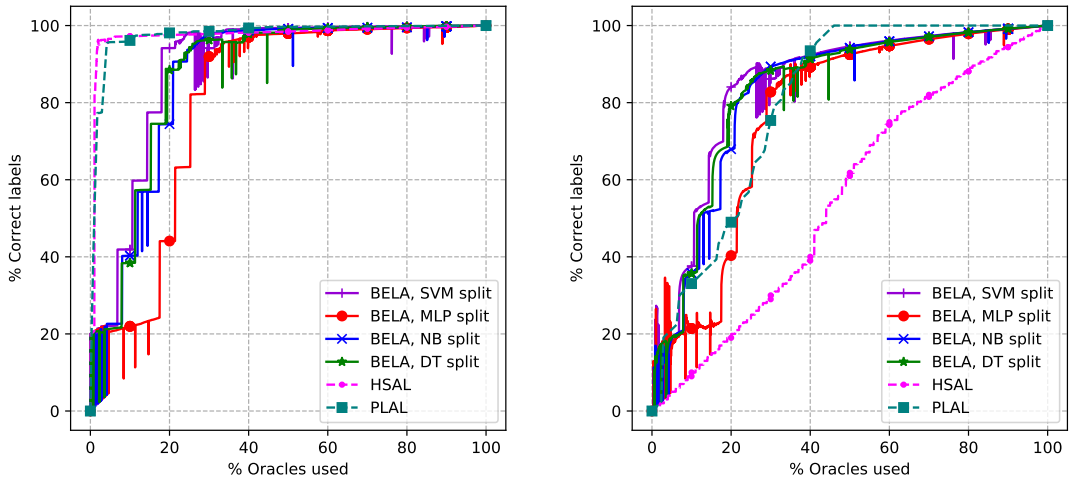


(a) True data points labeled correct

(b) Bound on correct labels

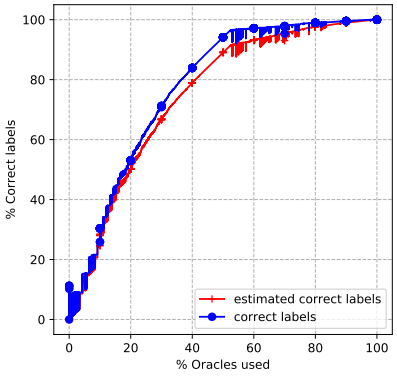
Figure 5.3: 20 Newsgroups Results

querying the oracle for labels. Later in the experiments, when we have queried higher than 50% of the dataset labels from the oracle, BELA is able to achieve a higher maximum bound than both PLAL and HSAL (except for in the synthetic dataset). We believe that this is because supervised splitting is able to perform a more refined split on the dataset that aligns better with the true labels of the data. In Section 6.1 we consider methods that take advantage of the benefits of both supervised and unsupervised splitting.

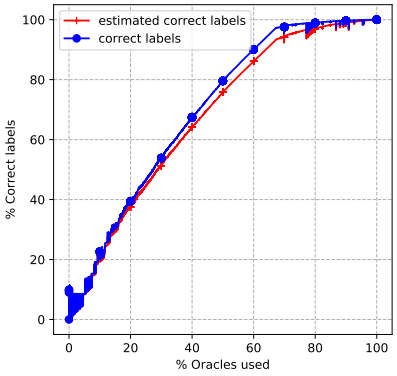


(a) True data points labeled correct (b) Bound on correct labels

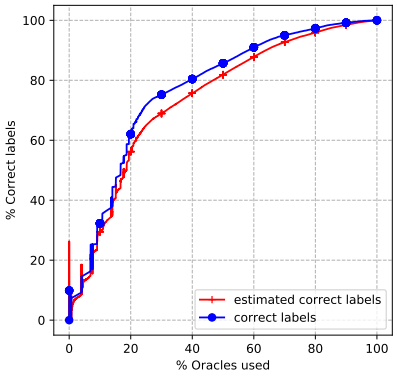
Figure 5.4: Synthetic Dataset Results



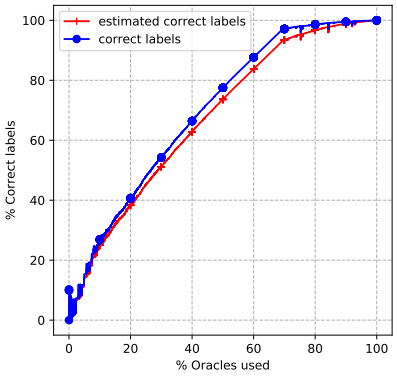
(a) Correct labels obtained from SVM BELA



