Active Covering

Heinrich Jiang [1] **Afshin Rostamizadeh** [1]

Abstract

We analyze the problem of active covering, where the learner is given an unlabeled dataset and can sequentially label query examples. The objective is to label query all of the positive examples in the fewest number of total label queries. We show under standard non-parametric assumptions that a classical support estimator can be repurposed as an offline algorithm attaining an excess query cost of $\widetilde{\Theta}(n^{D/(D+1)})$ compared to the optimal learner, where n is the number of datapoints and D is the dimension. We then provide a simple active learning method that attains an improved excess query cost of $\widetilde{O}(n^{(D-1)/D})$. Furthermore, the proposed algorithms only require access to the positive labeled examples, which in certain settings provides additional computational and privacy benefits. Finally, we show that the active learning method consistently outperforms offline methods as well as a variety of baselines on a wide range of benchmark image-based datasets.

1. Introduction

Active learning is an increasingly important practical area of machine learning as the amount of data grows faster than the resources to label these datapoints, which can be costly. In this paper, we introduce a variant of the active learning problem, called active covering, where the goal is to label query all of the positive examples given an unlabeled dataset in as few total queries as possible.

Active covering arises in many machine learning problems. In credit card fraud detection, one goal is to find the instances of fraud with as few queries as possible to the user asking if a transaction was fraudulent (Awoyemi et al., 2017). In mineral exploration, the goal is to find all of the valuable resources with as little exploration as necessary (Acosta et al., 2019). In computational drug discovery, one goal is to discover all of the effective drugs with as few suggested candidates as possible, as running trials on each candidate can be costly (Ou-Yang et al., 2012). In bank loan applications, a label query means granting the loan to an applicant (as the only way to know if the applicant would default or not)– as such, negative label queries can be costly (Khandani et al., 2010). Finally, many internet applications need to moderate abusive content, and the goal is to quickly identify and remove all of the abusive content with as few label queries as possible to human operators (Nobata et al., 2016).

For our analysis, we assume that an unlabeled pool of n datapoints is generated by drawing i.i.d. samples from a mixture of two distributions, where one of the distributions represents the positive examples and the other the negative examples. Our goal is to retrieve all of the positive examples from this pool with as few label queries as possible. Furthermore, we assume that the support of the positive examples is compact and its density function is lower bounded by a positive quantity. Finally, we leverage a few additional standard and mild nonparametric regularity assumptions on the the curvature of the support of positive examples. These assumptions allow for a wide range of distributions (e.g. mixtures of truncated multivariate Gaussians), and in particular the support of positives and negatives can be of any shape and can arbitrarily overlap.

We first establish the optimal active covering algorithm, which has knowledge of the support of positive examples. This algorithm, thus, will query all of the examples that lie in the support of positive examples (which will contain some negative examples where there is an overlap with support of negative examples). Then, the performance of any active covering algorithm can be compared to the optimal learner via the expected *excess query cost* that the algorithm incurs in addition to what the optimal learner is expected to incur.

To begin with, we analyze an offline algorithm that is based on a classical method in support estimation, providing both upper and lower bounds on the excess query cost. We then provide an active learning procedure based on the explore-then-commit strategy, where we first explore with an initial random sample, and then exploit by querying the example closest to any of the positive examples found thus far. We show that the active learning procedure achieves a provably better excess query cost than the offline algorithm.

The presented methods also have the beneficial property of

[1]Google Research. Correspondence to: Heinrich Jiang <heinrichj@google.com>.

Proceedings of the 38^{th} International Conference on Machine Learning, PMLR 139, 2021.

using only the queried positive examples for learning. This not only has desirable computational implications in that we don't need to store the negative examples (especially in highly label imbalanced datasets with few positives), but also is practical in situations where there are strict privacy requirements for the negative examples. For example, this arises in the problem of fake account detection faced by social network platforms (García-Recuero, 2016), where real account information is highly sensitive data and may even not be allowed to use for training. Another such application is spam email detection, where it may be desirable (or necessary) to avoid training with non-spam emails (Li et al., 2008). While this privacy aspect is not a focus of the paper, it may be a feature of independent interest.

We now summarize our contributions as follows.

- In Section 2, we introduce and formalize the active covering problem and establish the notion of excess query cost.
- In Section 3, we analyze the offline learner and show matching upper and lower bounds on the excess query cost of $\widetilde{\Theta}(n^{D/(D+1)})$.
- In Section 4, we introduce and analyze the Explore-then-Commit active algorithm and show it has excess query cost $\widetilde{O}(n^{(D-1)/D})$.
- In Section 6, we show empirical results on a wide range of benchmark image-based datasets (Letters, MNIST, Fashion MNIST, CIFAR10, CelebA) comparing the Explore-then-Commit algorithm to a number of offline and active baselines.

2. Active Covering

In this section, we formulate the active covering problem. We are given an unlabeled dataset of n datapoints, X, and the goal is to minimize the number of label queries necessary until all positive examples are labeled. We provide the assumptions on the underlying distribution from which X is drawn from and establish the notion of excess query cost, which is the additional label queries compared to the optimal procedure, which will be defined later.

2.1. Theoretical Setup

We make the following assumption on the data generating process, which says that with probability p, we draw a positive example from distribution $\mathcal{P}_+$ and with probability $1-p$, we draw a negative example from distribution $\mathcal{P}_-$.

Assumption 1. *The dataset X is drawn i.i.d. from a distribution $\mathcal{P} := p \cdot \mathcal{P}_+ + (1-p) \cdot \mathcal{P}_-$, for some $p \in (0,1)$ and $\mathcal{P}_+$ and $\mathcal{P}_-$ are distributions over $\mathbb{R}^D$ of the positive and negative examples respectively.*

We then require the following regularity assumption on the distribution of positive examples. The first part ensures that the density of positive examples is lower bounded in its support; otherwise, some positive examples will only appear as outliers, making it impossible for any learner to find them in a non-trivial manner. The second part ensures that the support of positive examples does not become arbitrarily thin anywhere, otherwise there may not be any samples drawn from these regions and it will be very difficult to recover the entire support from a finite sample. This is a standard assumption in a variety of density estimation scenarios e.g. (Cuevas et al., 1997; Singh et al., 2009).

