Improved Active Covering via Density-Based Space Transformation

MohammadHossein Bateni
Google Research
New York City, New York, USA
bateni@google.com

Hossein Esfandiari
Google Research
London, UK
esfandiari@google.com

Samira HosseinGhorban
Institute for Research in Fundamental Sciences
School of Computer Science
Tehran, Iran
s.hosseinghorban@ipm.ir

Alipasha Montaseri
Sharif University of Technology
Tehran, Iran
apmontaseri@ce.sharif.edu

ABSTRACT

In this work, we study active covering, a variant of the active-learning problem that involves labeling (or identifying) all of the examples with a positive label. We propose a couple of algorithms, namely *Density-Adjusted Non-Adaptive (DANA) learner* and *Density-Adjusted Adaptive (DAA) learner*, that query the labels according to a distance function that is adjusted by the density function. Under mild assumptions, we prove that our algorithms discover all of the positive labels while querying only a sublinear number of examples from the support of negative labels for constant-dimensional spaces (see Theorems 5 and 6). Our experiments show that our champion algorithm DAA consistently improves over the prior work on some standard benchmark datasets, including those used by the previous work, as well as a couple of data sets on credit card fraud. For instance, when measuring performance using AUC, our algorithm is the best in 25 out of 27 experiments over 7 different datasets.

CCS CONCEPTS

• **Computing methodologies** → **Active learning settings**; *Online learning settings*; *Batch learning*; • **Theory of computation** → **Sketching and sampling**.

KEYWORDS

Active Covering, Active Learning, Crowdsourcing.

ACM Reference Format:
MohammadHossein Bateni, Hossein Esfandiari, Samira HosseinGhorban, and Alipasha Montaseri. 2024. Improved Active Covering via Density-Based Space Transformation. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '24), August 25–29, 2024, Barcelona, Spain.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3637528.3671794

KDD '24, August 25–29, 2024, Barcelona, Spain

ACM ISBN 979-8-4007-0490-1/24/08.
https://doi.org/10.1145/3637528.3671794

1 INTRODUCTION

It is widely recognized that data collection often involves costs. One example is when we are using crowdsourcing to collect data for machine learning tasks such as spam or fraud detection. This fact motivates a class of machine-learning tasks called active learning. In an active-learning task, we intend to indicate a limited set of valuable examples (iteratively, or in batches) and probe their labels in order to learn a model. Active covering is a variant of active learning, in the context of binary classification, where the set of examples with the positive label is considered valuable, hence we intend to probe all such examples. One instance of active covering is disease testing in a new pandemic such as Covid-19. In this case, we desire to identify and test all patients with Covid-19, while we do not like to waste many tests on patients without Covid-19.

Previously, Jiang and Rostamizadeh [17] studied the active-covering problem and proposed and analyzed a couple of algorithms called offline learner and active learner. Both algorithms are based on first sampling a set of examples to query and then querying the unlabeled examples close to the examples with a positive label. The only distinction between the two algorithms is that the offline learner exclusively considers the examples with a positive label in the initial samples and queries nearby examples whereas the active learner considers all queried examples with a positive label. They show that under the following four assumptions, their algorithms label all of the positive examples while querying only a sublinear number of examples from the support of the negative examples. First, a fixed lower bound on the density function of the positive labels in the support of the positive labels is assumed. Second, for every small ball around a point with a positive label, a constant fraction of the volume of the ball falls inside the support of the positive examples. Third, there is an upper bound on the density of the negative examples. Moreover, in their theorem statements, they use a parameter C which actually depends linearly on the surface area of the support of the positive examples. Hence, by hiding C in the $O(.)$ notation, they implicitly assume that the surface area of the support of the positive examples is bounded by a constant.

It is evident that both the offline learner and active learner algorithms are highly sensitive to the density of the distribution of the examples. For instance, in a scenario where the probability of observing positive labels divided by the probability of observing negative labels is uniform across the space, both the offline learner

Figure 1: One feature

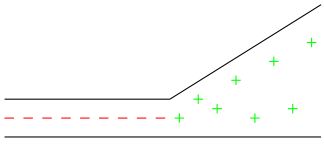

Figure 2: Two features

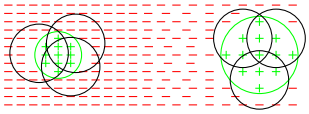

Figure 3: A learner that is not density adjusted.

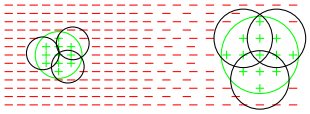

Figure 4: A density adjusted learner.

and active learner algorithms tend to query the areas with a higher density at a substantially higher rate and subsequently move to areas with a lower density, which is a counter-intuitive behavior. This is why it is necessary for these algorithms to have a *global* upper bound on the probability distribution of the negative examples and a *global* lower bound on the probability distribution of positive labels in the support of positive labels. Such global constraints on the probability distributions are not very desirable since in many instances we observe several colonies of points in some dense areas and some large and sparse pools of points around them.

Another issue is that the previous assumptions are very sensitive to feature selection, and adding one unnecessary or mildly relevant feature can break the assumptions. For example, Figure 1 is a very well-formed example and respects the assumptions of the previous work. However, adding one relevant but unnecessary feature turns it into Figure 2 which has very sparse areas with positive examples and no longer respects the assumptions of the previous work.

To address the aforementioned issues, we implicitly utilize a space transformation that expands the regions with high density in the probability distribution of the examples and shrinks the regions with low density, with the aim of achieving a more uniform probability distribution. This transformation allows us to remove the global upper and lower bounds on the probability distributions and instead use a local constraint that avoids a sudden change in the probability distribution in a small ball. This allows the probability distribution to smoothly change from one place in the dataset to another and hence accept inputs such as Figure 2. We adopt the offline learner and active learner in this transformed space. We call our algorithms *Density-Adjusted Non-Adaptive (DANA) learner* and *Density-Adjusted Adaptive (DAA) learner*, respectively. We extend the approach of Jiang and Rostamizadeh to prove that our algorithms label all of the positive examples while querying only a sublinear number of examples with negative labels.

We conduct experimental comparisons between our algorithms, the offline learner, the active learner, and the uniform sampler. Remarkably, in almost all of our experiments, our proposed algorithm DAA consistently emerges as the best algorithm in terms of labeling cost. For example, when we measure the performance via the AUC, our algorithm achieves the best performance in 25 out of 27 experiments over 7 different datasets.

Figures 3 and 4 give some intuitions on why our density-adjusted algorithms work better than the previous Euclidean-based algorithms. Note that, both Figures 3 and 4 represent the same instance. In this instance, there are two discs of positive points inside a pool of negative points. However, the density of the left side is higher than the right side. A Euclidean-based algorithm probes some discs of the same radius to find all of the positive points, regardless of the density function (see Figure 3). However, our density-adjusted algorithms probe the labels more cautiously in the denser areas via smaller discs (see Figure 4). Hence, a fewer number of negative points fall in the discs that a density-adjusted algorithm probes.

1.1 Problem Setting

In this section, we define the active-covering problem and our setting. There exists an unknown probability density function $f : \mathbb{R}^D \to \mathbb{R}$ representing examples with binary labels. We use $\mathcal{X} \subseteq \mathbb{R}^D$ to refer to the support of f, i.e., the set of all possible examples. We refer to the examples with label 1 as positive examples and to the examples with label 0 as negative examples. Accordingly, we have unknown density functions $f_+ : \mathbb{R}^D \to \mathbb{R}$ with support $\mathcal{X}_+$ and $f_- : \mathbb{R}^D \to \mathbb{R}$ with support $\mathcal{X}_-$, corresponding to the positive and negative examples respectively. We denote by $\mathcal{P}$, $\mathcal{P}_+$ and $\mathcal{P}_-$ the probability distributions corresponding to f, f_+ and f_-, respectively. Hence for a set $A \subseteq \mathbb{R}^D$, for example we have $\mathcal{P}(A) = \int_A f\, dA$.

For an arbitrary set $A \subseteq \mathcal{X}$, by the law of total probability [27, Page. 9] the probability of A is equal to

$$\mathcal{P}(A) = \mathcal{P}(\mathcal{X}_+)\mathcal{P}(A|\mathcal{X}_+) + \mathcal{P}(\mathcal{X}_-)\mathcal{P}(A|\mathcal{X}_-).$$

Let $p := \mathcal{P}(\mathcal{X}_+)$, then we have $\mathcal{P}(\mathcal{X}_-) = 1 - p$. Thus, for any $A \subseteq \mathcal{X}$ we have

$$\mathcal{P}(A) = p\mathcal{P}_+(A) + (1-p)\mathcal{P}_-(A).$$

We receive a set of unlabeled examples $X \subseteq \mathcal{X}$, with size n drawn i.i.d. from probability distribution $\mathcal{P}$. We are allowed to query the label of the examples in X. The goal is to discover all positive examples while minimizing the number of queries. We use X_+ and X_- to refer to the true set of positive and negative examples in X, respectively.

Similar to the prior work [17], we compare our algorithms to the optimal performance achievable by an algorithm that knows the support of $\mathcal{X}_+$, which requires $n\mathcal{P}(\mathcal{X}_+)$ queries in expectations. Note that, if a point x belongs to the support of $\mathcal{X}_+$, even though we may have some information about the probability that x has a negative label, it still may have a positive label and hence any algorithm is forced to query it. Hence, any algorithm requires to make at least $n\mathcal{P}(\mathcal{X}_+)$ queries. We refer to this as Q_{OPT}.