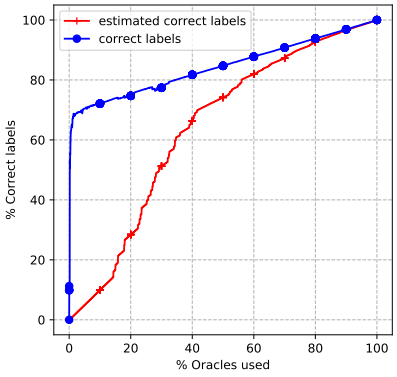
(b) Correct labels obtained from MLP BELA



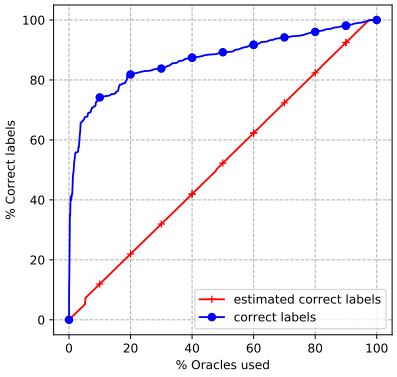
(c) Correct labels obtained from NB BELA



(d) Correct labels obtained from DT BELA

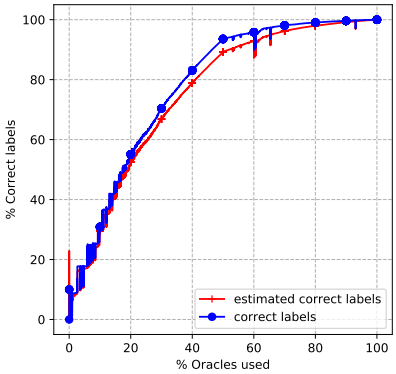


(e) Correct labels obtained from HSAL

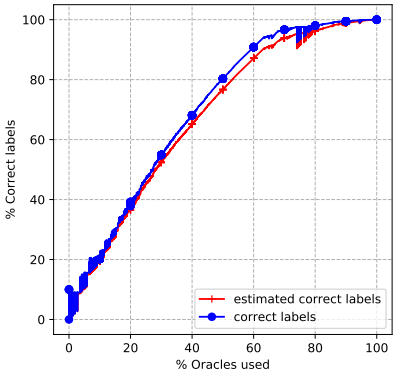


(f) Correct labels obtained from PLAL

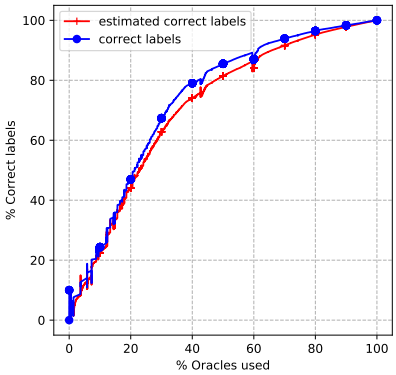
Figure 5.5: Bound Comparison for the Digit MNIST dataset



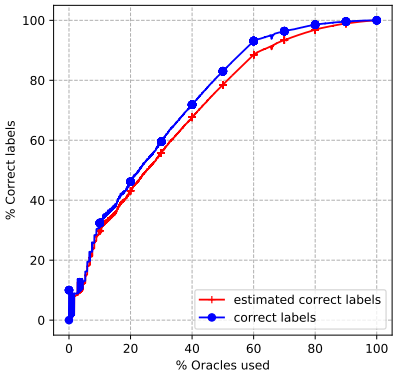
(a) Correct labels obtained from SVM BELA



