A Simple Baseline for Low-Budget Active Learning

Kossar Pourahmadi[1] Parsa Nooralinejad[1] Hamed Pirsiavash[2]

[1]University of Maryland, Baltimore County [2]University of California, Davis
{kossarp1, parsan1}@umbc.edu hpirsiav@ucdavis.edu

Abstract. Active learning focuses on choosing a subset of unlabeled data to be labeled. However, most such methods assume that a large subset of the data can be annotated. We are interested in low-budget active learning where only a small subset (*e.g.*, 0.2% of ImageNet) can be annotated. Instead of proposing a new query strategy to iteratively sample batches of unlabeled data given an initial pool, we learn rich features by an off-the-shelf self-supervised learning method only once, and then study the effectiveness of different sampling strategies given a low labeling budget on a variety of datasets including ImageNet. We show that although the state-of-the-art active learning methods work well given a large labeling budget, a simple K-means clustering algorithm can outperform them on low budgets. We believe this method can be used as a simple baseline for low-budget active learning on image classification. Code is available at: https://github.com/UCDvision/low-budget-al

Keywords: Low-Budget Active Learning, Self-Supervised Learning.

1 Introduction

We are interested in active learning with very low budget. Given a large set of unlabeled images and an oracle that can label a small set of images, we want to train an accurate image classification model by choosing a small set of images for the oracle to annotate. Active learning [14] has been studied for a long time. However, most active learning methods start with a large initial seed pool (usually randomly chosen images that are annotated) and also choose a large set of images to be annotated actively. For instance, [49,16,5] use more than 3% of ImageNet [44] as the initial pool, which contains more than $40,000$ images. In some applications, *e.g.*, medical image analysis, the labeling budget is much smaller, so the current active learning algorithms may not be a viable solution [32,55].

One may argue that our setting is similar to few-shot and semi-supervised learning where we assume a few examples of each class are labeled. However, we argue that those settings are not practical in many applications since sampling a subset of images for labeling, that are "uniformly" distributed across categories, itself needs a larger subset of data to be annotated first. In some real applications, even finding one example of an object to annotate may be challenging. For

(a) Self-supervised training (b) Feature extraction and clustering (c) Annotation and fine-tuning

Fig. 1: **Our simple baseline.** The goal is to train an accurate image classification model with a very small set of labeled images. **a)** A self-supervised learning model learns from the unlabeled data and provides the feature embeddings. **b)** K-means algorithm clusters the unlabeled data features and chooses data points nearest to the center of each cluster. **c)** These selected points are then annotated by an oracle. Finally, a linear classifier on the top of features learns from the annotated data and performs the image classification task.

instance, in a self-driving car application, one may want to build a detector for motorbikes with annotating a few motorbike examples only, but finding those few examples in the large dataset of many hours of driving video footage is a challenging task by itself.

As an example, Table 6a shows that some categories may not be even represented in a pool of $3,000$ randomly sampled ImageNet images. This makes standard few-shot or semi-supervised learning methods impractical. Hence, we believe active learning, when we want to have only a few annotations per category in average, is an important practical problem.

We believe given the recent progress in self-supervised learning (SSL) [1,10,12], active learning with very small budget (*e.g.*, annotating only $3,000$ images of ImageNet) should have become more tractable. Hence, we design a very simple baseline and show that it outperforms state-of-the-art active learning methods on small budgets. As shown in Figure 1, our baseline starts with training an off-the-shelf SSL method on the unlabeled data, running K-means clustering on the obtained features of unlabeled data, choosing examples closest to the center of each cluster, annotating those samples using the oracle, and finally training a linear classifier on the top of SSL features to perform the image classification task. Figure 2 shows how using K-means selection on SSL pre-trained features nicely covers categories of the unlabeled data.

Moreover, most active learning methods [46,20,49] annotate multiple batches of data sequentially which constrains the annotation work-flow since the oracle should wait for the model to be trained and select the next batch. We show that a variation of our method works well using a single annotation batch rather than multiple ones, reducing the cost of annotation work-flow.