Definition 1 (Excess Query Cost). *We define the excess query cost of an algorithm A, denoted by C_A, to be $Q_A - Q_{OPT}$, where Q_A is the number of queries made by algorithm A to label all the positive examples.*

It is not difficult to observe that, without any structural assumption on the relationship among the positive examples, querying almost all of the examples is necessary to retrieve the outlier positive examples. Therefore, to avoid having to locate positive examples in extremely narrow and sparse subspaces, some assumptions on the distribution of the positive examples are necessary [10, 17, 32].

Our assumptions here are similar to those of Jiang and Rostamizadeh [17] in nature. However, there are some technical differences that make our assumptions less restrictive. For example, instead of global upper or lower bounds on the density function, we require the change in the density function to be bounded in every small ball. Another example is that when they refer to a small ball they refer to a ball with a small radius, while we refer to a ball with a small probability. However, roughly speaking, since they have a lower bound on the density as well, a ball with a small probability will have a small radius as well, but not vice versa.

The following assumption ensures that there are no outlier positive examples. In other words, wherever a positive example exists, a nontrivial fraction of the examples surrounding it are also positive.

Assumption 2. *The support of positive examples is a compact subspace and a disjoint union of a finite number of connected components*[1] $\mathcal{X}_{+,1}, \ldots, \mathcal{X}_{+,c}$. *Moreover, there exist* ϕ_0 *and* $C_+ \in (0, 1]$ *such that for any point* $x \in \mathcal{X}_+$ *and positive number* ϵ *that satisfy* $\mathcal{P}(B(x, \epsilon)) \leq \phi_0$, *we have*

$$\mathcal{P}_+(B(x, \epsilon)) \geq C_+ \mathcal{P}(B(x, \epsilon)),$$

where $B(x, \epsilon)$ *is a ball with radius* ϵ *and center* x *with respect to the Euclidean metric.*

The next assumption says that the density function f does not significantly change in a small ball of its domain.

Assumption 3. *Let* ϕ_0 *be the parameter set in Assumption 2. For a point* $x \in \mathcal{X}$, *we define* λ_x *and* μ_x *to respectively be the minimum and maximum of* f *in* $B(x, r)$, *where* r *is chosen such that* $\mathcal{P}(B(x, r)) = \phi_0$. *We assume that* $\frac{\lambda_x}{\mu_x} \geq \alpha$, *for a constant* α.

The following assumption ensures that the support of $\mathcal{X}_+$ is not an excessively narrow subspace scattered throughout the domain of $\mathcal{X}_-$.

Assumption 4. *Let* ϕ_0 *be the parameter set in Assumption 2. There exists a constant* $C_{\mathcal{X}_+} > 0$ *such that for all* $\phi \in [0, \phi_0]$,

$$\mathcal{P}_-\left(\cup_{x \in \mathcal{X}_+} B\left(x, \sqrt[D]{\frac{\phi}{\mu_x v_D}}\right)\right) \leq \sqrt[D]{\phi} C_{\mathcal{X}_+},$$

where v_D *is the volume of a unit ball in a* D *dimensional space.*

The previous work requires a similar assumption to Assumption 4, that they do not explicitly mention. Specifically, they require the surface area of the domain of positive examples to be bounded. In fact, they use a parameter C in their excess query cost that *"depends on* $\mathcal{P}$*"*. The parameter C actually depends linearly on the surface area of the domain of positive examples and hence enforces it to be bounded.

1.2 Our Contributions

As our first result, we bound the excess query cost of our density-adjusted non-adaptive algorithm (i.e., Algorithm 1). This result is presented in Section 2.

[1] A set $\mathcal{A} \subseteq \mathbb{R}^D$ is connected if and only if for each pair of points $x, x' \in \mathcal{A}$, there is a curve from x to x', which completely lies in $\mathcal{A}$.

Theorem 5. *Suppose that Assumptions 2, 3, and 4 hold. For some* $m \in \widetilde{\Theta}(n^{D/(D+1)})$, *we have*

$$\mathbb{E}[C_{DANA}] \leq \widetilde{O}\left(n^{\frac{D}{D+1}}\right).$$

Next, we bound the excess query cost of our density-adjusted adaptive algorithm (i.e., Algorithm 2). This result is presented in Section 3.

Theorem 6. *Suppose that Assumptions 2, 3 and 4 hold. For some* $m \in \tilde{\Theta}(n^{(D-1)/D})$, *we have*

$$\mathbb{E}[C_{DAA}] \leq \tilde{O}(n^{\frac{D-1}{D}}).$$

Remark 7. *Both our algorithms assume that we know the density function* $f(\cdot)$ *of our examples. In Section 4 we show how to estimate the density function for our purpose.*

The previous work relies on explicitly knowing the number of positive examples and clarified that *"It's worth noting that we may not know when all of the positive examples are labeled– thus, in practice, we can terminate the algorithm when enough positives are found depending on the task or when the labeling budget runs out"* [17]. We resolve this issue from the theoretical perspective and provide a simple and asymptotically optimal termination condition that applies both to our algorithms and the algorithms of the previous work. This condition is provided in Section 5.

Finally, in section 6 we report our experimental study on datasets used by the previous work, as well as Mini-Imagenet and a couple of datasets on credit card fraud. We provide further experimental details in the appendix. Our experiments show that our DAA algorithm consistently improves over the previous work. Due to the space limit, we provide some of the proofs in the appendix.

1.3 Preliminaries

Subsequently, we present some definitions and lemmas that we use throughout the paper.

Definition 8. *Let* $d(\,,\,)$ *denote the Euclidean distance. Let* $\mathcal{A}, \mathcal{B} \subseteq \mathbb{R}^D$ *be non-empty sets and* $x \in \mathbb{R}^D$ *and* $\epsilon > 0$.

(1) *An* ϵ*-ball with center* x *is defined as*

$$B(x, \epsilon) = \left\{y \in \mathbb{R}^D | d(x, y) \leq \epsilon\right\}.$$

(2) *An* ϵ*-tubular neighborhood around the set* $\mathcal{A}$ *is defined as*

$$B(\mathcal{A}, \epsilon) = \left\{x \in \mathbb{R}^D \middle| \inf_{x' \in \mathcal{A}} d(x, x') \leq \epsilon\right\}.$$

(3) *The distance* x *from* $\mathcal{A}$ *is defined as*

$$d(x, \mathcal{A}) = \min_{y \in \mathcal{A}} d(x, y).$$

Definition 9. *Let* $\mathcal{A} \subseteq \mathbb{R}^D$, $S \subset \mathcal{A}$ *and* $\phi > 0$. *The set* $\mathcal{A}$ *is connected in the* ϕ*-neighborhood graph of* S *if and only if for each* $x, x' \in \mathcal{A}$, *there is a path* $x_1 = x \to x_2 \to \ldots \to x_\ell = x'$ *where for each* $1 < j < \ell$ *we have* $x_j \in S$ *and* $x_j \in B(x_{j-1}, r_{x_{j-1}})$ *where* $r_{x_{j-1}}$ *is selected such that* $\mathcal{P}\left(B(x_{j-1}, r_{x_{j-1}})\right) = \phi$.

We use $\mathrm{Vol}(B)$ to refer to the volume of a multi-dimensional ball B in Euclidean space. We use $Pr(E)$ to indicate the probability of an event E. We also use the following theorem from [6] to establish an upper bound on the number of our queries.

Theorem 10. *Let X be a set of i.i.d samples with size n drawn from a distribution $\mathcal{P}$. For $0 < \delta < 1$, there exists a universal constant C_0 such that with probability at least $1 - \delta$ uniformly over all balls $B \subseteq \mathbb{R}^D$, we have*

$$\mathcal{P}(B) \geq \frac{C_0 D \log \frac{2}{\delta} \log n}{n} \Rightarrow |B \cap X| > 0.$$

1.4 Other Related Work

The early studies on active learning date back to the 1990s [8, 9]. However, due to its significant role in machine learning tasks, there is still a lot of interest in developing more practical and effective active learning mechanisms [1, 7, 21, 26, 30].Active learning has been used in several learning tasks such as image processing [12, 20], fight against COVID-19 [30], text classification [31] and speech recognition [14]

Active covering as a variant of active learning appears in several machine learning tasks. For example active covering in useful in credit card fraud detection [2],computational drug discovery [29], bank loan applications [22], moderate abusive content[28], fake account detection in social network platforms [24], and, distances and uncontrollable situations such as the COVID-19 pandemic [34].

Garnett et al. study active search in order to retrieve as many positive labels as possible given a query budget [13]. Jiang et al. [18] provide a more time-efficient algorithm for this problem. Active search has also been formalized as a bandit problem in a few previous works [16, 19]. The active covering problem considers a more aggressive formulation compared to that of active search and attempts to find all (or practically almost all) of the positive labels. This is particularly important in sensitive situations such as credit card fraud and providing tests during a pandemic.

Retrieving the positive examples is also known as learning under one-sided feedback which was studied by Helmold et al. [15]. They used the standard online model, where the learning algorithm tries to minimize the failure. It is worth mentioning that active covering is related to the main-stream research path including the online learning tasks which are investigated widely [4, 5, 25] and the classical set-cover problem [33]. Our techniques have a connection to the support estimation literature [3, 10, 11, 23, 35], even though, these works do not directly consider the active search problem.

2 DENSITY-ADJUSTED NON-ADAPTIVE LEARNER

Algorithm 1 Density-adjusted Non-Adaptive Algorithm (DANA)

Input Dataset X, initial sample size m and density function f.

1: Let X_0 be m examples sampled uniformly without replacement from X.
2: Label query X_0 and let $X_{+,0}$ be the positive examples.
3: Label query remaining examples x in ascending order of $d(x, X_{+,0}) \cdot \sqrt[D]{f(x)}$ until all positive examples are labeled.

Initially, DANA learner uniformly samples a set with m points from the dataset X and queries their labels. Subsequently, it labels the remaining examples of X in ascending order of their minimum density-adjusted distance to the initially sampled positive examples until all positive examples are retrieved. Now, we are ready to prove Theorem 5.

