Active Learning Through a Covering Lens

Anonymous Author(s)
Affiliation
Address
email

Abstract

Deep active learning aims to reduce the annotation cost for deep neural networks, which are notoriously data-hungry. Until recently, deep active learning methods struggled in the *low-budget* regime, where only a small amount of samples are annotated. The situation has been alleviated by recent advances in self-supervised representation learning methods, which impart the geometry of the data representation with rich information about the points. Taking advantage of this progress, we study the problem of subset selection for annotation through a "covering" lens, proposing *ProbCover* – a new active learning algorithm for the low budget regime, which seeks to maximize *Probability Coverage*. We describe a dual way to view our formulation, from which one can derive strategies suitable for the high budget regime of active learning, related to existing methods like *Coreset*. We conclude with extensive experiments, evaluating *ProbCover* in the low budget regime. We show that our principled active learning strategy improves the state-of-the-art in the low-budget regime in several image recognition benchmarks. This method is especially beneficial in semi-supervised settings, allowing state-of-the-art semi-supervised methods to achieve high accuracy with only a few labels.

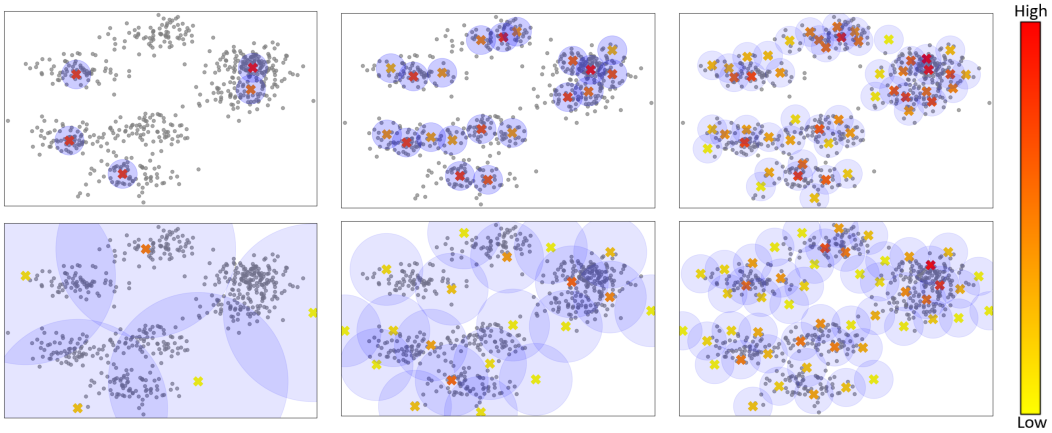
1 Introduction

For the most part, deep learning technology critically depends on access to large amounts of annotated data. Yet annotations are costly and remain so even in our era of *Big Data*. Deep active learning (AL) aims to alleviate this problem by improving the quality of the annotated data. Specifically, given a fixed budget b of examples that can be annotated, and some deep learner, AL algorithms aim to query those b examples that will most benefit this learner.

In order to optimally choose unlabeled examples to be annotated, most deep AL strategies follow some combination of two main principles: 1) Uncertainty sampling (e.g, [3, 13, 14, 24, 25, 35, 46]), in which examples that the learner is most uncertain about are picked, to maximize the added value of the new annotations. 2) Diversity Sampling (e.g, [16, 36, 39, 41, 44]), in which examples are chosen from diverse regions of the data distribution, to represent it wholly and reduce redundancy in the annotation.

Most AL methods fail to improve over random selection when the annotation budget is very small [2, 7, 20, 31, 34, 38, 51], a phenomenon sometimes termed "cold start" [15, 22, 47]. When the budget contains only a few examples, they will struggle to improve the model's performance, not even reaching the accuracy of the random baseline. Recently, it was shown that uncertainty sampling is inherently unsuited for the low budget regime, which may explain the cold start phenomenon [18]. The low-budget scenario is relevant in many applications, especially those requiring an expert tagger whose time is expensive (e.g a radiologist tagger for tumor detection). If we want to expand deep learning to new domains, overcoming the cold start problem is an ever-important task.

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.



(a) 5 Samples (b) 20 Samples (c) 50 Samples

Figure 1: *ProbCover* selection (top) vs *Coreset* selection (bottom) of 5/20/50 samples (out of 600). Selected points are marked by **x**, which is color-coded by density[1] (color code bar to the right). The covered area is marked in light blue. *Coreset* attempts to minimize ball size, constrained by complete coverage, while *ProbCover* attempts to maximize coverage, constrained by a fixed ball size. Note that especially in low budgets, *Coreset* selects outliers of the distribution (yellow), while *ProbCover* selects from dense regions of the distribution (red).

In this work, we focus on understanding the very low budget regime of AL, where the budget of b examples cannot dependably represent the data distribution. In Section 2 we model the problem as *Max Probability Cover*, defined as follows: given some data distribution, and a radius δ, select the b examples which maximize the probability of the union of balls with radius δ around each example. Under labeling separation assumptions on the data (which are realistic in semantic embedding spaces), *Max Probability Cover* is the optimal strategy for nearest-neighbor models, in that it minimizes an upper bound on their generalization error. In Section 2.3, we show a connection with existing deep AL methods, like *Coreset* [36], and explain why those methods are more suitable for the high-budget regime than the low-budget regime. This phenomenon is visualized in Fig. 1.

When the data distribution is not known, as is the case in most practical settings, we show that *Max Probability Cover* can be reduced to *Max Coverage* when using the empirical data distribution. *Max Coverage* is known to be NP-hard [33]. We then adapt a greedy algorithm for the selection of b examples from a fixed finite training set of unlabeled examples (the training set), which guarantees $1 - \frac{1}{e}$ approximation to the problem (see Section 3). We call this new method *ProbCover*.