Assumption 2. *There exists density function $f_+ : \mathbb{R}^D \to \mathbb{R}$ corresponding to $\mathcal{P}_+$ with a compact support $\mathcal{X}_+$ that can be decomposed into a finite number of connected components. There exists $\lambda_0, r_0, C_+ > 0$ such that the following holds:*

- *$f_+(x) \geq \lambda_0$ for all $x \in \mathcal{X}_+$.*
- *For all $0 < r < r_0$ and $x \in \mathcal{X}_+$, we have $Vol(B(x,r) \cap X_+) \geq C_+ \cdot Vol(B(x,r))$, where $B(x,r) := \{x' \in \mathbb{R}^D : |x - x'| \leq r\}$.*

The final assumption ensures that the negative example's density is upper bounded to ensure that there are no possible arbitrarily dense clumps of negative examples near the boundary of $\mathcal{X}_+$. Otherwise, querying in such regions may yield too few positive examples and it may be difficult to learn that such a region is part of the support of positive examples.

Assumption 3. *There exists density function $f_- : \mathbb{R}^D \to \mathbb{R}$ corresponding to $\mathcal{P}_-$ and $\lambda_1 > 0$ such that $f_-(x) \leq \lambda_1$ for all $x \in \mathcal{X}$.*

Our assumptions are quite mild as they allow for a wide range of distributions. In particular, they are non-parametric so there are no assumptions that the data is generated based on a parameterized model. Moreover, the support of the positive examples (as well as the support of the negative examples) can be of arbitrary bounded shape, need not be a single connected component (i.e. can appear as a number of connected components), and can intersect arbitrarily with the support of the opposite label. Such mild assumptions can model the wide range of data distributions arising in practice.

Perhaps the strongest of our assumptions is that the density on $\mathcal{X}_+$ is lower bounded. We stress that if the density of the positive examples on $\mathcal{X}_+$ can become arbitrarily small, then some regions can become so low-density that the positive examples in those regions can be considered outliers. For such outliers, it may be unrealistic to expect a procedure to efficiently find them. Nonetheless, in practice, our methodology can still be applied on datasets containing positive

outliers, but without guarantees that those outliers will be efficiently recovered. We made this assumption to keep our theoretical analysis from becoming too involved; however it's worth noting that it is possible to relax this assumption and allow the density to have smooth boundaries. To do so, we would assume parameters that bound the rate of decay of the density as it goes to 0 and provide final guarantees that depend on the decay parameters. Assumptions used in recent analysis on densities with smooth boundaries (Zhao & Lai, 2020) can be adapted here, which is a future research direction.

2.2. Optimal Learner and Excess Query Cost

We will analyze a number of new learners for this problem. To do so, we establish a metric called *excess query cost*, which compares the learner to that of the the *optimal* learner, which takes the optimal strategy given knowledge of $\mathcal{X}_+$. This learner is unattainable in practice, but serves as a theoretical limit in which to quantify the excess query cost of an algorithm with respect to, to be defined below.

The optimal learner has knowledge of $\mathcal{X}_+$ and therefore its strategy is to label query every point in $\mathcal{X}_+$. Let Q_{opt} be the number of label queries incurred by the optimal learner. Thus, the optimal learner attains an expected number of queries of:

$$\mathbb{E}[Q_{\text{opt}}] = n \cdot \mathcal{P}(\mathcal{X}_+).$$

We can then for any algorithm define the notion of excess query cost, which is the additional label queries needed compared to the optimal learner. That is, if the cost of an algorithm is C, then the excess query cost is

Definition 1 (Excess Query Cost)**.** *Suppose that an algorithm needs to make Q label queries before labeling all of the positive examples. Then the excess query cost of the procedure is defined as: $C := Q - Q_{opt}$.*

For the results in the paper, we analyze the *expected* excess query cost, where the expectation is taken over the distribution from which sample X is drawn from.

2.3. Passive Learner

We first provide a result for the passive learning algorithm, which queries labels uniformly at random until all positive examples have been retrieved and serves as our most basic baseline. In the following theorem we show a lower bound on the excess query cost that is linear in the pool size.

Theorem 1 (Lower bound for Passive Learner)**.** *There exists a distribution satisfying Assumptions 1, 2, and 3 such that with probability at least $\frac{1}{2}$, we have (letting $C_{passive}$ be the excess query cost of the passive learner): $\mathbb{E}[C_{passive}] \geq \frac{1}{2} \cdot n$.*

Algorithm	Excess Query Cost
Passive	$\Theta(n)$
Offline	$\widetilde{\Theta}(n^{D/(D+1)})$
Active	$\widetilde{O}(n^{(D-1)/D})$

Table 1. **Summary of algorithms and results**.

Algorithm 1 Offline Learner

Inputs: Dataset X, initial sample size m.
Let X_0 be m examples sampled uniformly without replacement from X.
Label query X_0 and let $X_{0,+}$ be the positive examples.
Label query remaining examples in ascending order of minimum distance to $X_{0,+}$ (i.e. $x \to \min_{x' \in X_{0,+}} |x - x'|$) until all positive examples are label queried.

3. Offline Learner

The offline learner (Algorithm 1) will first randomly sample m points to label query and then label queries the remaining examples in ascending order of minimum distance to any of the initially sampled positive examples until all positives are labeled. 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.

The offline learner is based on a classical technique in the support estimation literature (Cuevas et al., 1997), where the support of a distribution can be covered with a finite sample of m points drawn from this distribution by taking the ϵ-neighborhood of the sample for appropriate ϵ. We apply this methodology to only the positive examples in our initial sample of m points. We show that the Algorithm 1 finishes when it label queries everything within $\epsilon \approx (\log(m)/m)^{1/D}$ of the initial positive examples, and thus it will cover all the examples in $\mathcal{X}_+$ and the excess cost will be proportional to $\epsilon \cdot n$ (all details in the Appendix).