Proof of Theorem 5. To prove, we show that for any $0 < \delta < 1$, if $m \geq \max\left\{\frac{2\log\frac{2}{\delta}}{p^2}, \frac{2C_0 D \log\frac{2}{\delta}\log(mp/2)}{p^2 C_+ \alpha \phi_0}\right\}$, we have

$$\mathbb{E}[C_{\text{DANA}}] \leq (1-p)m + nC\left(\frac{\log(2/\delta)\log(pm/2)}{m}\right)^{1/D} + \delta n,$$

where $C = \left(\frac{2(1-p)C_{\mathcal{X}_+}C_0 D}{C_+ \alpha p^2}\right)^{\frac{1}{D}}$. By setting $m = \tilde{\Theta}(n^{D/(D+1)})$ and $\delta = \frac{1}{n}$, we obtain

$$\mathbb{E}[C_{\text{DANA}}] \leq \widetilde{O}\left(n^{D/(D+1)}\right).$$

Now, to prove the above claim we define binary random variables $Y_1, \ldots, Y_m$ to represent the labels of the examples that Algorithm 1 queries. Note that the probability of Y_i being 1 is p, i.e., $Pr(Y_i = 1) = p$. Thus the expected number of positive examples in the initial sample X_0 is $\mathbb{E}[\sum_{i=1}^m Y_i] = mp$. Since $Y_1, \ldots, Y_m$ are independent random variables, Hoeffding inequality gives

$$Pr\left(\sum_{i=1}^m Y_i - mp \leq -t\right) \leq e^{\frac{-2t^2}{m}},$$

for any arbitrary $t > 0$. Equivalently, we have

$$Pr\left(\sum_{i=1}^m Y_i - mp > -t\right) > 1 - e^{\frac{-2t^2}{m}}.$$

We set $t = \sqrt{\frac{m}{2}\log\frac{2}{\delta}}$ in the above to obtain the following lower bound on the number of positive labels in X_0, with probability at least $1 - \frac{\delta}{2}$:

$$\sum_{i=1}^m Y_i \geq mp - t = m(p - t/m) = m\left(p - \sqrt{\frac{1}{2m}\log\frac{2}{\delta}}\right).$$

Applying $m \geq \frac{2\log\frac{2}{\delta}}{p^2}$ gives us $\sum_{i=1}^m Y_i \geq mp/2$ with probability $1 - \frac{\delta}{2}$. Define $\phi = \frac{2C_0 D \log\frac{2}{\delta}\log(mp/2)}{p^2 C_+ \alpha m}$. Note that since $m \geq \frac{2C_0 D \log\frac{2}{\delta}\log(mp/2)}{p^2 C_+ \alpha \phi_0}$, we have $\phi_0 \geq \frac{2C_0 D \log\frac{2}{\delta}\log(mp/2)}{p^2 C_+ \alpha m}$, hence we have $\phi \leq \phi_0$. Therefore, for each $x \in \mathcal{X}_+$, the definition of μ_x implies $\mathcal{P}(B(x, \sqrt[D]{\frac{\phi}{\mu_x v_D}})) \leq \phi_0$. We claim that with probability $1 - \delta$, for each $x \in \mathcal{X}_+$, there exists a positive example in $B(x, \sqrt[D]{\frac{\phi}{\mu_x v_D}})$ that is queried in $X_{0,+}$, i.e., $B(x, \sqrt[D]{\frac{\phi}{\mu_x v_D}}) \cap X_{0,+} \neq \emptyset$. To see this, let us calculate the probability mass of positive examples in this ball:

$$p\mathcal{P}_+\left(B(x, \sqrt[D]{\frac{\phi}{\mu_x v_D}})\right) \geq pC_+\mathcal{P}\left(B(x, \sqrt[D]{\frac{\phi}{\mu_x v_D}})\right)$$

$$\geq pC_+\lambda_x \text{Vol}(B(x, \sqrt[D]{\frac{\phi}{\mu_x v_D}})) \geq pC_+\lambda_x v_D \frac{\phi}{\mu_x v_D} \geq pC_+\alpha\phi.$$

Combined with the lower bound $m \geq \frac{2C_0 D \log\frac{2}{\delta}\log(mp/2)}{p^2 C_+ \alpha\phi}$, Theorem (10) guarantees that one example from $X_{0,+}$ falls in this ball. Hence, all positive examples are retrieved by Algorithm (1). Next, we upper bound the excess query cost of this algorithm. The initial sample set X_0 contains $(1-p)m$ negative examples in expectation. Moreover, the number of negative examples in $X \setminus X_0$ is

$$(n-m)(1-p)\mathcal{P}_-\big(\cup_{x\in\mathcal{X}_{,+}} B(x, \sqrt[D]{\frac{\phi}{\mu_x v_D}})\big) \leq (n-m)(1-p)C_{\mathcal{X}_+}\sqrt[D]{\phi}$$

where the inequality follows from Assumptions 3 and 4. With probability δ our concentration bounds fail and in that case, the excess query cost is at most n otherwise our expected query cost is upper bounded by

$$(1-p)m + (n-m)(1-p)C_{\mathcal{X}_+}\sqrt[D]{\phi} \leq (1-p)m + n\Big(\frac{2(1-p)C_{\mathcal{X}_+}C_0 D}{C_+\alpha p^2}\Big)^{\frac{1}{D}}\Big(\frac{\log\left(\frac{2}{\delta}\right)\log\left(\frac{mp}{2}\right)}{m}\Big)^{\frac{1}{D}}.$$

Hence we have

$$\mathbb{E}[C_{\text{DANA}}] \leq \delta n + (1-p)m + n\Big(\frac{2(1-p)C_{\mathcal{X}_+}C_0 D}{C_+\alpha p^2}\Big)^{\frac{1}{D}}\Big(\frac{\log\left(\frac{2}{\delta}\right)\log\left(\frac{mp}{2}\right)}{m}\Big)^{\frac{1}{D}}. \qquad \square$$

3 DENSITY-ADJUSTED ADAPTIVE LEARNER

Initially, DAA learner uniformly samples a set with m points from the dataset X and queries their labels as same as DANA. Subsequently, it labels the remaining examples of X in ascending order of their minimum density-adjusted distance to the current sample points of X whose positive labels are revealed until all positive examples are retrieved. In summary, the key difference between the DANA and DAA is in ascending order of the remaining examples in X. In DANA, the ordering is done based on the initial positive sample while in DAA it is based on the updated ones.

Algorithm 2 Density-adjusted Adaptive Algorithm (DAA)

Input Dataset X, initial sample size m and density function f.

1: Let X_0 be m examples sampled uniformly without replacement from X.
2: Label query X_0 and let $X_{+,0}$ be the positive examples.
3: Initialize $X_p \leftarrow X_{+,0}$ and $X_a \leftarrow X_0$.
4: **while** not all positive examples in X are labeled **do**
5: Label query $x = \arg\min_{x\in X\setminus X_a} d(x, X_p)\cdot\sqrt[D]{f(x)}$
6: **if** x has a positive label **then**
7: $X_p \leftarrow X_p \cup \{x\}$
8: **end if**
9: $X_a \leftarrow X_a \cup \{x\}$
10: **end while**

In the next lemma, first, we derive some conditions on m to guarantee, with high probability, that at least one example is chosen from each connected component of $\mathcal{X}_+$. Next, a bound on ϕ is proposed such that $\mathcal{X}_{+,i}\cap X_+$ is connected in the ϕ-neighborhood graph of X_+ w.h.p. This ensures that our algorithm probes all of the positive examples in X_+ via some paths through the ϕ-neighborhood graph.

Lemma 11. *Let Assumptions 2 and 3 hold and* $0 < \delta < 1$.

(1) *Define* $q = \min_{1\leq i\leq c}\mathcal{P}_+(\mathcal{X}_{+,i})$. *If*

$$m \geq \max\left\{\frac{2\log\left(\frac{2c}{\delta}\right)}{p\log\left(\frac{1}{1-q}\right)}, \frac{2\log\left(\frac{2}{\delta}\right)}{p^2}\right\},$$

then for all $1 \leq i \leq c$, *we have* $\mathcal{X}_{+,i}\cap X_0 \neq \emptyset$, *with probability at least* $1-\delta$.

(2) *Let*

$$\phi = \frac{3^D C_0 D\log\frac{2\times 3^D n^2}{\delta\alpha^4\phi}\log n}{pC_+\alpha^4 n},$$

where n *is chosen sufficiently large such that* $\phi \leq \phi_0$. *Then for all* $1 \leq i \leq c$, $\mathcal{X}_{+,i}\cap X_+$ *is connected in the* ϕ*-neighborhood graph of* X_+, *with probability at least* $1-\delta$.