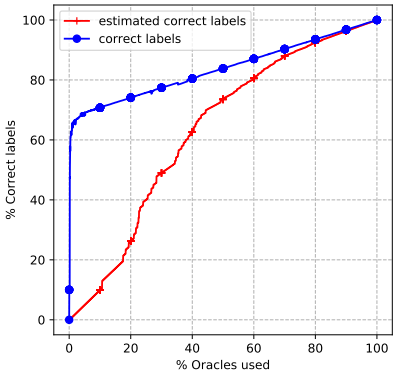
(b) Correct labels obtained from MLP BELA



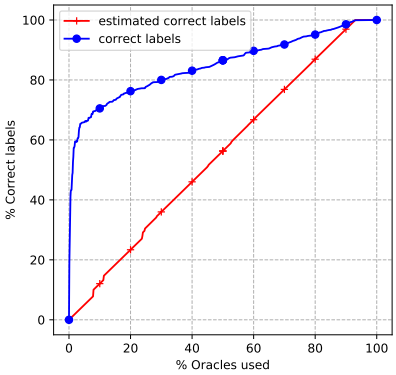
(c) Correct labels obtained from NB BELA



(d) Correct labels obtained from DT BELA

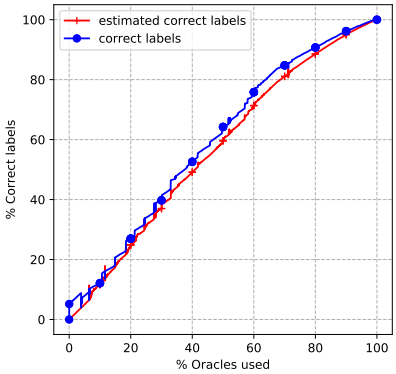


(e) Correct labels obtained from HSAL

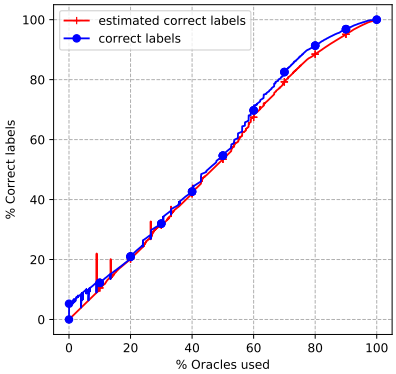


(f) Correct labels obtained from PLAL

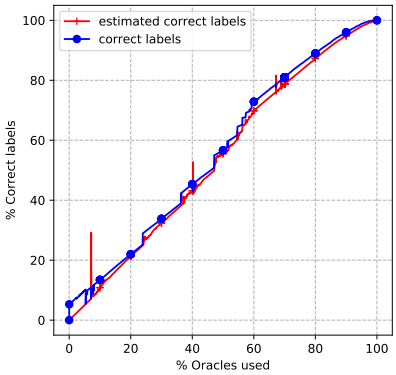
Figure 5.6: Bound Comparison for the Fashion MNIST dataset



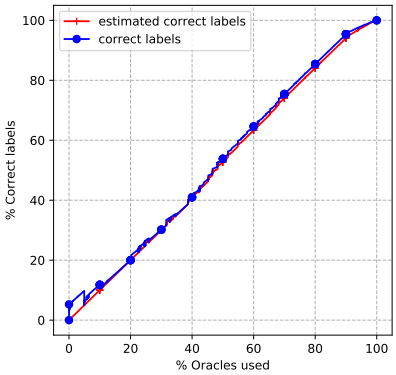
(a) Correct labels obtained from SVM BELA



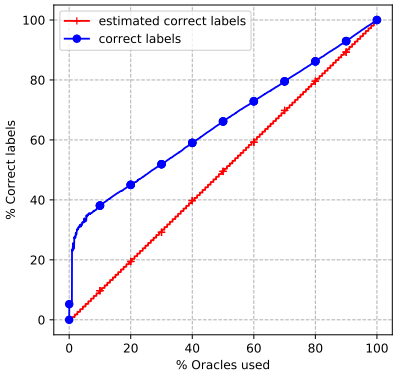
(b) Correct labels obtained from MLP BELA