The formal guarantee for the excess query cost is a follows:

Theorem 2 (Excess Query Cost for Offline Learner)**.** *Suppose that Assumptions 1, 2, and 3 hold. Let $0 < \delta < 1$. There exists $C, M_0 > 0$ depending on $\mathcal{P}$ such that the following holds. In Algorithm 1, suppose that m is chosen sufficiently large such that $m \geq \frac{2\log(2/\delta)}{p^2}$ and $\frac{m}{\log m} \geq \log(4/\delta) \cdot M_0$. Then, with probability at least $1 - \delta$, Algorithm 1 has an expected excess query cost of:*

$$\mathbb{E}[C_{offline}] \leq (1-p) \cdot m + C \cdot \left(\frac{\log(4/\delta) \cdot \log(p \cdot m/2)}{m} \right)^{1/D} \cdot n.$$

We now have the following immediate result by optimizing

Algorithm 2 Active Explore-then-Commit Learner

Inputs: Dataset X, initial sample size m.
Let X_0 be m examples sampled uniformly without replacement from X.
Label query X_0 and let $X_{+,0}$ be the positive examples.
Initialize $X_p \leftarrow X_{+,0}$ and $X_a \leftarrow X_0$
while not all positives examples in X are labeled **do**
  Label query $x = \arg\min_{x \in X \setminus X_a} d(x, X_p)$
  if x has a positive label **then**
    $X_p \leftarrow X_p \cup \{x\}$.
  end if
  $X_a \leftarrow X_a \cup \{x\}$.
end while

m as a function of n, trading off the cost from the initial exploration (first term) and the cost from the exploitation (second term):

Corollary 1. *Under the conditions of Theorem 2, setting $m \approx n^{D/(D+1)}$ results in expected excess query cost bounded as follows:*

$$\mathbb{E}[C_{\textit{offline}}] \leq \textit{PolyLog}(n, 1/\delta) \cdot n^{D/(D+1)}.$$

We also provide the following lower bound, which shows that the offline learner cannot achieve a better rate (proof found in the appendix).

Theorem 3 (Lower Bound for Offline Learner)**.** *There exists a distribution satisfying Assumptions 1, 2, and 3 such that with probability at least $\frac{1}{4}$, we have for n sufficiently large and some constant $C > 0$:*

$$\mathbb{E}[C_{\textit{offline}}] \geq (1-p) \cdot m + C \cdot \left(\frac{\log m}{m}\right)^{1/D} \cdot n.$$

4. Active Explore-then-Commit Learner

We next show an active approach (Algorithm 2) inspired by Explore-then-Commit strategy (Garivier et al., 2016) that proceeds by first exploring by randomly sampling a set of examples, and then commits to a greedy approach of choosing the closest unlabeled example to any positive example labeled thus far until all of the positive examples are labeled.

To analyze the algorithm there are three key steps: first we show that in the explore phase, we choose at least one example from each connected component (CC) of $\mathcal{X}_+$ (Lemma 1). Next, we show that in any CC of $\mathcal{X}_+$, all of the positive examples are in the same connected component in the ϵ-neighborhood graph for some ϵ specified later (Lemma 2). The final step is combining these two results to show a bound on the excess query cost.

We now give the following result which says that for m sufficiently large, depending on the probability mass distribution of the CCs of $\mathcal{X}_+$, we will with high probability have a positive example from each of the CCs in the initial sample.

Lemma 1. *Suppose Assumptions 1 and 2 hold and let $0 < \delta < 1$. Let the connected components of $\mathcal{X}_+$ be $\mathcal{X}_{+,1}, ..., \mathcal{X}_{+,c}$. Let $q := \min_{i \in [c]} \mathcal{P}_+(\mathcal{X}_{+,i})$. If*

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

then with probability at least $1-\delta$, the initial m examples will contain a positive example in each of $\mathcal{X}_{+,i}$ for $i \in [c]$.

The next result shows that the positive examples in each CC of $\mathcal{X}_+$ appear in the same CC of the ϵ-neighborhood graph of the positive example for appropriate choice of ϵ. This will be important in showing that after greedily sampling enough examples after the explore phase, we will query all of the examples in $\mathcal{X}_+$ but not query examples that are more than ϵ away from $\mathcal{X}_+$.

Lemma 2 (Connectedness)**.** *Suppose Assumptions 1 and 2 hold. Let $0 < \delta < 1$ and $\mathcal{X}_{+,1}, ..., \mathcal{X}_{+,c}$ be the connected components of $\mathcal{X}_+$. The following holds with probability at least $1-\delta$. For each $i \in [c]$, we have that $\mathcal{X}_{+,i} \cap X_+$ is connected in the ϵ-neighborhood graph of X_+, where*

$$\epsilon = 3\left(\frac{C_0 \cdot D\log(2/\delta) \cdot \log n}{p \cdot C_+ \cdot \lambda_0 \cdot v_D \cdot n}\right)^{1/D},$$

and n is sufficiently large so that $\epsilon \leq r_0$.

We now combine the two results to obtain an excess query cost guarantee for the active learner. Lemma 1 ensures that our initial sample of m examples contains an example from each CC of $\mathcal{X}_+$ and Lemma 2 ensures that when we actively sample in a greedy manner, we eventually query all of the positive examples and never query any example that is too far from $\mathcal{X}_+$– this farness determines how much the active algorithm samples outside of $\mathcal{X}_+$ and hence determines the expected excess query cost.