In Section 4 we empirically evaluate the performance of *ProbCover* on several computer vision datasets, including CIFAR-10, CIFAR-100, Tiny-ImageNet, and subsets of ImageNet. *ProbCover* is thus shown to significantly outperform all alternative deep AL methods in the very low budget regime. Additionally, *ProbCover* improves the performance of state-of-the-art semi-supervised methods, which were thought until recently to make AL redundant [6], allowing for the learning of computer vision tasks with very few annotated examples.

Relation to prior art In addition to the principle of uncertainty sampling, many AL strategies employ the principle of diversity sampling, querying examples from different parts of the distribution (e.g, [1, 15, 17, 21, 25, 29, 37, 45]). While diverse sets of examples are expected to cover more of the training distribution, we maximize the coverage directly, which is especially beneficial in the low-budget regime.

Recently, a line of works propose AL methods based on an approximation to the facility location problem [12, 29, 36, 42, 43, 50], which is a variation of a covering problem. In the facility location problem, the entire distribution is covered with a fixed number of balls, which can vary in size, while in *ProbCover* the size of the balls is fixed, and we allow the coverage of smaller parts of the distribution. While this difference may seem minor, in the low-budget regime, when the budget is not large enough to represent the data, examples chosen by the facility location problem are not representative, and lead to poor performance (see Fig. 9 and discussion in Section 2.3)

[1] Density is measured using Gaussian Kernel Density Estimation.

Summary of contribution

- Develop a theoretical framework to analyze AL strategies in embedding spaces, with "dual" low and high-budget interpretations.
- Introduce *ProbCover*, a low-budget AL strategy motivated by our framework, which significantly outperforms other methods in the low-budget regime.
- Demonstrate the outstanding competence of *ProbCover* in semi-supervised learning with very few labeled examples.

2 Theoretical Analysis

We develop a rigorous framework, within which different AL methods can be evaluated and designed. Since AL is about the selection of examples, we chose to analyze the Nearest Neighbor (NN) classification model, as it is a model that depends almost exclusively on the set of training examples and does not involve almost any external inductive bias.

Accordingly, in Section 2.1 we develop a bound on the generalization error in 1-NN models. The bound involves two terms, which ideally should be jointly minimized. We show that one term is more suitable for the low budget regime, where the number of labeled examples is small, while the other term is more suitable for the high budget regime, where the number of labeled examples is large.

In the low budget regime, we are likely to benefit more from minimizing the first term, which leads us to the *Max Probability Cover* problem defined in Section 2.2. We then show that this problem is NP-hard. Furthermore, when the data distribution is not known and is being approximated by the empirical distribution, we show that it is equivalent to the *Max Coverage* Problem. Our method, described in Section 3, is designed to solve this problem.

In contrast, in the high budget regime, we are likely to benefit more from minimizing the second term, which is equivalent to *optimal facility location*, a problem that is also known to be NP-hard and which the AL *Coreset* algorithm [36] is designed to solve. Not surprisingly, *Coreset* is indeed only effective in the high budget regime, as we empirically show in Section 4.

2.1 Bounding the Generalization Error

We shall now derive a bound on the generalization error of a 1 Nearest Neighbor (1-NN) classifier. We start with some necessary notations and definitions. Most important is the assumption of δ-purity, which states that most of the time, points that are less than δ apart have the same label. We then prove a lemma, showing that given a labeled set L and the coverage it achieves, and given the δ-purity assumption, the probability of a point being inside this cover and still being falsely labeled is small. From this, we finally derive our bound on the generalization error, which is stated in Thm. 1.

Notations Let $\mathbb{X}$ denote the input domain whose underlying distribution is $\mathcal{D}$, and let $\mathbb{Y} = [k]$ denote the target domain. Assume that a true labeling function $f : \mathbb{X} \to \mathbb{Y}$ exists. The *Generalization Error* of a hypothesis $\hat{f} : \mathbb{X} \to \mathbb{Y}$ is defined as $L_{\mathcal{D}}(\hat{f}) = \mathbb{E}_{x\sim\mathcal{D}}\left[\mathbb{1}_{\hat{f}(x)\neq f(x)}\right]$.

Let $X = \{x_i\}_{i=1}^{m} \sim \mathcal{D}^m$ denote an unlabeled set of points, and $b \leq m$ the annotation budget. Let $L \subseteq X$ denote the labeled set, where $|L| = b$. Let $B_\delta(x) = \{x' : \|x' - x\|_2 \leq \delta\}$ denote a ball centered at x of radius δ. Let $C(L, \delta) = \bigcup_{c\in L} B_\delta(c)$ denote the region covered by δ-balls centered at the labeled examples in L (sometimes abbreviated as C). We call $C(L, \delta)$ the *covered region* and $P(C)$ the *coverage*.

Definition 2.1. We say that a ball $B_\delta(x)$ is *pure* if $\forall x' \in B_\delta(x) : f(x') = f(x)$.

Definition 2.2. We define the *purity* of δ as: $\pi(\delta) = P(\{x : B_\delta(x) \text{ is pure}\})$. Notice that $\pi(\delta)$ is monotonically decreasing.