Proof. For the first part, we define binary random variables $Y_1, \ldots, Y_m$ to be the labels of the examples that Algorithm 2 queries. Note that the probability of Y_i being 1 is p, i.e., $Pr(Y_i = 1) = p$. Thus the expected value of $\sum_{i=1}^m Y_i$ is mp which represents the expected number of positive examples in the initial sample X_0 of Algorithm (2). Since $Y_1, \ldots, Y_m$ are independent random variables, by Hoeffding inequality, we have

$$Pr\Big(\sum_{i=1}^m Y_i - mp \leq -t\Big) \leq e^{\frac{-2t^2}{m}}$$

for any arbitrary $t > 0$. Equivalently, we have

$$Pr\Big(\sum_{i=1}^m Y_i - mp > -t\Big) > 1 - e^{\frac{-2t^2}{m}}.$$

Let us set $t = \sqrt{\frac{m}{2}\log\frac{2}{\delta}}$. By applying this to the above inequality, with probability at least $1-\frac{\delta}{2}$ we can lower bound the number of positive examples in the initial sample set X_0 as follows.

$$\sum_{i=1}^m Y_i \geq mp - t = m(p - \frac{t}{m}) = m\left(p - \sqrt{\frac{1}{2m}\log\frac{2}{\delta}}\right).$$

Note that from the statement of the lemma we have $m \geq \frac{2\log(\frac{2}{\delta})}{p^2}$. Applying this to the above inequality gives us $|X_+| \geq m\frac{p}{2}$ with probability $1-\frac{\delta}{2}$. Moreover, for each $1 \leq i \leq c$, the probability that none of these $mp/2$ are in $\mathcal{X}_{+,i}$ is $\left(1-\mathcal{P}_+(\mathcal{X}_{+,i})\right)^{m\frac{p}{2}} \leq (1-q)^{m\frac{p}{2}}$. Let $m \geq \frac{2\log\left(\frac{2c}{\delta}\right)}{p\log\left(\frac{1}{1-q}\right)}$, then we have $(1-q)^{m\frac{p}{2}} \leq \frac{\delta}{2c}$. Hence, for each i we have $\mathcal{X}_{+,i}\cap X_0 \neq \emptyset$ with probability at least $1-\delta/2c$. Applying a simple union bound for all i gives us $\mathcal{X}_{+,i}\cap X_0 \neq \emptyset$ with probability at least $1-\delta/2$. This completes the proof of the first part.

Now, for the second part, pick two arbitrary points $x, y \in \mathcal{X}_{+,i}$ for some $i \in \{1, \cdots, c\}$. Since $\mathcal{X}_{+,i}$ is connected, there is a carve between x, y in $\mathcal{X}_{+,i}$. We consider a sequence of points on this carve

namely, $x = x_1 \to x_2 \to \cdots \to x_\ell = y$ such that for every x_j, we have

$$x_{j+1} \in B\Big(x_j, \frac{r_{x_j}\sqrt[D]{\alpha^3}}{3}\Big),$$

where radius r_{x_j} is set such that $\mathcal{P}\big(B(x_j, r_{x_j})\big) = \phi$. Next, we use this to show that, with high probability, there exists a path $x = x_1 \to x'_1 \to x'_2 \to \cdots \to x'_\ell \to x_\ell = y$ in $\mathcal{X}_{+,i} \cap X_+$, where for every x'_j, we have $x'_{j+1} \in B(x'_j, r'_{x'_j})$, where radius $r'_{x'_j}$ is set such that $\mathcal{P}\big(B(x'_j, r'_{x'_j})\big) = \phi$. This means that x and y are connected in the ϕ-neighborhood graph of X_+ as desired. Next, we show that for every j the ball $B\Big(x_j, \frac{r_{x_j}\sqrt[D]{\alpha^3}}{3}\Big)$ contains at least one positive example in data set X, w.h.p. Recall that we have $\mathcal{P}\big(B(x_j, r_{x_j})\big) = \phi$ and hence we have $\sqrt[D]{\frac{\phi}{v_D \mu_{x_j}}} \le r_{x_j}$. To prove our claim, we calculate the probability mass of positive examples in

$$B\Big(x_j, \frac{\sqrt[D]{\alpha^3}}{3}\sqrt[D]{\frac{\phi}{v_D\mu_{x_j}}}\Big).$$

By Assumption 2 we have

$$\begin{aligned}
&p\mathcal{P}_+\Big(B(x_j, \tfrac{\sqrt[D]{\alpha^3}}{3}\sqrt[D]{\tfrac{\phi}{v_D\mu_{x_j}}})\Big)\\
&\ge pC_+\mathcal{P}\Big(B(x_j, \tfrac{\sqrt[D]{\alpha^3}}{3}\sqrt[D]{\tfrac{\phi}{v_D\mu_{x_j}}})\Big)\\
&\ge pC_+\lambda_{x_j}\mathrm{Vol}(B(x_j, \tfrac{\sqrt[D]{\alpha^3}}{3}\sqrt[D]{\tfrac{\phi}{v_D\mu_{x_j}}}))\\
&\ge pC_+\lambda_{x_j}\frac{\alpha^3\phi}{3^D\mu_{x_j}} \ge pC_+\alpha^4\frac{\phi}{3^D}\\
&\ge \frac{C_0 D\log\big(\frac{2n^2}{\delta\phi}\big)\log n}{n}.
\end{aligned}$$

Thus, by Theorem (10), with probability at least $1 - \frac{\delta\alpha^4\phi}{3^D n^2}$, there exists a point $x'_j \in B(x_j, \frac{\sqrt[D]{\alpha^3}}{3}\sqrt[D]{\frac{\phi}{v_D\mu_{x_j}}}) \cap X_+$. Note that we have

$$\begin{aligned}
&d(x'_j, x'_{j+1}) \le d(x'_j, x_j) + d(x_j, x_{j+1}) + d(x_{j+1}, x'_{j+1})\\
&\le \frac{\sqrt[D]{\alpha^3}}{3}\big(r_{x_j} + r_{x_j} + r_{x_{j+1}}\big)\\
&\le \frac{\sqrt[D]{\alpha^3}}{3}\Big(\sqrt[D]{\frac{\phi}{v_D\lambda_{x_j}}} + \sqrt[D]{\frac{\phi}{v_D\lambda_{x_j}}} + \sqrt[D]{\frac{\phi}{v_D\lambda_{x_{j+1}}}}\Big)\\
&\le \frac{\sqrt[D]{\alpha^3}}{3}\Big(\sqrt[D]{\frac{\phi}{v_D\lambda_{x_j}}} + \sqrt[D]{\frac{\phi}{v_D\lambda_{x_j}}} + \sqrt[D]{\frac{\phi}{\alpha v_D\lambda_{x_j}}}\Big)\\
&\le \frac{\sqrt[D]{\alpha^3}}{3}\Big(\sqrt[D]{\frac{\phi}{\alpha v_D\mu_{x_j}}} + \sqrt[D]{\frac{\phi}{\alpha v_D\mu_{x_j}}} + \sqrt[D]{\frac{\phi}{\alpha^2 v_D\mu_{x_j}}}\Big)\\
&\le \frac{\sqrt[D]{\alpha^3}}{3}\frac{3}{\sqrt[D]{\alpha^2}}\sqrt[D]{\frac{\phi}{v_D\mu_{x_j}}} \le \frac{\sqrt[D]{\alpha^3}}{3}\frac{3}{\sqrt[D]{\alpha^2}}\sqrt[D]{\frac{\phi}{v_D\alpha\mu_{x'_j}}}\\
&\le \frac{\sqrt[D]{\alpha^3}}{3}\frac{3}{\sqrt[D]{\alpha^3}}\sqrt[D]{\frac{\phi}{v_D\mu_{x'_j}}} = \sqrt[D]{\frac{\phi}{v_D\mu_{x'_j}}}.
\end{aligned}$$

Note that the probability of a ball with radius $\sqrt[D]{\frac{\phi}{v_D\mu_{x'_j}}}$ around x'_j is at most ϕ as claimed. Therefore, for every x'_j, we have $x'_{j+1} \in B(x'_j, r'_{x'_j})$, where radius $r'_{x'_j}$ is set such that $\mathcal{P}\big(B(x'_j, r'_{x'_j})\big) = \phi$. Now note that each point in the space belongs to at most two of the balls $x_1, x_2, \cdots, x_\ell$, and the probability of each of the balls is $\frac{\alpha^4\phi}{3^D}$, hence we have $\ell \le \frac{2\times 3^D}{\alpha^4\phi}$. By a union bound with probability at least $1 - \frac{2\delta}{n^2}$, we have sampled at least one positive example from each of the balls $x_1, x_2, \cdots, x_\ell$. This means that the $\mathcal{X}_{+,i} \cap X_+$ is connected in the ϕ-neighborhood graph of X_+, with probability at least $1 - \frac{2\delta}{n^2}$. Note that there are at most $\binom{n}{2}$ possible choices for x and y. By a union bound all of the points in $\mathcal{X}_+$ that belong to the same connected component are connected int the ϕ-neighborhood graph, with probability at least $1 - \frac{2\delta}{n^2}\binom{n}{2} \ge 1 - \delta$ as claimed. □

Now, we are ready to prove Theorem 6.

Proof of Theorem 6. Define $q = \min_{1\le i\le c}\mathcal{P}_+(\mathcal{X}_{+,i})$. Pick an arbitrary $0 < \delta < 1$. By Lemma 11, all of the positive examples are retrieved by Algorithm 2 with probability at least $1 - 2\delta$. If

$$m \ge \max\left\{\frac{2\log\left(\frac{2c}{\delta}\right)}{p\log\left(\frac{1}{1-q}\right)}, \frac{2\log\left(\frac{2}{\delta}\right)}{p^2}\right\},$$

then the excess query cost of Algorithm 2 is upper bounded by

$$\mathbb{E}[C_{\mathrm{DAA}}] \le m + C\left(\log\big(\frac{2}{\delta}\big)\log(n)n^{D-1}\right)^{\frac{1}{D}} + 2\delta n,$$

where $C = \left(\frac{(1-p)C_{\mathcal{X}_+}C_0 D}{pC_+\alpha v_D^2}\right)^{\frac{1}{D}}$. By setting $m = \tilde{\Theta}(n^{(D-1)/D})$ and $\delta = \frac{1}{n}$, we obtain

$$\mathbb{E}[C_{\mathrm{DANA}}] \in \widetilde{O}\Big(n^{(D-1)/D}\Big).$$

Note that the excepted number of negative examples in X_0 is equal to $(1-p)m$. Also, the number of negative examples in $X \setminus X_0$ is

$$(n-m)(1-p)\mathcal{P}_-\left(\mathbb{U}_{x\in\mathcal{X}_+}B(x, \sqrt[D]{\frac{\phi}{\mu_x}})\right).$$