Theorem 4 (Excess query cost for Algorithm 2)**.** *Suppose Assumptions 1, 2, and 3 hold and let $0 < \delta < 1$. Let the connected components of $\mathcal{X}_+$ be $\mathcal{X}_{+,1}, ..., \mathcal{X}_{+,c}$ and $q := \min_{i \in [c]} \mathcal{P}_+(\mathcal{X}_{+,i})$. There exists constants $C, N_0 > 0$ depending on $\mathcal{P}$ such that the following holds. If*

$$m \geq \max\left\{\frac{2\log(4c/\delta)}{p \cdot \log(1/(1-q))}, \frac{2\log(4/\delta)}{p^2}\right\},$$

and $\frac{n}{\log n} \geq N_0 \log(4/\delta)$, then with probability at least $1-\delta$, we have the following excess query cost guarantee for Algorithm 2:

$$\mathbb{E}[C_{\textit{exp-commit}}] \leq m + C \cdot ((\log(4/\delta) \cdot \log n)^{1/D} \cdot n^{(D-1)/D}.$$

Remark 1. *Our requirement for m is tight w.r.t. q and c in the case where $q = \frac{1}{c}$ (i.e. equal probability of each CC). In this case, it reduces down to the classic coupon collector problem (Boneh & Hofri, 1997): each CC is a coupon and the expected number of times we draw a coupon with replacement until we receive one example from each is $\Omega(c \log c) = \Omega(\log c / \log(1/(1-q)))$ by Taylor expansion of $\log(1/(1-q))$.*

Remark 2. *While our results all have a strong dependence on the dimension (commonly referred to as the curse of dimensionality), it's been shown that non-parametric techniques such as these algorithms can automatically adapt to the intrinsic dimension of the data and the convergence rates can be shown to depend only on this dimension and not the ambient dimension (i.e. arguments from Pelletier (2005); Kpotufe (2011); Jiang (2017; 2019) can be adapted here).*

5. Related Works

The problem of actively retrieving the positive labeled examples was studied as *active search* by Garnett et al. (2012), where the goal is to label query as many positive examples given a fixed budget and they propose a sequential Bayesian approach that optimizes the expected number of positive labels across timesteps; however their method is computationally expensive requiring $O((2 \cdot n)^{\ell})$ runtime where ℓ is the number of lookahead timesteps to optimize over. Efficient approximations to this active search technique have been proposed (Jiang et al., 2018; 2019). In our theoretical setting, the goal is to label query all of the positive examples with as few label queries as possible rather than having a fixed known labeling budget.

A recent work by Jain & Jamieson (2019) designs an algorithm to actively identify the largest number of positive examples while minimizing or constraining the false-discovery rate (i.e. false-negative rate). They propose a bandit-style active elimination procedure which strategically label queries datapoints (i.e. each datapoint can be seen as an arm and label querying can be seen as pulling the arm) to find the best classification. In our work, we leverage the structure of the data while Jain & Jamieson (2019) considers each datapoint as its own bandit arm and doesn't explicitly use information about the location of these datapoints. The contributions of (Jain & Jamieson, 2019) are primarily in the theoretical analysis of the setting and proposed algorithm, while the practicality of the algorithm itself is limited due to its computation complexity.

A related line of work is learning under one-sided feedback, first studied under the name *apple tasting* by Helmbold et al. (2000), where the learner receives the true labels for only examples it predicted positively on and the goal is to have as high accuracy as possible. Recently, Jiang et al. (2020) studied the one-sided feedback problem for generalized linear models and attain regret guarantees using an adaptive UCB-based approach under their proposed one-sided loss. This problem is similar to our proposed active covering problem in that both cases we desire to label query the positive examples; however, a key difference is that both Helmbold et al. (2000) and Jiang et al. (2020) operate in the *streaming* setting, where predictions must be made in real-time whereas here, the learner has access to the entire corpus of unlabeled data.

It's also worth mentioning the tangentially related set cover problem (Slavık, 1997), where the goal is identify the smallest sub-collection of sets whose union equals the whole. The submodular set cover problem (Iwata & Nagano, 2009) involves finding a set that minimizes a modular cost function subject to a submodular function constraint. Guillory & Bilmes (2010) propose an active approach to solve the submodular set cover problem. Active covering however is a different problem that tries to recover the set of datapoints rather than a collection of subsets.

Our work is also related to the support estimation literature, which has a long history. Some works include Geffroy (1964); Devroye & Wise (1980); Korostelev & Tsybakov (1993); Cuevas et al. (1997); Biau et al. (2008). Our offline algorithm applies the classical support estimator on the positive samples found to find a covering for $\mathcal{X}_+$, which is the union of the ϵ-balls around the initial positive examples. Those works established both upper and lower bounds on ϵ of order $(\log m/m)^{1/D}$, which were key to the analysis for the offline algorithm.

More broadly, support estimation has also been studied under the name of one-class classification, where the goal is identify the examples of a particular class given only training examples from that class. There have been a wide range of approaches proposed including using SVMs (Schölkopf et al., 2001; Manevitz & Yousef, 2001), density functions (Hempstalk et al., 2008), clustering (Ypma & Duin, 1998), optimization (Crammer & Chechik, 2004), and deep learning (Ruff et al., 2018).

6. Experiments

In this section, we describe the details of our experimental results. We test the Explore-then-Commit algorithm (Algorithm 2) against a number of baselines including the offline algorithm (Algorithm 1). We note that these algorithms do not come with any additional hyperparameters.

6.1. Baselines

We propose a number of additional baselines based on the one-class classification methods implemented in scikit-learn (Pedregosa et al., 2011) that can score examples based on