Let $\hat{f}$ denote the 1-NN classifier based on L. We split the covered region $C(L, \delta)$ to two sets:

$$C_{right} = \{x \in C \colon \hat{f}(x) = f(x)\},\ C_{wrong} = C \setminus C_{right}$$

Lemma 1. $C_{wrong} \subseteq \{x : B_\delta(x) \text{ is not pure}\}$

Proof. Let $x \in C_{wrong}$. Let $c \in L$ denote the nearest neighbor to x. Then they have the same predicted label, $\hat{f}(x) = \hat{f}(c)$, and $f(c) = \hat{f}(c)$ because c is labeled. Since x is labeled wrongly, $\hat{f}(x) \neq f(x)$, which means

$$f(c) = \hat{f}(c) = \hat{f}(x) \neq f(x)$$

Finally, since $x \in C_{wrong} \subseteq C$ is in the coverage, $d(x, c) < \delta$, which means that $c \in B_\delta(x)$ with a different label and so $B_\delta(x)$ is not pure. □

Corollary 1.

$$P(C_{wrong}) \leq P(\{x : B_\delta(x) \text{ is not pure}\}) = 1 - \pi(\delta)$$

Theorem 1. The generalization error of the 1-NN classifier $\hat{f}$ is bounded by

$$L_{\mathcal{D}}(\hat{f}) \leq (1 - P(x \in C)) + (1 - \pi(\delta)) \tag{1}$$

Proof.

$$\begin{aligned} L_{\mathcal{D}}(\hat{f}) &= \mathbb{E}[\mathbb{1}_{f(x) \neq \hat{f}(x)}] = \mathbb{E}[\mathbb{1}_{f(x) \neq \hat{f}(x)} \mathbb{1}_{x \notin C}] + \mathbb{E}[\mathbb{1}_{f(x) \neq \hat{f}(x)} \mathbb{1}_{x \in C}] \\ &\leq P(x \notin C) + \mathbb{E}[\mathbb{1}_{f \neq \hat{f}} \mathbb{1}_{x \in C_{right}}] + \mathbb{E}[\mathbb{1}_{f(x) \neq \hat{f}(x)} \mathbb{1}_{x \in C_{wrong}}] \\ &\leq P(x \notin C) + 0 + P(x \in C_{wrong}) \\ &\leq (1 - P(x \in C)) + (1 - \pi(\delta)) \end{aligned}$$

□

Notice that this gives us a different bound for different δ values, and different labeled sets L. This bound introduces a trade-off: as δ increases, the *coverage* increases, but the *purity* decreases. Therefore, we wish to find the pair δ, L that gives the tightest bound.

Discussion We can interpret (1) in the context of two boundary conditions of AL: high-budget and low-budget. In the high-budget regime, achieving full coverage $P(C) = 1$ is easy as we have many points, and the remaining challenge is to reduce $1 - \pi(\delta)$. Since $\pi(\delta)$ is monotonically decreasing, we can seek to minimize δ subject to the constraint $P(C) = 1$. This is similar to *Coreset* [36]. In the low-budget regime, full coverage entails very low purity, which (if sufficiently low) makes the bound trivially 1. Thus, instead of insisting on full coverage, we can fix a δ that yields "large enough" purity $\pi(\delta) > 0$, and then seek a labeled set L that maximizes the coverage $P(C)$. We call this problem *Max Probability Cover*.

2.2 Max Probability Cover

Definition 2.3 (*Max Probability Cover*)**.** Fix $\delta > 0$, and obtain a subset $L \subset X$, $|L| = b$, that maximizes *coverage*

$$\underset{L \subseteq X; |L| = b}{\text{argmax}} \; P(\bigcup_{x \in L} B_\delta(x)) \tag{2}$$

An optimal solution to (2) would minimize the bound in (1), when δ is fixed (which fixes $1 - \pi(\delta)$).

Unfortunately, when moving to practical settings there are two obstacles. The first is about complexity:

Theorem 2. *Max Probability Cover* is NP-hard.

Proof. (Sketch, full proof in App. B) We construct a reduction from an established NP-hard problem (*Max Coverage*, see Def. A.1) to *Max Probability Cover*. For the collection of subsets $S = \{S_1, \ldots, S_m\}$, we consider the space $\mathbb{R}^m$ and a collection of δ-balls $\{B_\delta(x_i)\}_{i=1}^m$ with the special property that for any subset of the balls, their intersection includes at least one unique point (see Lemma 2). We then assign each S_i to $B_\delta(x_i)$, and each element in S_i is sent to a point in the intersection of all the balls assigned to subsets that contain it. Each such point then defines a Dirac measure, the normalized sum of which determines a probability distribution on $\mathbb{R}^m$. The selection of δ-balls that is the solution to the *Max Probability Cover* can be translated back to a selection of subsets, which is the solution to the original *Max Coverage* problem. □

The second problem is that we are almost always not supplied with the data distribution, and even if we were, the probability computation would most likely be intractable and hard to approximate.

If we are not supplied with the data distribution, we can use the *empirical distribution* $\tilde{P}(A) = \frac{1}{n}\sum_{i=1}^{n} \mathbb{1}_{x_i \in A}$ to approximate it. This gives us the following result:

Proposition. When P is the empirical distribution $\tilde{P}$, the *Max Probability Cover* objective function is equivalent to the *Max Coverage* objective, with $\{B_\delta(x_i) \cap X\}_{i=1}^{n}$ as the collection of subsets.

Proof. Given a labeled set $L = \{x_i\}_{i=1}^{b}$, we show equality of objectives up to constant $\frac{1}{|X|}$

$$P\left(\bigcup_{i=1}^{b} B_\delta(x_i)\right) = P\left(\{y \in \mathbb{R}^d \mid \exists i \quad \|x_i - y\| < \delta\}\right) = \frac{1}{|X|}\left|\{x \in X \mid \exists i \quad \|x_i - x\| < \delta\}\right|$$
$$= \frac{1}{|X|}\left|\bigcup_{i=1}^{b}(B_\delta(x_i) \cap X)\right|$$

□

2.3 The "Duality" of *Max Probability Cover* and *Coreset*

The *Coreset* AL method [36] minimizes the objective

$$\delta(L) = \max_{x \in X} \min_{c \in L} d(x, c) = \min\{\delta \in \mathbb{R}_+ \colon X \subseteq \bigcup_{c \in L} B_\delta(c)\}$$

We can rewrite the above in the language of distributions as

$$\delta'(L) = \min\{\delta \in \mathbb{R}_+ \colon P(\bigcup_{c \in L} B_\delta(c)) = 1\}$$

If we use the empirical distribution then $\delta(L) = \delta'(L)$. In this framework we can say that *Max Probability Cover* and *Coreset* are dual problems (dual in a loose sense) as follows:

1. *Max Probability Cover* minimizes the generalization error bound (1) when we fix δ and seek to maximize the coverage, which is suitable for the low budget regime.
2. *Coreset* minimizes the generalization error bound (1) when we fix the coverage to 1 and minimize δ, which is suitable for the high budget regime because only then can we fix the coverage to 1.