Thus, by Assumption 4, we have

$$\begin{aligned}
&\mathbb{E}[C_{\mathrm{DAA}}] \le (1-p)m + (n-m)(1-p)\mathcal{P}_-\left(\mathbb{U}_{x\in\mathcal{X}_+}B(x, \sqrt[D]{\frac{\phi}{\mu_x}})\right)\\
&\le (1-p)m + (n-m)(1-p)C_{\mathcal{X}_+}\sqrt[D]{\frac{\phi}{v_D}}\\
&\le (1-p)m + n(1-p)C_{\mathcal{X}_+}\left(\frac{C_0 D\log\frac{2}{\delta}\log n}{pC_+\alpha v_D^2 n}\right)^{\frac{1}{D}}\\
&\le (1-p)m + n\left(\frac{(1-p)C_{\mathcal{X}_+}C_0 D}{pC_+\alpha v_D^2}\right)^{\frac{1}{D}}\left(\frac{\log\frac{2}{\delta}\log n}{n}\right)^{\frac{1}{D}}.
\end{aligned}$$

Let $C = \left(\frac{(1-p)C_{X_+}C_0 D}{pC_+\alpha v_D^2}\right)^{\frac{1}{D}}$, then we have

$$\mathbb{E}[C_{\text{DAA}}] \leq m + C\left(\log\left(\frac{2}{\delta}\right)\log(n)n^{D-1}\right)^{\frac{1}{D}}.$$

With probability 2δ Lemma 11 fails and in that case, the excess query cost is at most n. Hence we have

$$\mathbb{E}[C_{\text{DAA}}] \leq m + C\left(\log\left(\frac{2}{\delta}\right)\log(n)n^{D-1}\right)^{\frac{1}{D}} + 2\delta n,$$

as claimed. □

4 DENSITY ESTIMATION

The algorithms presented in Section 3 assume that the density function f is known. However, such an assumption is often unrealistic, and in practice this function is unknown. In this section, we use the k-nearest neighbor (k-NN) method to estimate the density function f with a function $\hat{f}$ such that for all $x \in \mathcal{X}$, we have $\frac{f(x)}{\hat{f}(x)} \in [\frac{\alpha}{2}, \frac{3}{2\alpha}]$ with probability at least $1-\delta$ for an arbitrary small $\delta \in (0,1]$. In this section, we show that incorporating k-nearest neighbor only affects the excess query cost by replacing α with $O(\alpha^2)$.

Pick an arbitrary $x \in X$. Next, we show how to estimate the value of the density function at x such that $\frac{f(x)}{\hat{f}(x)} \in [\frac{\alpha}{2}, \frac{3}{2\alpha}]$ with probability at least $1-\frac{\delta}{n}$. This together with a simple union bound gives us our desired approximate function $\hat{f}(x)$ for all $x \in X$ with probability at least $1-\delta$. Let $k = \sqrt{2n\log(\frac{\delta}{2n})}$ and let r_x be the distance form x to its k-th nearest neighbor. Note that we have $|B(x, r_x) \cap X| = k$. We define

$$\hat{f}(x) = \frac{k}{n}\frac{1}{v_D r_x^D}.$$

In the rest we upper and lower bound r_x and then make use of this to upper and lower bound $f(x)$ via $\hat{f}(x)$.

We define $r_{x,\frac{k}{2}} = \sqrt[D]{\frac{k}{2n}\frac{\alpha}{v_D f(x)}}$. We have

$$\mathcal{P}\left(B(x_j, r_{x,\frac{k}{2}})\right) = \int_{y\in B(x,r_{x,\frac{k}{2}})} f(y) \leq \int_{y\in B(x,r_{x,\frac{k}{2}})} \frac{f(y)}{\alpha}$$
$$= v_D r_{x,\frac{k}{2}}^D \frac{f(x)}{\alpha} = v_D \frac{k}{2n}\frac{\alpha}{v_D f(x)}\frac{f(x)}{\alpha} = \frac{k}{2n}.$$

Note that the definition of $B(x_j, r_{x,\frac{k}{2}})$ is independent of X. Moreover, the expected number of points from X that falls in $B(x_j, r_{x,\frac{k}{2}})$ is $\frac{k}{2}$. Hence by Hoeffding inequality, we have

$$Pr\left(|B(x_j, r_{x,\frac{k}{2}}) \cap X| \geq k\right) = Pr\left(|B(x_j, r_{x,\frac{k}{2}}) \cap X| - \frac{k}{2} \geq \frac{k}{2}\right)$$
$$\leq \exp\left(\frac{k^2}{2n}\right) = \exp\left(\frac{2n\log(\frac{\delta}{2n})}{2n}\right) = \frac{\delta}{2n}.$$

Similarly, we define $r_{x,\frac{3k}{2}} = \sqrt[D]{\frac{3k}{2n}\frac{1}{\alpha v_D f(x)}}$, and we have $\mathcal{P}\left(B(x_j, r_{x,\frac{3k}{2}})\right) \geq \frac{3k}{2n}$, and $Pr\left(|B(x_j, r_{x,\frac{3k}{2}}) \cap X| \leq k\right) \leq \frac{\delta}{2n}$. Therefore with probability $\frac{\delta}{2}$ we have $r_{x,\frac{k}{2}} \leq r_x \leq r_{x,\frac{3k}{2}}$. This means that

$$\sqrt[D]{\frac{k}{2n}\frac{\alpha}{v_D f(x)}} \leq r_x \leq \sqrt[D]{\frac{3k}{2n}\frac{1}{\alpha v_D f(x)}}.$$

The first inequality gives us $\frac{k}{2n}\frac{\alpha}{v_D r_x^D} \leq f(x)$ and the second one gives us $f(x) \leq \frac{3k}{2n}\frac{1}{\alpha v_D r_x^D}$. These imply $\frac{f(x)}{\hat{f}(x)} \in [\frac{\alpha}{2}, \frac{3}{2\alpha}]$ as claimed.

Note that, even though in our algorithms we are adjusting the distance using $f(x)$, we are only using the fact that this adjusting factor is, by Assumption 3, an estimation (i.e., α to $\frac{1}{\alpha}$ factor) of the density of any point within a small ball of x. If we use $\hat{f}(x)$ to factor the estimation is within $\frac{\alpha}{2}\times\alpha$ to $\frac{3}{2\alpha}\frac{1}{\alpha}$ factor, then this change only injects some constant factors of α to the excess query cost.

5 TERMINATION CONDITION

In this subsection, we provide a simple but asymptotically optimal termination condition for active covering algorithms when we are not aware of the exact number of positive examples. Let *ALG* be an active learning algorithm (without a termination constraint) and let C_{ALG} be an upper bound on the expected excess query cost of *ALG* if it (hypothetically) stops immediately after querying the last positive example. Note that, Theorems 5 and 6 provide upper bounds on excess query costs of our algorithms DANA and DAA, which can be used as C_{ALG} to plug into the following theorem. The following theorem provides the termination condition. We provide the proof of this theorem in appendix B

Theorem 12. *Pick an arbitrarily small probability $\delta \in (0, 0.33]$ and let ALG′ be an algorithm that runs ALG as defined above and terminates as soon as it observes $\frac{C_{ALG}}{\delta}+\log(\frac{n}{\delta C_+})$ consecutive negative labels. ALG′ queries all of the positive examples and has an excess query cost of at most $O(C_{ALG})+\tilde{O}(1)$, with probability at least $1-3\delta$.*

6 EXPERIMENTS

In this section, we compare our density-adjusted methods with the methods presented in [17]. Given the superior performance of the methods outlined in [17] compared to the other baseline approaches, it is reasonable to compare our algorithms against theirs. The offline algorithm [17] selects an initial sample and queries the other datapoints in ascending order of their distance to the positive initial samples. The active algorithm [17] does the same but it also considers positive samples retrieved after the initial queries. Our density-adjusted algorithms query the remaining datapoints in the order of their distance to the positive datapoints times their density. In the experiments we do not have access to the function f, so we approximate the density of a datapoint by calculating the inverse of its distance to its k'th nearest neighbor, where k is a hyperparameter of the algorithm. It is not hard to see that this converges to $\sqrt[D]{f(x)}$ as the number of samples grows. Due to the space limit, the modified algorithms are available in the appendix (3 and 4). Since calculating the exact k nearest neighbors is computationally expensive, we use locality-sensitive hashing for approximating the k nearest neighbors.

The effect of hyperparameter k: The density-adjusted algorithms come with a hyperparameter k. We compare different values

of $k \in \{10, 20, 50, 100\}$. The results presented in Figure 7 in the appendix show that the choice of k does not noticeably affect the performance of our algorithms.

Validating the assumptions on the datasets: We conduct some experiments to validate our assumptions on the datasets. For validating assumption 2, we will calculate the percentage of positive points around each positive point by considering its 100 nearest points. The results are presented in table 4. For validating assumption 3, we will compare the density of each point with its 100 nearest points. The results are presented in table 3.

6.1 Experiment Setup

For each dataset, we use all the available data and their original features for evaluating the results. Throughout the experiments, we set the initial sample size to be a uniformly random sample of $\frac{1}{60}$ of the datapoints. We then run the experiments using each class as the positive label and the rest of the classes as negative labels (note that this turns the problem into a binary classification). The only hyperparameter in our algorithm is k. In the course of the experiments, we set $k = 50$, and for visualization, we set the batch size to $\frac{1}{60}$ of the size of the datasets. Subsequently, we plot the percentage of positives retrieved against the number of batches. For each experiment, we run it 5 times and average out the results. The experiments were conducted on a machine equipped with an Intel(R) Xeon(R) CPU running at 2.20 GHz. Running the exact experiments on the full datasets requires huge computational power, therefore we use locality-sensitive hashing as an approximation for calculating the nearest neighbors.