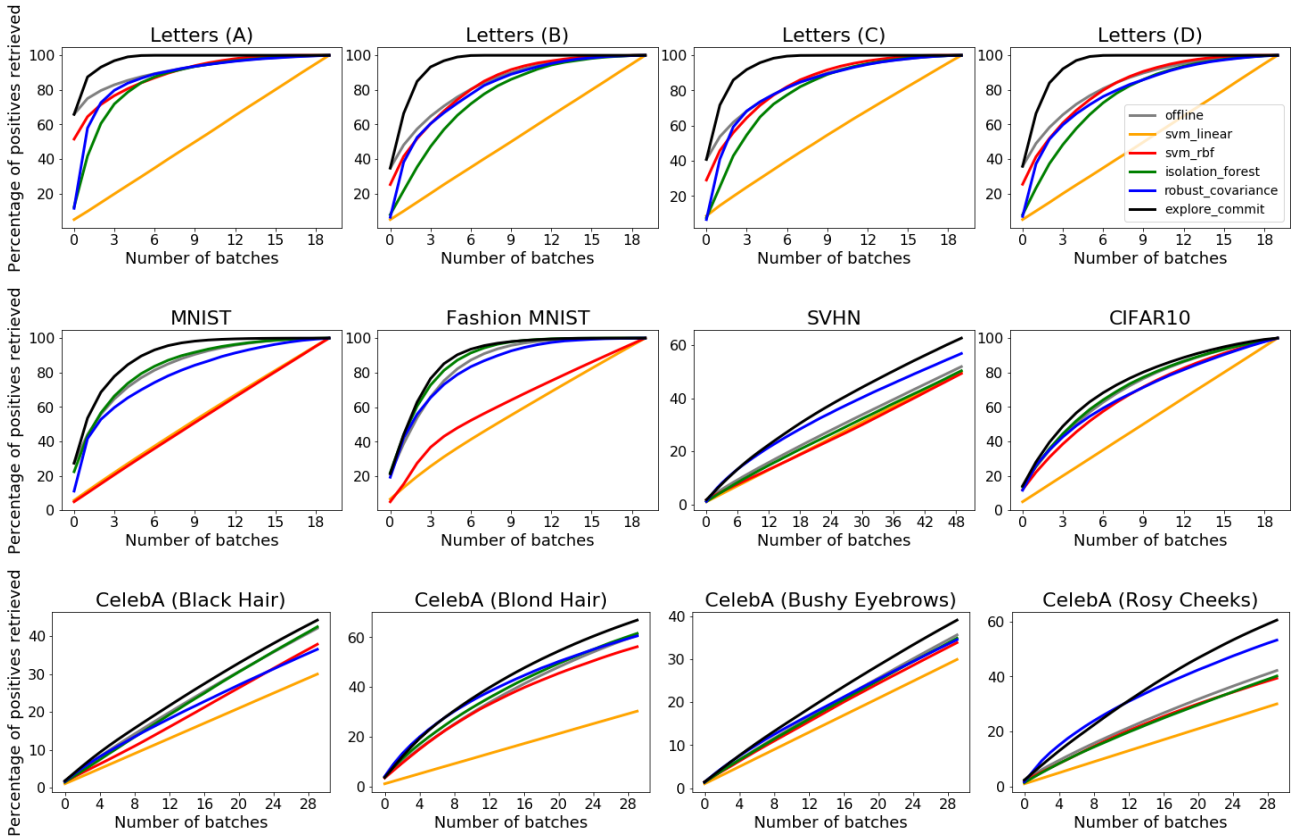

Figure 1. **Plots of percentage of all the positive examples retrieved after each batch**. **Top**: Letters Recognition dataset for the first 4 letters as the positive classes. **Middle**: Various datasets using the label 4 as the positive class. **Bottom**: CelebA using various attributes as the label. We compare our Explore-Commit method against the offline algorithm as well as the active variants of the baselines we tested across. We see that in all these cases, our method performs the best across batch sizes. Results averaged across 100 runs.

likelihood that they are in-class, namely the One-Class SVM (Schölkopf et al., 2001), Isolation Forest (Liu et al., 2008), and Robust Covariance (Rousseeuw & Driessen, 1999). For each of these baselines, we have variants: offline and active. The offline version trains the one-class classifier on the positive examples in the initial sample (see next subsection) and scores all of the unlabeled examples and then samples in order from most likely to least likely of being in-class. The active version retrains the one-class classifier on the label queried examples found thus far after each batch, and then scores the remaining examples to choose the next batch. All methods have the property of only utilizing the positive queried examples for learning.

We thus can list the baselines: **1.** Offline (O); **2.** Offline Linear SVM (O-LS); **3.** Active Linear SVM (A-LS); **4.** Offline RBF SVM (O-RS); **5.** Active RBF SVM (A-RS); **6.** Offline Isolation Forest (O-IF); **7.** Active Isolation Forest SVM (A-IF); **8.** Offline Robust Covariance (O-RC); **9.** Active Robust Covariance (A-RC).

6.2. Experiment Setup

For each of the datasets, we combine all of the data (i.e. any predetermined train/test/validation splits) into one dataset and operate on this dataset. We fix the initial sample size to a random stratified sample of 100 datapoints. We train a neural network on the initial sample and use the activations of the second-last layer (i.e. the layer immediately before the logits) as an embedding for the data and we fix the embedding throughout and all of the methods will operate on this embedding instead of the original input. This is because recent work has shown that it's effective to use classical methods on the intermediate embeddings of the neural network (Papernot & McDaniel, 2018; Bahri et al., 2020; Bahri & Jiang, 2021); moreover, the original features may have undesirable properties (i.e. relative scaling of the features, high-dimensional pixel data, etc) which can hurt the performance of some of the baselines.

For each dataset, we run experiments using each of the classes as the positive class, as the datasets we used were all multiclass (with the exception of CelebA, which came with a wide range of binary attributes which we used as classes).

Then for each of the datasets with the exception of SVHN and CelebA, we let the batch size be 5% of the remainder of the dataset (i.e. after removing the initial sample) to obtain 20 batches, and for SVHN and CelebA, due to their size, we let the batch size be 1% of the remainder of the dataset and ran for 50 batches for SVHN and 30 batches for CelebA. For all of the experimental results, we averaged across 100 runs randomizing over different initial samples and ran on a cluster of NVIDIA™ Tesla™ V100 Tensor Core GPUs.