This duality is visualized in Fig. 1.

3 Method: *ProbCover*

In order to deliver a practical method, we first note that our approach implicitly relies on the existence of a good embedding space [4, 9, 49], where distance is correlated with semantic similarity, and where similar points are likely to bunch together in high-density regions. In such spaces the labeling is also likely to be correlated with distance, making it friendly to 1-NN classification. Additionally, it implies the existence of large enough δ balls, with good purity and coverage properties.

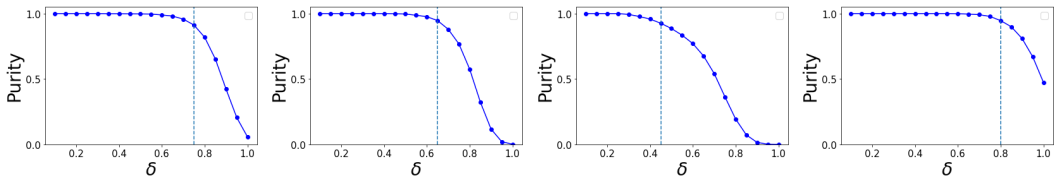
Secondly, we note that since *Max Coverage* is NP-hard, it cannot be solved efficiently. Instead, as its objective is a submodular monotone function, we use the greedy approximate algorithm, which achieves $\left(1 - \frac{1}{e}\right)$-approximation [26]. Using an algorithm that gives a better approximation is impractical, see App. D.3 for an in-depth discussion. We now turn to describe the greedy algorithm and our method for estimating the ball size δ.

3.1 Greedy Algorithm

The algorithm goes as follows: First, construct a directed graph $G = (V, E)$, with $V = X$ and $(x, x') \in E \iff x' \in B_\delta(x)$, where each vertex represents a specific example, and there is an edge between two vertices (x, x') if x' is covered by the δ-ball centered at x. The algorithm then performs b iterations of the following 2 steps: (i) pick the vertex x_{max} with the highest out-degree for annotation; (ii) remove all incoming edges to x_{max} and its neighbors. See Alg. 1 for pseudo-code.

Algorithm 1 *ProbCover*

Input: unlabeled pool U**, labeled pool** L**, budget** b**, ball-size** δ
Output: a set of points to query
$G = (V, E) \leftarrow$ Construct Graph
for all $c \in L$ **do**
  $C = \{x : (c, x) \in E\}$
  $E = E \setminus \{(z, x) : x \in C\}$
end for
Queries $\leftarrow \emptyset$
for all i=1,...,b **do**
  Choose $c \in U$ with highest out-degree in G
  $C = \{x : (c, x) \in E\}$
  $E = E \setminus \{(z, x) : x \in C\}$
  Queries = Queries $\cup$ $\{c\}$
end for
return Queries



(a) CIFAR-10 (b) CIFAR-100 (c) Tiny-ImageNet (d) ImageNet-100

Figure 2: Ball purity, as a function of δ, estimated from the unlabeled data (see text). The dashed line marks the highest δ, after which purity drops below $\alpha = 0.95$.

3.2 Estimating δ

Our algorithm requires us to specify the hyper-parameter δ, the ball radius, whose value depends on details of the embedding space. In choosing δ, we need to consider the trade-off between large coverage $P(C)$ and high purity $\pi(\delta)$. We resolve this trade-off with a heuristic, where we pick the largest δ possible, while maintaining purity above a certain threshold $\alpha \in (0, 1)$. Specifically,

$$\delta^* = \max \{\delta : \pi(\delta) \geq \alpha\}$$

Importantly, α is more intuitive to tune, and is kept constant across different datasets (unlike δ). We still need to be able to estimate the purity $\pi(\delta)$, which depends on the labels, from unlabeled data. To address this challenge, we estimate purity using unsupervised representation learning and clustering. First, we cluster self-supervised features using k-Means with k equal to the number of classes. For a given δ, we compute the purity $\pi(\delta)$ using the clustering labels as pseudo-labels for each example. Searching for the best δ, we repeat the process and pick the largest δ so that at least $\alpha = 0.95$ of the balls are pure.

In Fig. 2, we plot the percentage of pure balls across different datasets as a function of δ, where the dashed line represents the δ^* chosen by our method.

4 Empirical Results

We report a set of empirical results, comparing *ProbCover* to other AL strategies in a variety of settings. We focus on the very low budget regime, with a budget size b of a similar order of magnitude as the number of classes. Note that since the data is picked from an unlabeled pool, chances are that the initial labeled set is not going to be balanced across classes, and in the early stages of training some classes will almost always be missing.

4.1 Methodology

Three deep AL frameworks are evaluated:

(i) *Fully supervised*: train the deep learner only on the annotated data, as a fully supervised task.

(ii) *Semi-supervised by transfer learning*: create a representation of the data by training with a self-supervised task on the unlabeled data, then train a simple model on the ensuing representation in a supervised manner. This framework is intended to capture the basic benefits of semi-supervised learning, regardless of the added benefits provided by the sophistication of modern semi-supervised learning methods.