Time Complexity Analysis. All parts of our algorithms are linear (treating k as a constant), except for finding the nearest neighbors where we use locality-sensitive hashing. Therefore our algorithms have a time complexity of $O(n + l)$ where l is the time complexity for finding the nearest neighbors of all the datapoints with locality-sensitive hashing, which is practically linear and sub-quadratic in theory.

Dataset	Label	Offline	Active	DANA	DAA
MNIST	0	90.47	92.98	91.74	**93.40**
	1	92.64	**92.77**	87.85	92.74
	2	76.11	86.82	91.80	**93.26**
	3	81.05	85.17	90.77	**92.90**
	4	83.84	88.68	90.67	**93.29**
CIFAR10	0	65.85	72.76	64.64	**78.45**
	1	46.09	52.02	74.53	**75.85**
	2	66.26	68.29	58.98	**70.06**
	3	50.12	53.36	62.82	**66.75**
	4	70.48	71.53	61.43	**72.10**
UCI	A	91.86	96.23	93.30	**96.38**
	B	86.84	94.99	85.00	**95.97**
	C	87.61	95.45	89.85	**96.17**
	D	86.47	94.85	85.41	**95.81**
	E	84.07	93.72	80.71	**95.01**
Fash. MNIST	0	84.79	87.06	88.75	**90.85**
	1	92.10	92.68	92.05	**93.33**
	2	82.96	84.64	86.58	**90.12**
	3	88.06	89.25	90.58	**91.93**
	4	82.81	85.38	86.37	**89.76**
Mini-Imagenet	0	57.02	64.32	72.33	**76.42**
	1	68.67	**71.05**	58.48	67.74
	2	60.83	65.53	65.60	**71.03**
	3	75.44	81.22	60.22	**81.28**
	4	74.52	84.50	81.71	**87.99**
C. C. Fraud 2013	+	46.96	71.40	93.98	**95.19**
C. C. Fraud 2023	+	47.22	85.49	94.74	**94.84**

Table 1: Area under the curve of each algorithm for 7 different datasets. Each experiment is performed 5 times and averaged across all runs. The highest values are bolded out.

6.2 Datasets

The experiments are tested on the following datasets.

UCI Letters, consisting of 20,000 datapoints with 16 numerical features and 26 classes representing each letter. The dataset has been obtained by generating letters using 20 fonts, which were randomly distorted to generate 20,000 unique images. Sixteen features have been extracted as a result of pixel count and correlations.

MNIST, consisting of 70,000 28×28 grayscale images of handwritten digits and 10 classes each representing a digit. We use the pixel intensities of the images as the features.

CIFAR10, consisting of 60,000 32×32 colorized images, and 10 classes representing an object in the image. Same as MNIST, we use the pixel intensities of the images as the features.

Fashion MNIST, consisting of 70,000 28×28 grayscale images each associated with a label from 10 classes. Same as MNIST, we use the pixel intensities of the images as the features.

Mini-Imagenet, consisting of 60,000 84×84 colorized images, and 100 classes organized using the WordNet hierarchy. Same as MNIST, we use the pixel intensities of the images as the features.

Credit Card Fraud, consisting of transactions made by credit cards by European card holders. It contains only numerical input variables which are the result of a PCA transformation.

6.3 Evaluation Metrics

We plot the percentage of positives retrieved against the number of batches to demonstrate the performance of each of the methods. We calculate the area under the curve for each of the methods as an evaluation metric (table 1). Note that as the domain of positive datapoints is not well defined and the excess cost and the number of positive datapoints add up to the total number of queried datapoints, this is a reasonable evaluation metric. The percentage of datapoints required for reaching 95% and 98% of the positive datapoints is also calculated as an alternative evaluation metric. To get more accurate results, each experiment is performed 5 times and averaged across all runs. The results for the 95% and 98% metrics are briefly discussed in the next subsection, however, due to the space limit, the detailed results for 95% are deferred to the appendix (See Table 2) and the detailed results for 98% are deferred to the full version.

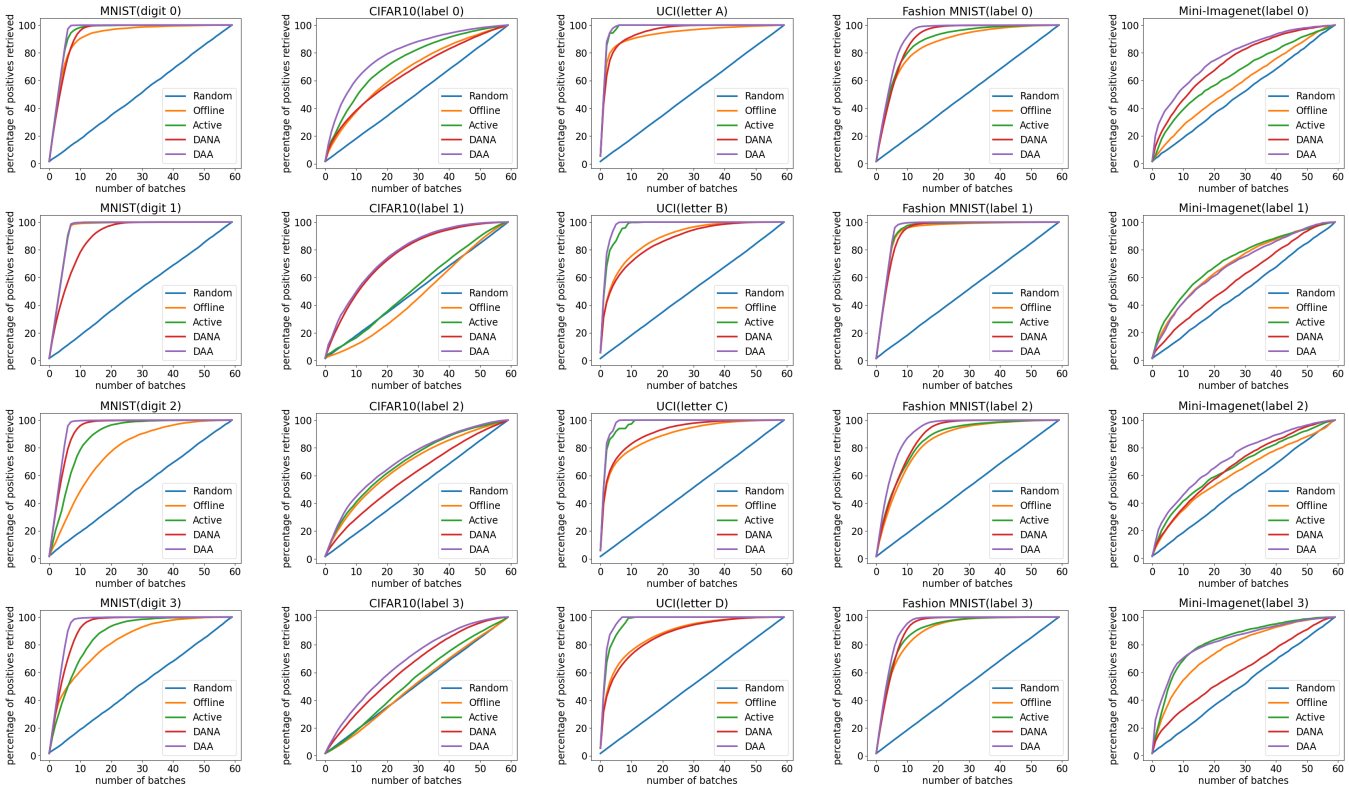

Figure 5: The percentage of positive samples retrieved after each batch on image datasets for each of the algorithms is plotted.

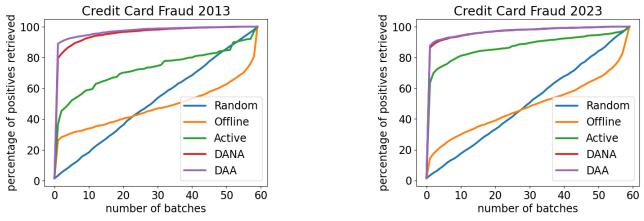

Figure 6: The percentage of positive samples retrieved after each batch on the Credit Card Fraud datasets for each of the algorithms is plotted.

6.4 Results

In each case, the result is either competitive or the density-adjusted adaptive algorithm outperforms the other algorithms. The plots are shown in Figures 5 and 6.

1. MNIST. The density-adjusted adaptive algorithm outperforms the other algorithms on 4 out of 5 tasks on all three metrics. It outperforms 3 of them by a large margin.

2. CIFAR10. The density-adjusted adaptive algorithm outperforms the other algorithms on 4 out of 5 tasks on all three metrics, outperforming 2 of them by a large margin.

3. UCI. The active density algorithm outperforms the other algorithms on all of the 5 tasks for the area under the curve metric, but the results are closely competitive.

4. Fashion MNIST. The density-adjusted adaptive algorithm outperforms the other algorithms on all of the tasks on all three metrics. It outperforms 3 of them by a large margin.

5. Mini-Imagenet. The density-adjusted adaptive algorithm outperforms the other algorithms on 3 out of 5 tasks by a large margin on all three metrics. It performs competitively on 1 of the remaining tasks and falls off on the last task.

6. Credit Card Fraud. The density-adjusted adaptive and non-adaptive algorithms outperform the other algorithms on both datasets by a large margin on all three metrics.

7 CONCLUSION

By considering the density function of the examples and adjusting the distance function, we present two algorithms DANA and DAA for active covering. Under some necessary conditions, we prove that both our algorithms discover all of the positive labels with a sublinear excess cost for constant-dimensional spaces. Moreover, our experiments on the same set of datasets used by the previous work show the superiority of our method.