6.3. Datasets and Embeddings

We tested on the following datasets:
1: UCI Letters Recognition (Dua & Graff, 2017), which has 20000 datapoints and 16 features based on various statistics on the pixel intensities of the original images of the letters, with 26 classes – one for each letter. To train the embedding, we used a fully-connected network with one hidden layer of 100 units and ReLU activations and trained for 20 epochs.
2: MNIST, with 70000 28x28 pixel grayscale images of handwritten digits and 10 classes. We use the same model and epochs as Letters for the embeddings.
3: Fashion MNIST (Xiao et al., 2017), with same dimensions and embedding training procedure as that of MNIST.
4: CIFAR10 with 60000 32x32 colour images in 10 classes. For the embeddings use a simple VGG (Zhang et al., 2015) network and extract the second-last layer which has 128 dimensions and train for 100 epochs.
5: SVHN (Netzer et al., 2011) with 99289 color images, cropped to 32x32 pixels. For the embeddings, we use LeNet5 (LeCun et al., 1998) and train for 20 epochs.
6. CelebA (Liu et al., 2018) a large-scale face attributes dataset with more than 162770; we resized the images to 28x28 celebrity images. The dataset has 40 attribute annotations which we use as separate binary classification tasks. We use the same embedding procedure as that of SVHN.

6.4. Hyperparameters

Our method doesn't come with any additional hyperparameters; however the baselines do require the tuning of hyperparameters. For these methods, we perform 5-fold cross-validation on the initial sample using accuracy as the metric (these methods as implemented in scikit-learn have predict methods which classifies whether an example is an outlier relative to the positive class). For the SVM methods, we tune the gamma parameter (kernel coefficient) and nu (upper bound on the fraction of training errors and a lower bound of the fraction of support vectors). For Isolation Forest, we tune the number of estimators in the ensemble. For Robust Covariance, we tune the proportion of contamination of the data set. For all of these aforementioned hyperparameters, we search over a grid of powers of two.

6.5. Evaluation Metrics

We plot the percentage of positive examples label queried across batches for each of the methods to illustrate the performance of each method. We also compute the area under the curve for each method, defined as the average number of positive examples retrieved across each batch, and use this as the primary evaluation metric to compare the methods. Since we average over 100 runs, we also compute an error band on the area under the curve metric. We do this by computing the standard deviation of percentage of positive examples retrieved for each of the 20 (or 50 and 30 in the case of SVHN and CelebA) batches. We average across these standard deviations and divide by square root of the number of runs to obtain an estimate of the standard deviation of the mean. We then form a 95% confidence band based on this and consider methods which have overlapping bands as statistical ties.

6.6. Results

We show the results for each baseline and dataset/label pair under our area under the curve metric in Table 2. Due to space, we could only show partial results for Letters and defer the rest along with the CelebA results to the Appendix. We nonetheless summarize all of the results here:
1. Letters. Our proposed method, Explore-then-Commit, outright outperforms all the other baselines on all 26 tasks.
2. MNIST. Our method again outright outperforms all the other baselines on all 10 tasks.
3. Fashion MNIST. Our method performs competitively on 9 out of the 10 tasks, with the next most competitive baseline (Active Isolation Forest) being competitive on 7 out of the 10 tasks.
4. CIFAR10. Here, our method performs competitively on 6 of the 10 tasks. It's worth noting that we only perform poorly when all of the methods perform poorly suggesting that in such settings not much learning is possible (i.e. from Table 2, we only perform non-competitively when the AUC metric is under 55%. A passive learner that samples uniformly at random is expected to have an AUC of 50%).
5. SVHN. Our method is competitive for all tasks and outright wins for all but one task.
6. CelebA. Our method is competitive for 32 out of the 40 tasks. Due to space, the results are shown in the Appendix. We again see a similar pattern as in CIFAR10 where our method only performs poorly when all of the methods perform poorly (i.e. we only perform non-competitively when the AUC metric is under 20%. For comparison, a passive learner is expected to have an AUC of 15%).

7. Conclusion

We have formalized the problem of active covering and introduced several baselines, including a principled active