(iii) *Fully semi-supervised*: train a semi-supervised model on both the annotated and unlabeled data using FlexMatch [48].

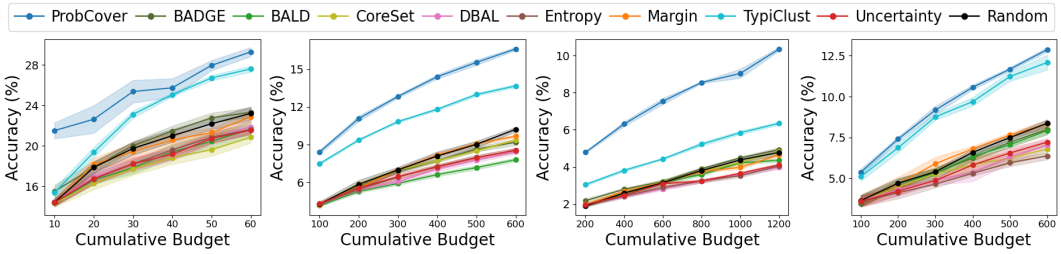
In frameworks (i) and (ii) we adopt the evaluation framework created by Munjal et al. [32], in which we can compare multiple deep AL strategies in a principled way. In framework (iii), we adopt the code and hyper-parameters provided by FlexMatch.

When evaluating frameworks (i) and (ii), we compare *ProbCover* to 9 deep AL strategies as baselines. (1) ***Random*** query uniformly. (2)-(4) query examples with the lowest score, using the following basic scores: (2) ***Uncertainty*** – max softmax output, (3) ***Margin*** – margin between the two highest softmax outputs, (4) ***Entropy*** – inverse entropy of softmax outputs. (5) ***BADGE*** [1]. (6) ***DBAL*** [14]. (7) ***TypiClust*** [18]. (8) ***BALD*** [25]. (9) ***Coreset*** [36]. We note that while most baseline methods are suitable for the high budget regime, *TypiClust* is also suitable for the low budget regime.

These AL methods are evaluated on the following classification datasets: CIFAR-10/100 [27], Tiny-ImageNet [28] and ImageNet-50/100 (the latter group includes subsets of ImageNet [11] containing 50/100 classes respectively, following Van Gansbeke et al. [40]). When considering CIFAR-10/100, and TinyImageNet, we use as input the embedding of SimCLR [8]. When considering ImageNet-50/100 we use as input the embedding of DINO [5]. Results on ImageNet-50 are deferred to the appendix and can be seen in App. E. Details on the specific networks and hyperparameters can be found in App. D.2. When evaluating frameworks (i) and (ii), we perform 5 active learning rounds, querying a fixed budget of b examples in each round. In framework (iii), as FlexMatch is computationally demanding, we only evaluate methods on their initial pool selection capabilities.

4.2 Main Results

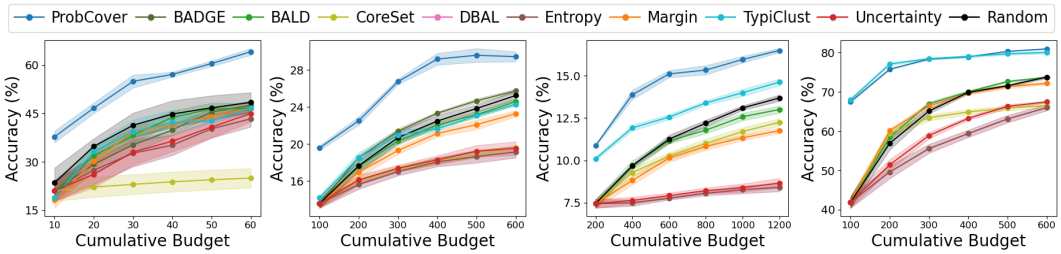
(i) Fully supervised framework. We evaluate different AL methods based on the performance of a deep neural network trained directly on the raw queried data. On each round, we query b samples (where b is equal to the number of and train a ResNet-18 on the accumulated queried set. We repeat this for 5 active learning rounds. In Fig. 3, we plot the mean accuracy of 5 repetitions. Additional results can be seen in App. E.



(a) CIFAR-10 (b) CIFAR-100 (c) Tiny-ImageNet (d) ImageNet-100

Figure 3: Framework (i), fully supervised: comparing *ProbCover* with baseline AL strategies on image classification tasks in the low budget regime. The budget b has an average of 1 sample per class (the initial sample may be imbalanced). The final average test accuracy in each iteration is reported, using 5 repetitions. The shaded area reflects the standard error across repetitions.

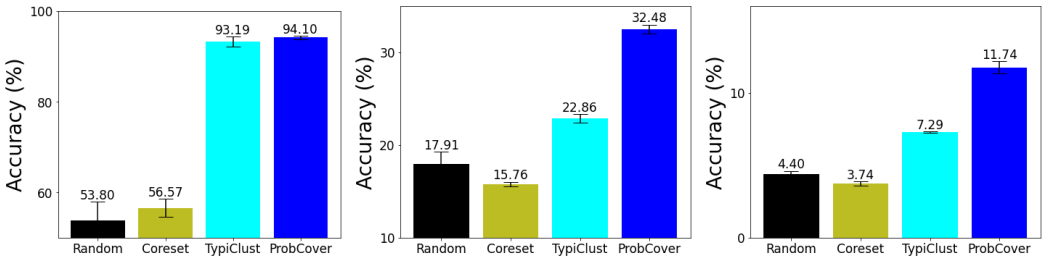
(ii) Semi-supervised by transfer learning. In this framework, we make use of pretrained self-supervised features, and measure classification performance using the 1-NN classifier. Accordingly, each point is classified by the label of its nearest neighbor (within the selected labeled set L) in the self-supervised features space. In low budgets, this framework outperforms the fully-supervised framework (i), but is not as effective as fully semi-supervised learning (iii), demonstrating that these findings are not specific to any given semi-supervised method. Similarly to Fig. 3, in Fig. 4, we plot the mean accuracy of 5 repetitions for the different tasks.