We admit that our method avoids sudden changes in the density function (By Assumption 3). It might be possible to avoid this assumption by considering a distance function that takes the average of the density function over a path between the two endpoints. However, calculating this distance function may not be trivial in practice. We leave this as an open problem for future work.

REFERENCES

[1] Umang Aggarwal, Adrian Popescu, and Céline Hudelot. 2020. Active learning for imbalanced datasets. In *IEEE/CVF-WACV*. 1428–1437.

[2] John O Awoyemi, Adebayo O Adetunmbi, and Samuel A Oluwadare. 2017. Credit card fraud detection using machine learning techniques: A comparative analysis. In *ICCNI*. IEEE, 1–9.

[3] Gérard Biau, Benoît Cadre, and Bruno Pelletier. 2008. Exact rates in density support estimation. *Journal of Multivariate Analysis* 99, 10 (2008), 2185–2207.

[4] Avrim Blum. 1990. Learning boolean functions in an infinite attribute space. In *STOC*. 64–72.

[5] Avrim Blum, Lisa Hellerstein, and Nick Littlestone. 1995. Learning in the presence of finitely or infinitely many irrelevant attributes. *J. Comput. System Sci.* 50, 1 (1995), 32–40.

[6] Kamalika Chaudhuri and Sanjoy Dasgupta. 2010. Rates of convergence for the cluster tree. *NeurIPS* 23 (2010).

[7] Gui Citovsky, Giulia DeSalvo, Claudio Gentile, Lazaros Karydas, Anand Rajagopalan, Afshin Rostamizadeh, and Sanjiv Kumar. 2021. Batch active learning at scale. *NeurIPS* 34 (2021), 11933–11944.

[8] David Cohn, Les Atlas, and Richard Ladner. 1994. Improving generalization with active learning. *Machine learning* 15 (1994), 201–221.

[9] David A Cohn, Zoubin Ghahramani, and Michael I Jordan. 1996. Active learning with statistical models. *Journal of artificial intelligence research* 4 (1996), 129–145.

[10] Antonio Cuevas and Ricardo Fraiman. 1997. A plug-in approach to support estimation. *The Annals of Statistics* (1997), 2300–2312.

[11] Luc Devroye and Gary L Wise. 1980. Detection of abnormal behavior via nonparametric estimation of the support. *SIDMA* 38, 3 (1980), 480–488.

[12] Yarin Gal, Riashat Islam, and Zoubin Ghahramani. 2017. Deep bayesian active learning with image data. In *ICML*. PMLR, 1183–1192.

[13] Roman Garnett, Yamuna Krishnamurthy, Xuehan Xiong, Jeff Schneider, and Richard Mann. 2012. Bayesian optimal active search and surveying. *arXiv preprint arXiv:1206.6406* (2012).

[14] Dilek Hakkani-Tür, Giuseppe Riccardi, and Allen Gorin. 2002. Active learning for automatic speech recognition. In *ICASSP*, Vol. 4. IEEE, IV–3904.

[15] David P Helmbold, Nicholas Littlestone, and Philip M Long. 2000. Apple tasting. *Information and Computation* 161, 2 (2000), 85–139.

[16] Lalit Jain and Kevin G Jamieson. 2019. A new perspective on pool-based active classification and false-discovery control. *NeurIPS* 32 (2019).

[17] Heinrich Jiang and Afshin Rostamizadeh. 2021. Active Covering. In *ICML*. PMLR, 5013–5022.

[18] Shali Jiang, Roman Garnett, and Benjamin Moseley. 2019. Cost effective active search. *NeurIPS* 32 (2019).

[19] Shali Jiang, Gustavo Malkomes, Matthew Abbott, Benjamin Moseley, and Roman Garnett. 2018. Efficient nonmyopic batch active search. *NeurIPS* 31 (2018).

[20] Ajay J Joshi, Fatih Porikli, and Nikolaos Papanikolopoulos. 2009. Multi-class active learning for image classification. In *CVPR*. IEEE, 2372–2379.

[21] Seyed Mehran Kazemi, Anton Tsitsulin, Hossein Esfandiari, MohammadHossein Bateni, Deepak Ramachandran, Bryan Perozzi, and Vahab Mirrokni. 2022. Tackling Provably Hard Representative Selection via Graph Neural Networks. *arXiv preprint arXiv:2205.10403* (2022).

[22] Amir E Khandani, Adlar J Kim, and Andrew W Lo. 2010. Consumer credit-risk models via machine-learning algorithms. *Journal of Banking & Finance* 34, 11 (2010), 2767–2787.

[23] Alexander P Korostelev and Aleksandr Borisovich Tsybakov. 1993. Estimation of the density support and its functionals. *Problemy Peredachi Informatsii* 29, 1 (1993), 3–18.

[24] Kang Li, Zhenyu Zhong, and Lakshmish Ramaswamy. 2008. Privacy-aware collaborative spam filtering. *IEEE Transactions on Parallel and Distributed systems* 20, 5 (2008), 725–739.

[25] Wolfgang Maass. 1991. *On-line learning with an oblivious environment and the power of randomization.* International Computer Science Institute.

[26] Venkata Vamsikrishna Meduri, Lucian Popa, Prithviraj Sen, and Mohamed Sarwat. 2020. A comprehensive benchmark framework for active learning methods in entity matching. In *SIGMOD*. 1133–1147.

[27] Michael Mitzenmacher and Eli Upfal. 2017. *Probability and computing: Randomization and probabilistic techniques in algorithms and data analysis.* Cambridge university press.

[28] Chikashi Nobata, Joel Tetreault, Achint Thomas, Yashar Mehdad, and Yi Chang. 2016. Abusive language detection in online user content. In *WWW*. 145–153.

[29] Si-sheng Ou-Yang, Jun-yan Lu, Xiang-qian Kong, Zhong-jie Liang, Cheng Luo, and Hualiang Jiang. 2012. Computational drug discovery. *Acta Pharmacologica Sinica* 33, 9 (2012), 1131–1140.

[30] KC Santosh. 2020. AI-driven tools for coronavirus outbreak: need of active learning and cross-population train/test models on multitudinal/multimodal data. *Journal of medical systems* 44 (2020), 1–5.

[31] Christopher Schröder and Andreas Niekler. 2020. A survey of active learning for text classification using deep neural networks. *arXiv preprint arXiv:2008.07267* (2020).

[32] Aarti Singh, Clayton Scott, and Robert Nowak. 2009. Adaptive hausdorff estimation of density level sets. *The Annals of Statistics* 37, 5B (2009), 2760–2782.

[33] Petr Slavík. 1996. A tight analysis of the greedy algorithm for set cover. In *STOC*. 435–441.

[34] Hui Ru Tan, Wei Heng Chng, Christian Chonardo, Magdeline Tao Tao Ng, and Fun Man Fung. 2020. How chemists achieve active learning online during the COVID-19 pandemic: using the Community of Inquiry (CoI) framework to support remote teaching. *Journal of Chemical Education* 97, 9 (2020), 2512–2518.

[35] Puning Zhao and Lifeng Lai. 2022. Analysis of knn density estimation. *IEEE Transactions on Information Theory* 68, 12 (2022), 7971–7995.

A ADDITIONAL EXPERIMENT DETAILS

Algorithm 3 Density-adjusted Non-Adaptive Algorithm (DANA) without access to f

Input Dataset X, initial sample size m and density parameter k.

1: Let r_x be the distance of x to its k'th nearest neighbor in X.
2: Let X_0 be m examples sampled uniformly without replacement from X.
3: Label query X_0 and let $X_{+,0}$ be the positive examples.
4: Label query remaining examples in ascending order of $\frac{d(x,X_{+,0})}{r_x}$ until all positive examples are labeled.

Algorithm 4 Density-adjusted Adaptive Algorithm (DAA) without access to f

Input Dataset X, initial sample size m and density parameter k.

1: Let r_x be the distance of x to its k'th nearest neighbor in X.
2: Let X_0 be m examples sampled uniformly without replacement from X.
3: Label query X_0 and let $X_{+,0}$ be the positive examples.
4: Initialize $X_p \leftarrow X_{+,0}$ and $X_a \leftarrow X_0$
5: **while** not all positive examples in X are labeled **do**
6: Label query $x = \arg\min_{x \in X \setminus X_a} \frac{d(x,X_p)}{r_x}$
7: **if** x has a positive label **then**
8: $X_p \leftarrow X_p \cup \{x\}$
9: **end if**
10: $X_a \leftarrow X_a \cup \{x\}$
11: **end while**

Dataset	100%	99.9%	99.5%	99%	98%	95%
MINST	0.16	0.28	0.38	0.47	0.57	0.67
CIFAR10	0.30	0.43	0.48	0.50	0.53	0.58
UCI	0.19	0.41	0.53	0.57	0.62	0.69
Fashion MNIST	0.24	0.39	0.47	0.51	0.55	0.63
Mini-Imagenet	0.27	0.42	0.47	0.50	0.53	0.58

Table 3: Verifying Assumption 3: The density ratio between each point and its 100 nearest neighbors is computed. For each threshold $t \in$ 100%, 99.9%, 99.5%, 99%, 98%, 95%, the displayed values represent the minimum ratio observed among at least t percent of the computed ratios.