Dataset	Label	O	O-LS	A-LS	O-RS	A-RS	O-IF	A-IF	O-RC	A-RC	EC (Ours)
Letters	A	91.14	52.52	52.49	84.81	89.02	59.52	84.69	64.27	86.87	**97.12**
	B	83.41	52.73	52.57	75.95	82.41	56.13	75.89	61.38	80.18	**93.76**
	C	84.48	56.19	56.08	75.92	83.78	55.78	78.68	59.21	81.92	**94.2**
	D	83.51	52.57	52.45	76.14	82.23	56.09	76.03	61.51	78.83	**93.73**
	E	78.6	52.85	52.99	74.84	81.41	55.77	73.7	60.17	77.27	**89.5**
	F	83.63	53.4	53.41	78.72	83.16	57.44	77.83	64.69	80.88	**94.0**
	G	82.23	52.72	52.67	75.76	81.88	58.15	74.58	63.45	79.79	**92.35**
MNIST	0	86.67	81.81	81.96	52.53	52.95	83.44	90.8	74.31	86.48	**94.44**
	1	95.89	55.22	55.11	58.47	90.31	90.16	94.27	87.34	89.8	**96.46**
	2	75.86	60.64	60.37	52.54	52.56	72.66	80.18	62.81	77.83	**85.47**
	3	80.77	60.64	60.36	52.52	52.65	76.67	84.31	66.87	81.03	**87.63**
	4	83.05	54.23	54.14	52.47	53.08	76.86	83.61	66.71	78.31	**89.14**
	5	75.59	52.88	52.81	52.51	52.63	61.65	69.44	57.82	71.18	**87.24**
	6	86.53	59.98	59.77	52.49	54.03	81.19	88.33	67.37	81.95	**93.24**
	7	87.05	55.83	55.63	52.54	57.02	80.71	87.26	70.14	81.76	**91.62**
	8	75.7	56.27	56.17	52.49	52.62	69.73	78.3	61.71	77.32	**83.37**
	9	84.91	54.64	54.7	52.51	55.06	77.22	84.66	67.88	79.79	**90.71**
Fashion MNIST	0	87.81	54.51	54.49	52.9	66.76	86.35	**90.14**	81.5	87.44	**89.75**
	1	94.73	55.84	55.67	55.12	85.21	92.67	94.73	90.42	92.54	**95.93**
	2	84.44	55.57	55.57	52.72	63.65	82.97	**87.6**	78.46	85.6	**87.19**
	3	88.86	53.15	53.15	52.64	63.56	85.39	89.74	83.08	87.06	**91.29**
	4	84.9	56.47	56.41	52.68	62.54	83.23	**87.25**	79.68	83.61	**88.09**
	5	88.09	52.54	52.52	52.62	57.14	79.59	84.7	82.92	75.2	**89.16**
	6	77.31	52.5	52.5	52.98	63.62	75.94	**81.63**	71.46	80.18	**81.1**
	7	94.41	52.47	52.49	52.97	69.91	92.66	**95.06**	90.72	93.05	**95.17**
	8	82.86	55.43	55.46	52.5	54.1	78.23	**86.56**	78.33	83.97	**85.96**
	9	92.5	70.39	70.13	52.59	58.31	90.93	**94.12**	90.35	91.4	93.49
CIFAR10	0	67.06	54.33	54.27	64.84	64.8	64.6	68.03	65.49	69.36	**70.69**
	1	**57.24**	52.57	52.54	**55.67**	55.01	54.25	53.68	**57.68**	50.3	54.06
	2	65.43	52.48	52.51	59.75	59.92	62.09	64.61	63.42	65.15	**68.71**
	3	**53.98**	52.5	52.51	**53.21**	**53.28**	**53.34**	52.8	**54.13**	50.09	52.2
	4	70.94	52.5	52.55	66.68	67.52	68.54	71.52	67.95	68.29	**73.97**
	5	**57.59**	52.53	52.48	56.13	56.09	56.55	56.7	**58.83**	53.12	53.9
	6	72.16	52.48	52.49	67.67	67.89	67.79	70.87	70.78	65.26	**74.73**
	7	**58.51**	52.54	52.53	56.0	55.93	56.48	**57.7**	**58.07**	53.92	**58.36**
	8	**70.25**	52.48	52.47	66.67	66.86	67.92	**71.57**	68.13	69.91	**71.42**
	9	62.79	52.54	52.49	61.8	61.74	63.51	**66.18**	63.97	62.65	55.63
SVHN	0	31.11	25.47	25.49	28.54	29.89	28.0	30.12	31.57	**39.59**	**39.67**
	1	28.23	25.53	25.52	25.63	25.25	25.08	25.44	32.81	35.19	**37.32**
	2	28.66	25.53	25.52	26.28	26.49	26.32	26.69	30.86	32.68	**34.31**
	3	28.19	25.57	25.56	26.11	26.47	26.34	26.79	29.37	31.0	**33.81**
	4	28.01	25.51	25.49	25.54	25.16	25.21	26.66	27.31	33.4	**36.31**
	5	29.02	25.56	25.53	26.55	26.98	26.77	27.75	29.67	32.81	**34.44**
	6	28.8	25.49	25.5	26.37	26.6	26.24	27.72	28.7	32.34	**35.29**
	7	29.36	25.52	25.48	26.46	26.18	25.7	27.22	27.79	35.14	**37.62**
	8	28.04	25.51	25.48	26.29	27.03	26.41	27.43	27.29	31.43	**33.37**
	9	28.48	25.49	25.49	26.82	27.5	26.28	28.04	27.62	32.83	**34.36**

Table 2. **Area under the curve metric for various benchmark image-based datasets**. For each of the datasets and possible labels, we show the area under the curve metric averaged across 100 runs, with the top value bolded (any methods whose 95% confidence intervals overlap were considered statistical ties). Due to space, we show the rest of the Letters results as well as the CelebA results in the Appendix.

approach that attains better guarantees than the offline algorithm. We showed in experiments that our proposed Explore-the-Commit algorithm has strong performance against a number of baselines while having desirable properties including not having additional hyperparameters, and not needing to store or use the queried negative examples. Future work involves extending theoretical analysis relaxing the hard boundary density assumption on $\mathcal{X}_+$, letting $\mathcal{X}_+$ be a lower dimensional manifold embedded in the D-dimensional space (rather than being full-dimensional) and attain excess query cost guarantees that depend on this lower dimension, and investigating the computational and privacy implications of such approaches.

References

Acosta, I. C. C., Khodadadzadeh, M., Tusa, L., Ghamisi, P., and Gloaguen, R. A machine learning framework for drill-core mineral mapping using hyperspectral and high-resolution mineralogical data fusion. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 12(12):4829–4842, 2019.

Awoyemi, J. O., Adetunmbi, A. O., and Oluwadare, S. A. Credit card fraud detection using machine learning techniques: A comparative analysis. In *2017 International Conference on Computing Networking and Informatics (ICCNI)*, pp. 1–9. IEEE, 2017.

Bahri, D. and Jiang, H. Locally adaptive label smoothing for predictive churn. *arXiv preprint arXiv:2102.05140*, 2021.

Bahri, D., Jiang, H., and Gupta, M. Deep k-nn for noisy labels. In *International Conference on Machine Learning*, pp. 540–550. PMLR, 2020.

Biau, G., Cadre, B., and Pelletier, B. Exact rates in density support estimation. *Journal of Multivariate Analysis*, 99 (10):2185–2207, 2008.

Boneh, A. and Hofri, M. The coupon-collector problem revisited—a survey of engineering problems and computational methods. *Stochastic Models*, 13(1):39–66, 1997.

Chaudhuri, K. and Dasgupta, S. Rates of convergence for the cluster tree. In *Advances in neural information processing systems*, pp. 343–351, 2010.

Crammer, K. and Chechik, G. A needle in a haystack: local one-class optimization. In *Proceedings of the twenty-first international conference on Machine learning*, pp. 26, 2004.

Cuevas, A., Fraiman, R., et al. A plug-in approach to support estimation. *The Annals of Statistics*, 25(6):2300–2312, 1997.

Devroye, L. and Wise, G. L. Detection of abnormal behavior via nonparametric estimation of the support. *SIAM Journal on Applied Mathematics*, 38(3):480–488, 1980.

Dua, D. and Graff, C. UCI machine learning repository, 2017. URL http://archive.ics.uci.edu/ml.