(a) CIFAR-10 (b) CIFAR-100 (c) Tiny-ImageNet (d) ImageNet-100

Figure 4: Same as caption of Fig. 3, for framework (ii) - semi-supervised by transfer learning.

(iii) Semi-supervised framework. We compare the performance of different AL strategies used prior to running FlexMatch, a state-of-the-art semi-supervised method. In Fig. 5 we show results with 3 repetitions of FlexMatch, using the labeled sets provided by different AL strategies and budget b equal to the number of classes. We see that our method outperforms random sampling and other AL baselines by a large margin. We note that in agreement with previous works [6, 18], AL strategies that are suited for high budgets do not improve the results of random sampling, while AL strategies that are suited for low budgets do.



(a) CIFAR-10 (b) CIFAR-100 (c) Tiny-ImageNet

Figure 5: Framework (iii) Semi-supervised: comparison of AL strategies in a semi-supervised task. Each bar shows the mean test accuracy after 3 repetitions of FlexMatch trained using b labeled examples, where b is equal to the number of classes in each task. Error bars denote the standard error.

4.3 Ablation Study

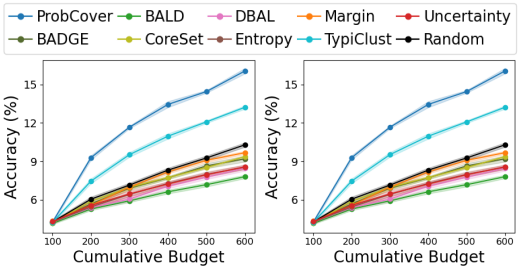
We report a set of ablation studies, evaluating the added value of each step of *ProbCover*.

Random initial selection When following the uncertainty sampling principle, as many AL methods do, a trained learner is needed. Any such method requires therefore a non-empty initial pool of labeled examples to train a rudimentary learner, from which uncertainty selection can be bootstrapped. In the set of methods evaluated here (see Section 4.1), only two - *ProbCover* and *TypiClust* - are not affected by this problem. This can be seen in Fig. 3, noting that only these two methods do better than random in the initial step. Is this the only reason they outperform other methods on low budgets?

To address this concern, we repeat the experiments reported in Fig. 3a-3b, using an initial random set of annotated examples across the board and by all methods. Results are reported in Fig. 6. When comparing Fig. 3a-3b and Fig. 6, we see that the advantage of *ProbCover* and *TypiClust* goes beyond the initial set selection, and can be seen even if this factor is eliminated.

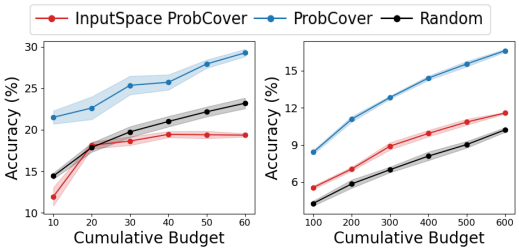
Input space distances As discussed in Section 3, our approach relies on the existence of a good embedding space, where distance is correlated with semantic similarity. We now verify this claim by repeating our basic fully-supervised experiments (Fig. 3) with one difference: *ProbCover* can only use the original pixel space representation to compute coverage. Results are show in Fig. 7. When comparing Fig. 3 with Fig. 7, a significant drop in performance is seen as expected, demonstrating the importance of the semantic embedding space.

The interaction between δ and budget size To understand the interaction between the hyper-parameter δ and budget b, we repeat our basic experiments (Fig. 3) with different choices of δ and b on CIFAR-10. For each pair (δ, b), we select an initial pool of b examples using *ProbCover* with δ balls, and report the difference in accuracy from the selection of b random points. Average results



(a) CIFAR-10 (b) CIFAR-100

Figure 6: Random Initial pool in the supervised framework, an average of 1 sample per class.



(a) CIFAR-10 (b) CIFAR-100

Figure 7: Comparison of *ProbCover* when applied to the raw data vs the embedding space.

across 3 repetitions are shown in Fig. 8 as a function of b. We see that as the budget b increases, smaller δ's are preferred.

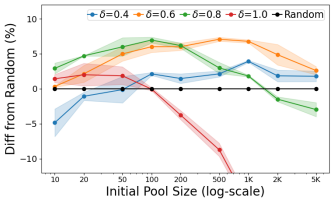
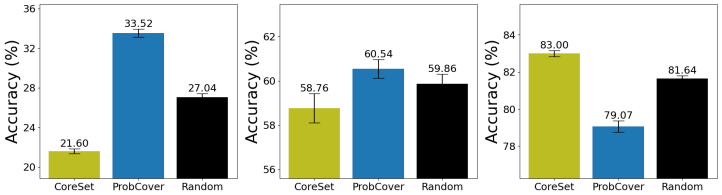


Figure 8: The accuracy difference between *ProbCover* when using different δ values, and the outcome of b random samples (average over 3 repetitions).



(a) Low budget (b) Mid budget (c) High Budget

Figure 9: Comparing the performance under the supervised framework of *ProbCover* and *Coreset* on different budget regimes. Low budget shows an initial pool selection of 100 samples. Mid/High budget start with 1K/5K samples and query additional 1K/5K samples (see text).