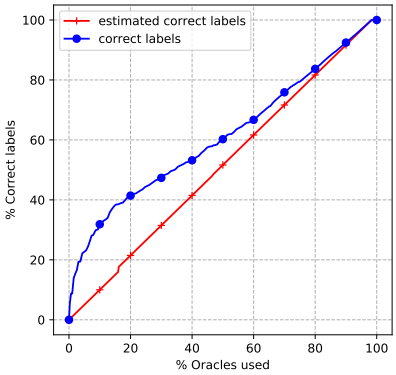
(c) Correct labels obtained from NB BELA



(d) Correct labels obtained from DT BELA

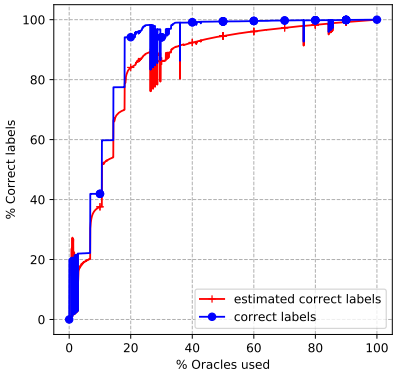


(e) Correct labels obtained from HSAL

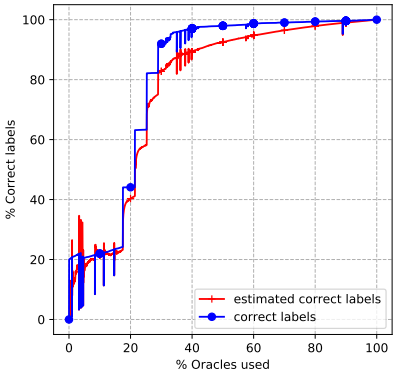


(f) Correct labels obtained from PLAL

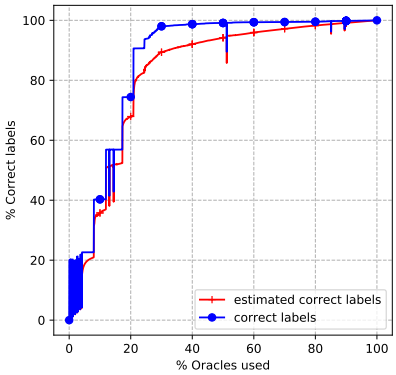
Figure 5.7: Bound Comparison for the 20 Newsgroups dataset



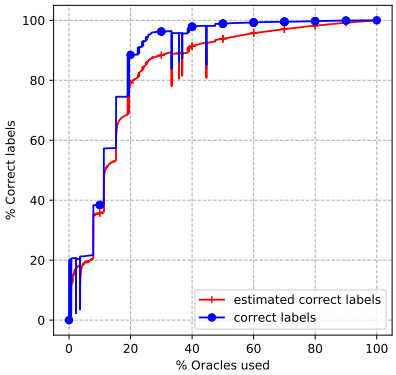
(a) Correct labels obtained from SVM BELA



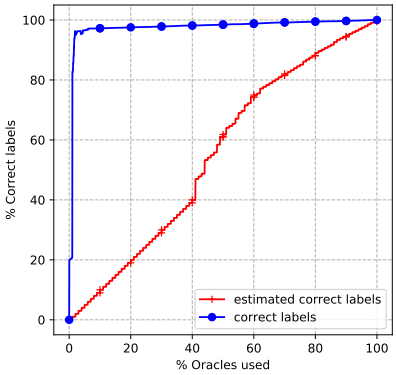
(b) Correct labels obtained from MLP BELA



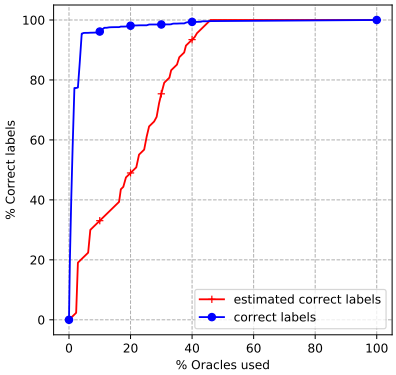
(c) Correct labels obtained from NB BELA