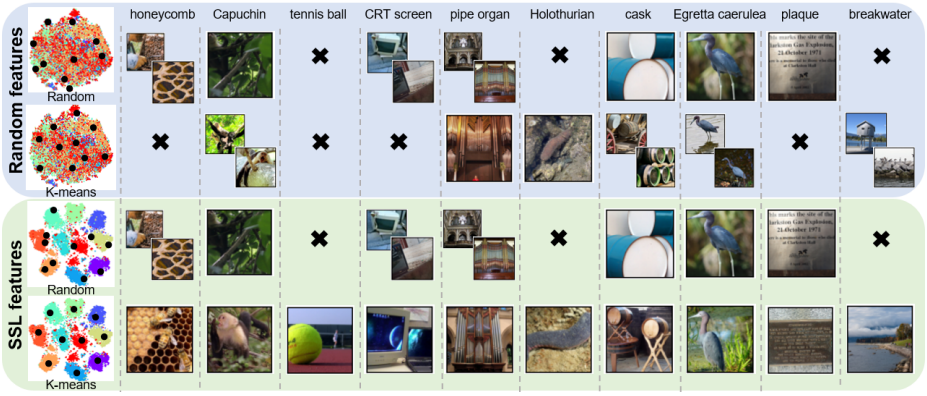

Fig. 2: **Visualization of category coverage of 10 samples on 10 randomly selected ImageNet categories.** We show t-SNE plots of two different feature initializations (random and SSL) and selection methods (Random and K-means) along with 10 selected images (black dots) for annotation. We have not done any cherry-picking or manual inspection for this visualization. Note that random selection uses the same seed, so Rows 1 and 3 have the same selected images. SSL initialization along with K-means results in 100% coverage while some categories are not represented in other settings. Table 6a shows that on full ImageNet, our method covers 99.9% of all categories with 7 samples on average per class.

2 Related Work

There is a broad array of learning approaches to give deep learning-based methods an ability of learning new concepts without requiring large annotated datasets. These approaches include few-shot learning, active learning, semi-supervised learning, and self-supervised learning.

Few-shot learning. Few-shot learning aims to recognize a set of classes that there are very few uniformly distributed training examples available for each of them (*e.g.* 1 or 5 examples per class) [3,21,50]. Although the amount of few-shot learning training data is much less than of large-scale deep learning methods, preparing a training set with as few as 1% uniform annotation still needs more than 1% of the unlabeled dataset to be annotated.

Active learning. Active learning has been widely studied to answer *how* to choose a fixed number of samples to gain the highest accuracy. Our work lies in pool-based approaches which can be categorized as uncertainty-based and distribution-based methods. Uncertainty-based methods try to find the most uncertain samples [53,52,7,2]. In Bayesian uncertainty-based approaches, Gaussian processes estimate uncertainty [28,43,19]. However, it is shown in [46] that these methods are not suitable for deep neural networks and large-scale datasets.

Distribution-based methods try to maximize the diversity of selected data over the entire dataset. [39] clusters the dataset at first and then uses a hier-

archical approach to avoid selecting repeated samples from the same cluster. Core-set [46] increases diversity in the selected batch of data by minimizing the Euclidean distance between instances of already labeled data and instances in the unlabeled pool. Despite working well on datasets with small number of classes, the performance of Core-set degrades when p-norms suffer from the curse of dimensionality in high-dimensions [18].

Some methods take advantage of both uncertainty and diversity [34,29,4]. VAAL [49] and DFAL [15] use adversarial learning to learn the representation of data points. It is shown in MAL [16] that although VAAL does not need the annotations to sample data, it may result in selecting multiple instances of the same class while there are already plenty of them in the labeled pool.

Many active learning methods require a large initial labeled pool and sampling budget [46,49,16,47]. This condition is difficult to meet in medical image analysis or other domains that the size of unlabeled training data is very large and it costs experts a lot of effort to label a large subset of them [26]. An approach to mitigate this problem is presented in [36]. However, it requires computing large distance matrices to solve Wasserstein distances for large datasets (*e.g.*, with 1M images) that presents a scalability bottleneck.

In this paper, we train an off-the-shelf SSL method on the unlabeled data to provide rich feature vectors only once and select a small proportion of them in a non-iterative approach to achieve strong performance on a variety of datasets, as well as large-scale datasets, without requiring an initial labeled pool.

Semi-supervised learning. Semi-supervised learning for image classification aims in making best use of limited labeled data while leveraging the unlabeled dataset [11,54,31]. Three most explored directions are consistency regularization [45,33], entropy minimization [23], and pseudo-labeling [6,51]. Despite achieving highly competitive performance to supervised methods, semi-supervised learning, similar to few-shot learning, assumes there are a few equal number of examples per category and needs more than the amount of training set to be annotated from the unlabeled dataset to ensure a uniform distribution. Therefore, we cannot compare active and semi-supervised methods directly. However, in Section 4.5, we will introduce two labeled set generation settings to show that active strategies can benefit the performance of semi-supervised models.