***Coreset* vs. *ProbCover*.** In Section 2.3 we argue that *ProbCover* is suitable for low budgets, while *Coreset* is suitable for high budgets. To verify this claim, we compare their performance under the following 3 setups while using the same embedding space, and report results on CIFAR-10:

- Low budget - Select an initial pool of 100 samples using the SimCLR representation.
- High budget - Train a model on 5K randomly selected examples. Then select an additional set of 5K examples using the learner's latent representation. This is the setup used in Sener and Savarese [36].
- Mid budget - Same as high budget, except the initial pool size and added budget are 1K.

Results are reported in Fig. 9. In the low budget, *ProbCover* outperforms *Coreset* as would be expected. In the mid budget, where the feature space of the learner is informative, only *ProbCover* achieves significant improvement over random selection. In the high budget, *Coreset* improves over random selection, while *ProbCover* does not.

5 Summary and Discussion

We study the problem of AL in the low-budget regime. We model the problem as *Max Probability Cover*, showing that under certain assumptions on the data distribution, which are likely to hold in self-supervised embedding spaces, it optimizes an upper-bound on the generalization error of a 1-NN classifier. We devise an AL strategy *ProbCover* that approximates the optimal solution. We empirically evaluate it in supervised and semi-supervised frameworks across different datasets, showing that *ProbCover* significantly outperforms other methods in the low-budget regime.

In future work we intend to investigate (i) possible avenues for improving the choice of δ by making use of already known labels, or inferring a score for δ through the topology of the resulting covering graph. (ii) extending the current formulation of *Max Probability Cover*, by making δ - the radius of the balls - dependent on the samples instead of being uniform. (iii) soft-coverage approaches, where the covering notion is not binary but some continuous measure, may allow us to do away with δ.

References

[1] Jordan T. Ash, Chicheng Zhang, Akshay Krishnamurthy, John Langford, and Alekh Agarwal. Deep batch active learning by diverse, uncertain gradient lower bounds. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020.

[2] Josh Attenberg and Foster Provost. Why label when you can search? alternatives to active learning for applying human resources to build classification models under extreme class imbalance. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 423–432, 2010.

[3] William H Beluch, Tim Genewein, Andreas Nürnberger, and Jan M Köhler. The power of ensembles for active learning in image classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9368–9377, 2018.

[4] Javad Zolfaghari Bengar, Joost van de Weijer, Bartlomiej Twardowski, and Bogdan Raducanu. Reducing label effort: Self-supervised meets active learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1631–1639, 2021.

[5] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. *arXiv preprint arXiv:2104.14294*, 2021.

[6] Yao-Chun Chan, Mingchen Li, and Samet Oymak. On the marginal benefit of active learning: Does self-supervision eat its cake? In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3455–3459. IEEE, 2021.

[7] Akshay L Chandra, Sai Vikas Desai, Chaitanya Devaguptapu, and Vineeth N Balasubramanian. On initial pools for deep active learning. In *NeurIPS 2020 Workshop on Pre-registration in Machine Learning*, pages 14–32. PMLR, 2021.

[8] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR, 2020.

[9] Ting Chen, Simon Kornblith, Kevin Swersky, Mohammad Norouzi, and Geoffrey E. Hinton. Big self-supervised models are strong semi-supervised learners. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020.

[10] Ekin D Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V Le. Randaugment: Practical automated data augmentation with a reduced search space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 702–703, 2020.

[11] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[12] Reza Zanjirani Farahani and Masoud Hekmatfar. *Facility location: concepts, models, algorithms and case studies*. Springer Science & Business Media, 2009.

[13] Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, pages 1050–1059. PMLR, 2016.

[14] Yarin Gal, Riashat Islam, and Zoubin Ghahramani. Deep bayesian active learning with image data. In *International Conference on Machine Learning*, pages 1183–1192. PMLR, 2017.

[15] Mingfei Gao, Zizhao Zhang, Guo Yu, Sercan Ö Arık, Larry S Davis, and Tomas Pfister. Consistency-based semi-supervised active learning: Towards minimizing labeling cost. In *European Conference on Computer Vision*, pages 510–526. Springer, 2020.

[16] Yonatan Geifman and Ran El-Yaniv. Deep active learning over the long tail. *arXiv preprint arXiv:1711.00941*, 2017.

[17] Daniel Gissin and Shai Shalev-Shwartz. Discriminative active learning. *arXiv preprint arXiv:1907.06347*, 2019.

[18] Guy Hacohen, Avihu Dekel, and Daphna Weinshall. Active learning on a budget: Opposite strategies suit high and low budgets. *arXiv preprint arXiv:2202.02794*, 2022.

[19] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[20] Tao He, Xiaoming Jin, Guiguang Ding, Lan Yi, and Chenggang Yan. Towards better uncertainty sampling: Active learning with multiple views for deep convolutional neural network. In *2019 IEEE International Conference on Multimedia and Expo (ICME)*, pages 1360–1365. IEEE, 2019.

[21] SeulGi Hong, Heonjin Ha, Junmo Kim, and Min-Kook Choi. Deep active learning with augmentation-based consistency estimation. *arXiv preprint arXiv:2011.02666*, 2020.

[22] Neil Houlsby, José Miguel Hernández-Lobato, and Zoubin Ghahramani. Cold-start active learning with robust ordinal matrix factorization. In *International Conference on Machine Learning*, pages 766–774. PMLR, 2014.

[23] Harry B Hunt III, Madhav V Marathe, Venkatesh Radhakrishnan, Shankar S Ravi, Daniel J Rosenkrantz, and Richard E Stearns. Nc-approximation schemes for np-and pspace-hard problems for geometric graphs. *Journal of algorithms*, 26(2):238–274, 1998.

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

[25] Andreas Kirsch, Joost Van Amersfoort, and Yarin Gal. Batchbald: Efficient and diverse batch acquisition for deep bayesian active learning. *Advances in neural information processing systems*, 32:7026–7037, 2019.