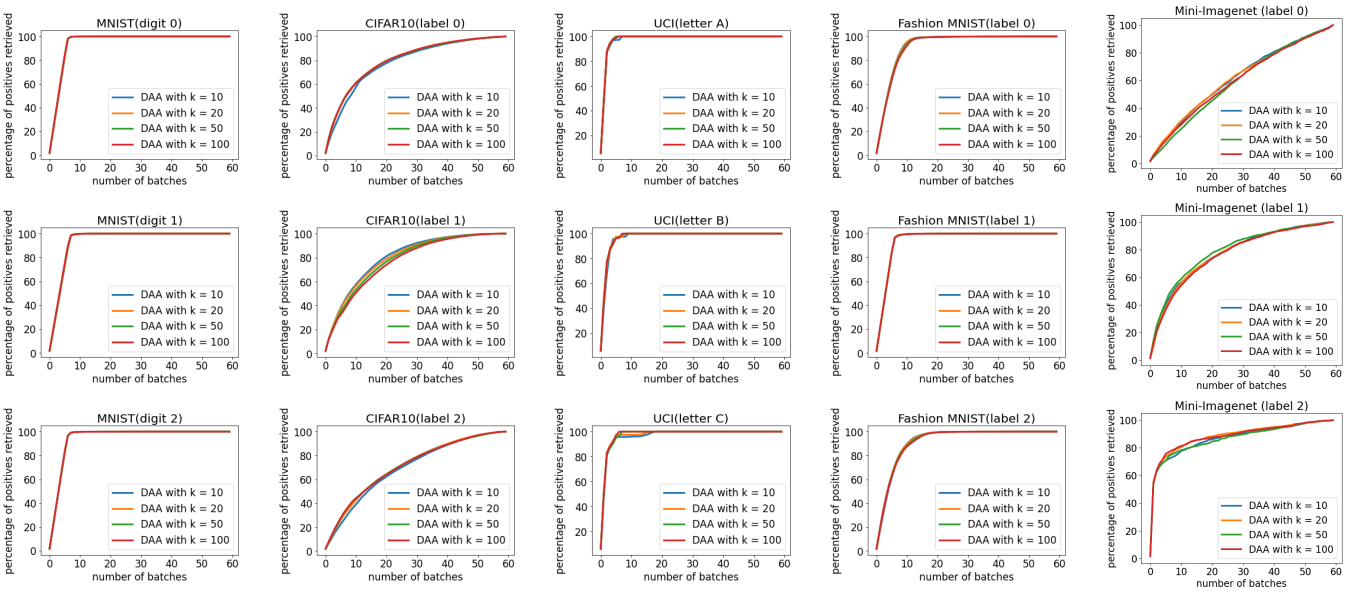

Figure 7: The percentage of positive samples retrieved after each batch, using the DAA algorithm for the first three labels with different values of k is plotted.

Dataset	Label	Offline	Active	DANA	DAA
MNIST	0	27.62	13.79	17.57	**11.30**
	1	12.33	**12.20**	30.03	12.49
	2	63.51	32.05	17.42	**11.50**
	3	56.46	38.13	20.39	**12.88**
	4	44.50	27.55	22.35	**11.74**
CIFAR10	0	87.38	79.33	88.56	**71.03**
	1	93.08	91.18	69.29	**65.87**
	2	87.10	83.07	90.80	**81.19**
	3	95.07	93.09	83.19	**79.44**
	4	81.80	**78.36**	86.20	79.22
UCI	A	38.82	8.25	23.99	**6.55**
	B	46.12	13.08	51.46	**8.90**
	C	48.49	16.13	38.01	**8.54**
	D	51.06	14.22	51.29	**9.35**
	E	54.52	20.21	63.48	**17.32**
Fashion	0	53.03	40.56	26.18	**20.20**
	1	16.75	15.01	17.18	**11.34**
	2	48.09	43.44	31.54	**25.34**
	3	35.47	32.30	20.80	**18.11**
	4	51.07	41.78	32.71	**25.20**
Mini-Imagenet	0	90.17	88.72	73.74	**72.30**
	1	83.48	84.86	86.65	**82.22**
	2	93.94	89.23	82.76	**78.78**
	3	73.53	**67.45**	89.31	72.73
	4	73.28	61.91	86.10	**59.74**
C. C. Fraud 2013	+	99.56	87.84	26.05	**20.63**
C. C. Fraud 2023	+	98.92	86.39	**21.69**	22.26

Table 2: The percentage of datapoints required, to obtain 95% of the positives labels.

Dataset	#	100%	99.9%	99.5%	99%	98%	95%
MNIST	0	0.01	0.02	0.09	0.14	0.30	0.68
	1	0.01	0.01	0.06	0.16	0.55	0.99
	2	0.01	0.01	0.02	0.03	0.06	0.15
	3	0.01	0.01	0.03	0.05	0.09	0.21
	4	0.01	0.01	0.02	0.05	0.13	0.28
CIFAR10	0	0.01	0.01	0.02	0.02	0.03	0.05
	1	0.01	0.01	0.01	0.01	0.01	0.01
	2	0.01	0.02	0.03	0.04	0.06	0.08
	3	0.01	0.01	0.01	0.02	0.02	0.03
	4	0.01	0.01	0.02	0.03	0.04	0.06
UCI	A	0.10	0.10	0.11	0.12	0.14	0.18
	B	0.10	0.10	0.12	0.14	0.16	0.18
	C	0.03	0.03	0.07	0.08	0.11	0.17
	D	0.07	0.07	0.10	0.13	0.15	0.19
	E	0.06	0.06	0.11	0.13	0.15	0.20
Fash. MNIST	0	0.01	0.01	0.02	0.03	0.05	0.14
	1	0.01	0.01	0.03	0.04	0.08	0.21
	2	0.01	0.01	0.02	0.03	0.07	0.14
	3	0.01	0.01	0.02	0.03	0.06	0.16
	4	0.01	0.01	0.03	0.05	0.09	0.15
Mini-Imagenet	0	0.01	0.01	0.01	0.01	0.01	0.01
	1	0.01	0.01	0.01	0.01	0.01	0.01
	2	0.01	0.01	0.01	0.01	0.01	0.01
	3	0.01	0.01	0.01	0.01	0.01	0.01
	4	0.01	0.01	0.01	0.01	0.01	0.01

Table 4: Verifying Assumption 2: The percentage of positive points within the 100 nearest neighbors of each positive point is computed. For each threshold $t \in \{100\%, 99.9\%, 99.5\%, 99\%, 98\%, 95\%\}$, the depicted values represent the proportion of positive points found within at least t percent of the positive points.

B PROOF OF THEOREM 12

Proof. First, using a simple Markov inequality we show that ALG' has an excess query cost of at most $O(C_{ALG}) + \tilde{O}(1)$ with probability at least $1-\delta$, then we show that ALG' queries all of the positive examples with probability at least $1-2\delta$.

Note that, when ALG queries the last positive example, its the expected excess query cost is C_{ALG}. Hence by Markov inequality, when ALG queries all of the positive examples its excess query cost is at most $\frac{C_{ALG}}{\delta}$ with probability at least $1-\delta$. After that, ALG only queries negative examples. Hence ALG' stops after querying at most $\frac{C_{ALG}}{\delta} + \log(\frac{n}{\delta})$ extra queries. Hence the excess query cost of ALG' is at most $\frac{C_{ALG}}{\delta} + \frac{C_{ALG}}{\delta} + \log(\frac{n}{\delta C_+}) \in O(C_{ALG}) + \tilde{O}(1)$ as claimed.

Next, we show that ALG' queries all of the positive examples with probability at least $1-2\delta$. In order to show this we show that the probability of observing $\frac{C_{ALG}}{\delta} + \log(\frac{n}{\delta C_+})$ consecutive negative labels before the last positive example is at most 2δ. As we mentioned above the probability that we query more than $\frac{C_{ALG}}{\delta}$ examples out of the domain of the positive examples is at most δ. Hence, with probability at least $1-\delta$ $\log(\frac{n}{\delta C_+})$ out of $\frac{C_{ALG}}{\delta} + \log(\frac{n}{\delta C_+})$ consecutive negative labels are queried from the domain of positive examples. In the rest, we look at the queries that have been made from the domain of positive examples and bound the probability that $\log(\frac{n}{\delta C_+})$ consecutive queries from the domain of positive examples are negative, then we apply a union bound over all such subsequences to calculate the probability of failure.

Note that each query from the domain of positive examples is positive with probability at least C_+. Hence the probability that $\log(\frac{n}{\delta C_+})$ consecutive queries from the domain of positive examples are negative is

$$(1-C_+)^{\log(\frac{n}{\delta C_+})} = e^{-\log(\frac{n}{\delta})} = \frac{\delta}{n}.$$

There are at most n such subsequences, and hence the probability that we observe one such subsequence is at most δ. □

C EMPIRICAL ESTIMATION OF λ_0 AND λ_1 [17]

In this section, we conduct some experiments to estimate the parameters λ_0 and λ_1 as defined by Jiang and Rostamizadeh [17] across different datasets. The parameter λ_0 represents the lower bound of $f_+(x)$ and λ_1 represents the upper bound of $f_-(x)$. For a given k, we empirically estimate $f(x)$ by dividing k by the volume of the smallest ball containing the k nearest neighbors of x and multiplying this result by the volume of the smallest enclosing ball divided by the total number of points in the dataset. Note that the computed values are scaled by the volume of the smallest enclosing ball. The results indicate that the parameters and their difference are significantly large, showing that even scaling alone will not be effective.

Dataset	Label	$\log_{10}\lambda_0$	$\log_{10}\lambda_1$
MNIST	0	127	680
	1	90	475
	2	105	693
	3	110	693
	4	92	690
CIFAR	0	996	2828
	1	1025	2768
	2	989	2672
	3	886	2851
	4	948	2712
UCI	0	3	14
	1	5	15
	2	4	15
	3	5	15
	4	4	14
Fashion	0	126	746
	1	80	713
	2	182	744
	3	195	746
	4	152	742
Mini-Imagenet	0	394	2341
	1	411	2326
	2	445	2418
	3	765	2375
	4	638	2394

Table 5: Approximate values of $\log_{10}(\lambda_0)$ and $\log_{10}(\lambda_1)$ across different datasets for $k = 100$.