Self-supervised learning. SSL provides a strategy to train a neural network on the unlabeled data and creates rich features that can be fine-tuned for different downstream tasks using limited labeled data. SSL methods coarsely learn to solve a pretext task [22,42,41] or contrast between similar pairs of an image and its negative pairs [8,10,25]. CompRess [1] is one of the recent state-of-the-art SSL models that compresses a deep teacher network to a smaller student network such that for any query, the student ranks anchors similar to the teacher.

Prior active learning methods that take advantage of unsupervised learning tackle with time-consuming human-in-the-loop workflow of waiting for the new model to train on previous batches to annotate a new selected batch [48,20,56,16], or use large sampling budgets [9,38,17]. In contrast, we select a single batch of a few examples using the initial SSL pre-trained model.

3 Low-Budget Active Learning

3.1 Standard Active Learning

Assuming a fixed sampling budget N and a large pool of unlabeled data D_u, standard active learning algorithms train a model $\mathcal{M}$ using an initial labeled pool D_l^0. Then, at iteration t, they sample a subset D_l^t of images from D_u to be labeled manually. This subset expands the labeled data to $D_l = \cup_t D_l^t$. The number of iterations and size of each subset are chosen so that $|D_l|$ matches the available budget N. The final model $\mathcal{M}$ is trained on all labeled data.

3.2 Sampling with K-means Clustering Method

In this paper, we use two forms of K-means sampling including: *i*) *single-batch K-means* and *ii*) *multi-batch K-means*.

Single-batch K-means. Since we are interested in the low-budget setting, we eliminate the need for initial seed of labeled samples and in contrast to standard iterative active learning methods, we perform only one iteration of sampling. We simply apply K-means algorithm on feature outputs of an SSL pre-trained model only once and choose samples closest to the cluster centers. The single-batch method simplifies the annotation workflow as the annotators do not need to wait for the new model to train before annotating a new batch. However, one may need to know the whole number of samples that need annotation. We call single-batch K-means simply as *K-means* in the rest of the paper.

Multi-batch K-means. In contrast to single-batch version, multi-batch K-means is dependent on the previous sampling rounds. This method uses the difference of two consecutive budget sizes as the number of clusters and picks those nearest examples to centers that have not been labeled previously by the oracle. We call this variant of K-means sampling as *multi K-means*. Although we no longer waste previously labeled data and accumulate them to the new set of labeled examples, this process is iterative and we should wait for all previous rounds to finish. As a result, both scenarios (single-batch and multi-batch) have their own advantages and disadvantages and one can choose the method that fits the best to the application setting.

4 Experiments and Results

In this section, we evaluate different active learning baselines on image classification task.

Datasets. We use CIFAR-10/100 [30], ILSVRC-2012 ImageNet [44], and ImageNet-LT [35] to compare different sampling strategies. CIFAR-10/100 with 10/100 categories have $60,000$ images of size 32×32, where $50,000$ are training and $10,000$ are test images. ImageNet contains more than 1.2M images that are almost uniformly distributed over $1,000$ categories. ImageNet-LT is truncated from ImageNet and has the same $1,000$ categories, but the number of images per class ranges from 1280 to 5. We resize ImageNet/LT images into 224×224

pixels. The validation set for ImageNet/LT experiments is the same and contains $50,000$ samples. All datasets are augmented by horizontally flipping the images.

Baselines. We compare K-means and multi K-means with the following baselines: *i*) **Random** in which samples are selected randomly (uniformly) from the entire dataset; *ii*) **Max-Entropy** [53] which samples the points with the highest probability distribution entropy; *iii*) **Core-set** [46]; *iv*) **VAAL** [49]; and *v*) **Uniform** that selects equal number of random samples per class. Note that few-shot and semi-supervised learning methods use Uniform strategy to create training sets that may require more than the size of sets to be annotated.