(d) Correct labels obtained from DT BELA



(e) Correct labels obtained from HSAL



(f) Correct labels obtained from PLAL

Figure 5.8: Bound Comparison for the Synthetic dataset

Chapter 6

Conclusion

We present a method for finding uniform subsets of an unlabeled dataset by querying labels from an oracle and splitting datasets apart in a supervised manner. By finding uniform subsets of data, we may assign labels to be used for future learning tasks without enduring the cost of labeling the full dataset. Our method uses statistical analysis to reason about when it can stop labeling a data subset with high enough confidence.

Our framework recursively splits the dataset into subsets that are each assigned a label, seeking sets of data that all contain the same label. It decides on labeling or splitting actions to increase a confidence-based estimate of the number of correctly labeled points. Our framework uses strategies to allow the use of supervised learning to guide the splitting.

Supervised splitting is preferred in many cases to unsupervised splitting because it is better at finding splits that correspond to the true labels of data, and it is therefore more likely to find uniform clusters. While supervised splitting induces a bias on the final label assignment, we remove this bias by isolating data to be used to train splits and data to be used to calculate the bounds that score the actions. In our experiments, our method is able to correctly label significant proportions of datasets while only observing the labels of a small fraction of examples. Moreover, it performs better than similar methods restricted to unsupervised splitting. With the introduction and evaluation of BELA, we take a key step toward reducing the practical cost of machine learning.

6.1 Future Research Directions

In future research we recommend investigating the usage of both supervised and unsupervised splitting methods. Unsupervised splitting is better at performing the initial splits on data when few oracles are available, while supervised splitting is better when more oracles are available and a more refined split is necessary. In order to benefit from both labeled data and unlabeled data, we could use semi-supervised splitting methods [6]. We could also decide to use a combination of supervised and unsupervised splits, for example, if we were to perform the first 3 splits with unsupervised splits, then any further splits with supervised splits. We could also add the option to choose between a supervised split function and an unsupervised split function. This option could potentially introduce more bias, as we could choose the splitting option that happened to produce more pure subsets. As a result, our lower bound estimate $\tilde{f}$ would be overconfident. Bias prevention methods should be investigated.

In addition, all experiments proved difficult with the text dataset. We seek to improve our text label assignment. We believe the 20 Newsgroups dataset was a challenge because text data tends to be sparse. For future experiments, we recommend using a text embedding to preprocess text data such as word2vec [20] or GloVe [22]. This maps each word to a space in which distance corresponds to word similarity, and should make the data more dense.

In the future we hope to extend BELA to practical applications such as the labeling of social media data or other data that is difficult to label. Many real datasets tend to include errors in labeled data simply because a human is labeling the data and may make mistakes. Our method assumes all labeled data is clean and can be trusted as 100% correct. More analysis is necessary to compute our bound $\tilde{f}$ assuming some percentage of data is incorrect. It would also be useful to be able to correct mistakes if we discover an error in labelling after we have performed several splits of the dataset. Perhaps it is not appropriate to "undo" a split

performed early on in our algorithm, but instead provide the user with a more pessimistic bound $\tilde{f}$ instead.

We suggest more theoretical analysis on the iterative splitting and labeling of datasets. An analysis on the geometry of the data may lead to better theoretical guarantees on the resulting dataset. For example, coreset analysis (Agarwal et al. [1]) may be useful for providing geometric approximations on the label assignment based off of the labels obtained by the oracle.

Bibliography

[1] Pankaj K Agarwal, Sariel Har-Peled, and Kasturi R Varadarajan. Geometric approximation via coresets. Combinatorial and computational geometry, 52:1–30, 2005.

[2] Chidubem Arachie and Bert Huang. Adversarial label learning. In Proceedings of the AAAI Conference on Artificial Intelligence, 2019.

[3] Rémi Bardenet, Odalric-Ambrym Maillard, et al. Concentration inequalities for sampling without replacement. Bernoulli, 21(3):1361–1385, 2015.