[26] Andreas Krause and Daniel Golovin. Submodular function maximization. *Tractability*, 3: 71–104, 2014.

[27] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. *Online*, 2009.

[28] Ya Le and Xuan Yang. Tiny imagenet visual recognition challenge. *CS 231N*, 7(7):3, 2015.

[29] Rafid Mahmood, Sanja Fidler, and Marc T Law. Low budget active learning via wasserstein distance: An integer programming approach. *arXiv preprint arXiv:2106.02968*, 2021.

[30] Dániel Marx. Efficient approximation schemes for geometric problems? In *European Symposium on Algorithms*, pages 448–459. Springer, 2005.

[31] Sudhanshu Mittal, Maxim Tatarchenko, Özgün Çiçek, and Thomas Brox. Parting with illusions about deep active learning. *arXiv preprint arXiv:1912.05361*, 2019.

[32] Prateek Munjal, N. Hayat, Munawar Hayat, J. Sourati, and S. Khan. Towards robust and reproducible active learning using neural networks. *ArXiv*, abs/2002.09564, 2020.

[33] George L Nemhauser, Laurence A Wolsey, and Marshall L Fisher. An analysis of approximations for maximizing submodular set functions—i. *Mathematical programming*, 14(1):265–294, 1978.

[34] Kossar Pourahmadi, Parsa Nooralinejad, and Hamed Pirsiavash. A simple baseline for low-budget active learning. *arXiv preprint arXiv:2110.12033*, 2021.

[35] Hiranmayi Ranganathan, Hemanth Venkateswara, Shayok Chakraborty, and Sethuraman Panchanathan. Deep active learning for image classification. In *2017 IEEE International Conference on Image Processing (ICIP)*, pages 3934–3938. IEEE, 2017.

[36] Ozan Sener and Silvio Savarese. Active learning for convolutional neural networks: A core-set approach. In *International Conference on Learning Representations*, 2018.

[37] Changjian Shui, Fan Zhou, Christian Gagné, and Boyu Wang. Deep active learning: Unified and principled method for query and training. In *International Conference on Artificial Intelligence and Statistics*, pages 1308–1318. PMLR, 2020.

[38] Oriane Siméoni, Mateusz Budnik, Yannis Avrithis, and Guillaume Gravier. Rethinking deep active learning: Using unlabeled data at model training. In *2020 25th International Conference on Pattern Recognition (ICPR)*, pages 1220–1227. IEEE, 2021.

[39] Samarth Sinha, Sayna Ebrahimi, and Trevor Darrell. Variational adversarial active learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5972–5981, 2019.

[40] Wouter Van Gansbeke, Simon Vandenhende, Stamatios Georgoulis, Marc Proesmans, and Luc Van Gool. Scan: Learning to classify images without labels. In *European Conference on Computer Vision*, pages 268–285. Springer, 2020.

[41] Zengmao Wang, Bo Du, Lefei Zhang, and Liangpei Zhang. A batch-mode active learning framework by querying discriminative and representative samples for hyperspectral image classification. *Neurocomputing*, 179:88–100, 2016.

[42] Kai Wei, Rishabh Iyer, and Jeff Bilmes. Submodularity in data subset selection and active learning. In *International Conference on Machine Learning*, pages 1954–1963. PMLR, 2015.

[43] Ziting Wen, Oscar Pizarro, and Stefan Williams. Active self-semi-supervised learning for few labeled samples fast training. *arXiv preprint arXiv:2203.04560*, 2022.

[44] Yi Yang, Zhigang Ma, Feiping Nie, Xiaojun Chang, and Alexander G Hauptmann. Multi-class active learning by uncertainty sampling with diversity maximization. *International Journal of Computer Vision*, 113(2):113–127, 2015.

[45] Changchang Yin, Buyue Qian, Shilei Cao, Xiaoyu Li, Jishang Wei, Qinghua Zheng, and Ian Davidson. Deep similarity-based batch mode active learning with exploration-exploitation. In *2017 IEEE International Conference on Data Mining (ICDM)*, pages 575–584. IEEE, 2017.

[46] Donggeun Yoo and In So Kweon. Learning loss for active learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 93–102, 2019.

[47] Michelle Yuan, Hsuan-Tien Lin, and Jordan L. Boyd-Graber. Cold-start active learning through self-supervised language modeling. In Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu, editors, *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 7935–7948. Association for Computational Linguistics, 2020.

[48] Bowen Zhang, Yidong Wang, Wenxin Hou, Hao Wu, Jindong Wang, Manabu Okumura, and Takahiro Shinozaki. Flexmatch: Boosting semi-supervised learning with curriculum pseudo labeling. *CoRR*, abs/2110.08263, 2021.

[49] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 586–595, 2018.

[50] Fedor Zhdanov. Diverse mini-batch active learning. *arXiv preprint arXiv:1901.05954*, 2019.

[51] Yu Zhu, Jinghao Lin, Shibi He, Beidou Wang, Ziyu Guan, Haifeng Liu, and Deng Cai. Addressing the item cold-start problem by attribute-driven active learning. *IEEE Trans. Knowl. Data Eng.*, 32(4):631–644, 2020. doi: 10.1109/TKDE.2019.2891530.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] Throughout the entire paper.
   (c) Did you discuss any potential negative societal impacts of your work? [No] Irrelevant for this work.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes] Section 2.1
   (b) Did you include complete proofs of all theoretical results? [Yes] Section 2, and App. B
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] We included the necessary information for reproducibility. Furthermore, the code will be published upon acceptance.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes]
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes]
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [No]
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes] 4.1
   (b) Did you mention the license of the assets? [Yes] All assets are publicly available.
   (c) Did you include any new assets either in the supplemental material or as a URL? [No]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [Yes] The data is publicly available.
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [No] Irrelevant.
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]