García-Recuero, Á. Discouraging abusive behavior in privacy-preserving online social networking applications. In *Proceedings of the 25th International Conference Companion on World Wide Web*, pp. 305–309, 2016.

Garivier, A., Lattimore, T., and Kaufmann, E. On explore-then-commit strategies. *Advances in Neural Information Processing Systems*, 29:784–792, 2016.

Garnett, R., Krishnamurthy, Y., Xiong, X., Schneider, J., and Mann, R. Bayesian optimal active search and surveying. In *Proceedings of the 29th International Coference on International Conference on Machine Learning*, pp. 843–850, 2012.

Geffroy, J. Sur un probleme d'estimation géométrique. *Publ. Inst. Statist. Univ. Paris*, 13:191–210, 1964.

Gorin, A. On the volume of tubes. *Illinois Journal of Mathematics*, 27(1):158–171, 1983.

Guillory, A. and Bilmes, J. Interactive submodular set cover. *arXiv preprint arXiv:1002.3345*, 2010.

Helmbold, D. P., Littlestone, N., and Long, P. M. Apple tasting. *Information and Computation*, 161(2):85–139, 2000.

Hempstalk, K., Frank, E., and Witten, I. H. One-class classification by combining density and class probability estimation. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pp. 505–519. Springer, 2008.

Iwata, S. and Nagano, K. Submodular function minimization under covering constraints. In *2009 50th Annual IEEE Symposium on Foundations of Computer Science*, pp. 671–680. IEEE, 2009.

Jain, L. and Jamieson, K. G. A new perspective on pool-based active classification and false-discovery control. In *Advances in Neural Information Processing Systems*, pp. 13992–14003, 2019.

Jiang, H. Density level set estimation on manifolds with db-scan. In *International Conference on Machine Learning*, pp. 1684–1693. PMLR, 2017.

Jiang, H. Non-asymptotic uniform rates of consistency for k-nn regression. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pp. 3999–4006, 2019.

Jiang, H., Jiang, Q., and Pacchiano, A. Learning the truth from only one side of the story. *arXiv preprint arXiv:2006.04858*, 2020.

Jiang, S., Malkomes, G., Abbott, M., Moseley, B., and Garnett, R. Efficient nonmyopic batch active search. In *Advances in Neural Information Processing Systems*, pp. 1099–1109, 2018.

Jiang, S., Garnett, R., and Moseley, B. Cost effective active search. In *Advances in Neural Information Processing Systems*, pp. 4880–4889, 2019.

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

Korostelev, A. P. and Tsybakov, A. B. Estimation of the density support and its functionals. *Problemy Peredachi Informatsii*, 29(1):3–18, 1993.

Kpotufe, S. k-nn regression adapts to local intrinsic dimension. *arXiv preprint arXiv:1110.4300*, 2011.

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Li, K., Zhong, Z., and Ramaswamy, L. Privacy-aware collaborative spam filtering. *IEEE Transactions on Parallel and Distributed systems*, 20(5):725–739, 2008.

Liu, F. T., Ting, K. M., and Zhou, Z.-H. Isolation forest. In *2008 Eighth IEEE International Conference on Data Mining*, pp. 413–422. IEEE, 2008.

Liu, Z., Luo, P., Wang, X., and Tang, X. Large-scale celebfaces attributes (celeba) dataset. *Retrieved August*, 15: 2018, 2018.

Manevitz, L. M. and Yousef, M. One-class svms for document classification. *Journal of machine Learning research*, 2(Dec):139–154, 2001.

Netzer, Y., Wang, T., Coates, A., Bissacco, A., Wu, B., and Ng, A. Y. Reading digits in natural images with unsupervised feature learning. 2011.

Nobata, C., Tetreault, J., Thomas, A., Mehdad, Y., and Chang, Y. Abusive language detection in online user content. In *Proceedings of the 25th international conference on world wide web*, pp. 145–153, 2016.

Ou-Yang, S.-s., Lu, J.-y., Kong, X.-q., Liang, Z.-j., Luo, C., and Jiang, H. Computational drug discovery. *Acta Pharmacologica Sinica*, 33(9):1131–1140, 2012.

Papernot, N. and McDaniel, P. Deep k-nearest neighbors: Towards confident, interpretable and robust deep learning. *arXiv preprint arXiv:1803.04765*, 2018.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., et al. Scikit-learn: Machine learning in python. *the Journal of machine Learning research*, 12:2825–2830, 2011.

Pelletier, B. Kernel density estimation on riemannian manifolds. *Statistics & probability letters*, 73(3):297–304, 2005.

Rousseeuw, P. J. and Driessen, K. V. A fast algorithm for the minimum covariance determinant estimator. *Technometrics*, 41(3):212–223, 1999.

Ruff, L., Vandermeulen, R., Goernitz, N., Deecke, L., Siddiqui, S. A., Binder, A., Müller, E., and Kloft, M. Deep one-class classification. In *International conference on machine learning*, pp. 4393–4402, 2018.

Schölkopf, B., Platt, J. C., Shawe-Taylor, J., Smola, A. J., and Williamson, R. C. Estimating the support of a high-dimensional distribution. *Neural computation*, 13(7): 1443–1471, 2001.

Singh, A., Scott, C., Nowak, R., et al. Adaptive hausdorff estimation of density level sets. *The Annals of Statistics*, 37(5B):2760–2782, 2009.

Slavık, P. A tight analysis of the greedy algorithm for set cover. *Journal of Algorithms*, 25(2):237–254, 1997.

Xiao, H., Rasul, K., and Vollgraf, R. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

Ypma, A. and Duin, R. P. Support objects for domain approximation. In *International Conference on Artificial Neural Networks*, pp. 719–724. Springer, 1998.

Zhang, X., Zou, J., He, K., and Sun, J. Accelerating very deep convolutional networks for classification and detection. *IEEE transactions on pattern analysis and machine intelligence*, 38(10):1943–1955, 2015.

Zhao, P. and Lai, L. Analysis of knn density estimation. *arXiv preprint arXiv:2010.00438*, 2020.