[4] Doug Beeferman and Adam Berger. Agglomerative clustering of a search engine query log. In KDD, volume 2000, pages 407–416, 2000.

[5] Alina Beygelzimer, Sanjoy Dasgupta, and John Langford. Importance weighted active learning. In Proceedings of the 26th Annual International Conference on Machine Learning, ICML '09, pages 49–56, 2009. ISBN 978-1-60558-516-1.

[6] Olivier Chapelle, Bernhard Scholkopf, and Alexander Zien. Semi-supervised learning (chapelle, o. et al., eds.; 2006)[book reviews]. IEEE Transactions on Neural Networks, 20(3):542–542, 2009.

[7] David A Cohn, Zoubin Ghahramani, and Michael I Jordan. Active learning with statistical models. Journal of Artificial Intelligence Research, 4:129–145, 1996.

[8] Thomas H Cormen, Charles E Leiserson, Ronald L Rivest, and Clifford Stein. Introduction to algorithms. MIT press, 2009.

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

[10] Li Deng. The MNIST database of handwritten digit images for machine learning research [best of the web]. IEEE Signal Processing Magazine, 29(6):141–142, 2012.

[11] empty. 20 newsgroups dataset, empty. URL http://people.csail.mit.edu/jrennie/20Newsgroups/.

[12] Wassily Hoeffding. Probability inequalities for sums of bounded random variables. In The Collected Works of Wassily Hoeffding, pages 409–426. Springer, 1994.

[13] Sheng-Jun Huang, Rong Jin, and Zhi-Hua Zhou. Active learning by querying informative and representative examples. In Advances in Neural Information Processing Systems, pages 892–900, 2010.

[14] Thorsten Joachims. A probabilistic analysis of the rocchio algorithm with tfidf for text categorization. Technical report, Carnegie-mellon univ pittsburgh pa dept of computer science, 1996.

[15] Ajay J Joshi, Fatih Porikli, and Nikolaos Papanikolopoulos. Multi-class active learning for image classification. In 2009 IEEE Conference on Computer Vision and Pattern Recognition, pages 2372–2379. IEEE, 2009.

[16] Dave Kale. active-transfer. https://github.com/turambar/active-transfer, December 2013.

[17] Donald Ervin Knuth. The Art of Computer Programming: Sorting and Searching, volume 3. Pearson Education, 1997.

[18] Mingkun Li and Ishwar K Sethi. Confidence-based active learning. IEEE Transactions on Pattern Analysis and Machine Intelligence, 28(8):1251–1261, 2006.

[19] Colin McDiarmid. Concentration. In Probabilistic Methods for Algorithmic Discrete Mathematics, pages 195–248. Springer, 1998.

[20] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781, 2013.

[21] 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.

[22] Jeffrey Pennington, Richard Socher, and Christopher Manning. Glove: Global vectors for word representation. In Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP), pages 1532–1543, 2014.

[23] Alexander J Ratner, Christopher M De Sa, Sen Wu, Daniel Selsam, and Christopher Ré. Data programming: Creating large training sets, quickly. In Advances in Neural Information Processing Systems, pages 3567–3575, 2016.

[24] Burr Settles and Mark Craven. An analysis of active learning strategies for sequence labeling tasks. In Proceedings of the Conference on Empirical Methods in Natural Language Processing, pages 1070–1079. Association for Computational Linguistics, 2008.

[25] Christopher Tosh and Sanjoy Dasgupta. Interactive structure learning with structural query-by-committee. In Advances in Neural Information Processing Systems, pages 1121–1131, 2018.

[26] Ruth Urner, Sharon Wulff, and Shai Ben-David. Plal: Cluster-based active learning. In Conference on Learning Theory, pages 376–397, 2013.

[27] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. arXiv preprint arXiv:1708.07747, 2017.

[28] Shengjia Zhao, Enze Zhou, Ashish Sabharwal, and Stefano Ermon. Adaptive concentration inequalities for sequential decision problems. In Advances in Neural Information Processing Systems, pages 1343–1351, 2016.