Implementation details. In all experiments, unless specified, we use feature outputs of ResNet-18 that is pre-trained on unlabeled ImageNet using CompRess SSL method [1] for 130 epochs, which uses MoCo-v2 [12] as its teacher network. Note that this pre-trained feature extractor is used even for CIFAR experiments which means that technically, CIFAR experiments use the unlabeled data beyond CIFAR datasets. Max-Entropy sampling method [53] freezes the pre-trained backbone and trains an extra linear layer as the classifier on the top of that for 100 epochs. In all Max-Entropy experiments, we use Adam optimizer and lr=0.001 which is multiplied by 0.1 at epochs 50 and 75. The budget for iterative sampling methods is the difference between two consecutive budget sizes. For Random, Uniform, and K-means sampling, each budget size is equivalent to the amount of unlabeled data selection.

Evaluation metrics. Unless specified, all experiments are averaged over 3 runs with 3 constant random seeds. We follow four evaluation metric protocols:

i) **Linear classification.** We train a linear classifier on the top of the frozen backbone features (without back-propagation in the backbone weights) on the pool of labeled data for 100 epochs and report its top-1 accuracy on the test set. We apply the mean and standard deviation normalization at each dimension of backbone outputs to reduce the computational overhead of tuning the hyper-parameters per experiment. We use Adam optimizer and lr=0.01 that is multiplied by 0.1 at epochs 50 and 75. The batch size is 128 on ImageNet/LT. For CIFAR-10/100 experiments, initial pools contain only 10/100 examples, so we set batch size=4.

ii) **Nearest neighbor classification.** This uses cosine similarity as a distance metric to search for the most semantically similar neighbors of test set data from the pool of labeled images. When the pool of labeled data is small, this metric is faster than linear evaluation since nearest neighbor classification needs no hyper-parameter tuning. We use FAISS GPU library [27] for implementation.

iii) **Evaluation on fine-grained tasks.** We evaluate on Flowers-102 [40], DTD-47 [13], and Aircraft [37] as examples of fine-grained datasets. For feature embeddings, we use the same frozen backbone that is pre-trained on unlabeled ImageNet. In each dataset, we first choose a small subset of training data to be annotated, then we use the subset to train a linear classifier on top of the frozen backbone. This is similar to the transfer learning procedure in [10,24]. Full details on training sets are in the appendix Table 15.

iv) **Semi-supervised learning evaluation.** Both active and semi-supervised methods learn from limited labeled data. However, semi-supervised ones assume an equal number of examples per class are labeled, which is not practical in some applications. In contrast, active learning does not use any label information. Therefore, we cannot compare these two learning methods directly. Since selected samples by K-means cover a large proportion of all categories, one can train semi-supervised learning methods on the small annotated set selected by K-means. We follow the procedure of FixMatch [51], a state-of-the-art semi-supervised method, to evaluate K-means, Random and Uniform on CIFAR-10, Flowers-102 [40], and DTD-47 [13].

4.1 Performance Analysis on CIFAR-10/100

Tables 1 through 4 show the performance and category coverage of two K-means strategies compared to other active learning baselines on CIFAR-10/100. Max-Entropy, Core-set, and VAAL start from randomly selected 0.02% of CIFAR-10 and 0.2% of CIFAR-100. Multi K-means starts from K-means selected 0.02% of CIFAR-10 and 0.2% of CIFAR-100. We can observe in Table 1 that on CIFAR-10 both K-means and multi K-means outperform all sampling methods in nearest neighbor classification. In linear classification, despite not having equal number of examples per category, K-means performs better than Uniform at 0.02%, 0.04%, and 0.6%. Tables 3 and 4 compare different selection strategies on CIFAR-100 and show that in both evaluation metrics, K-means strategies outperform non-uniform methods and are on par with Uniform.

4.2 Performance Analysis on ImageNet

Tables 5 and 6a show the scalability of K-means sampling on the large-scale ImageNet dataset. 0.08% randomly and K-means selected ImageNet are used as the initial pools of iterative methods and multi K-means, respectively. In Table 5, K-means outperforms other baselines and has competitive results with Uniform sampling in both evaluation metrics.

Figure 6b illustrates the distribution of ImageNet categories over their number of occurrences in 0.2% of the unlabeled training data. They are $3,000$ total samples that equals to 3 per category in average. We prefer the distribution to have a peak at 3 and be zero otherwise which will happen with Uniform sampling. We see that K-means is closer to this peaky distribution compared to Random. Interestingly, Random sampling has almost 4 times more categories that are not represented in the samples (i.e., zero samples).

4.3 Performance Analysis on ImageNet-LT

It is important to use a sampling method that does not fail in real-world applications, in which data is not balanced. In Tables 7 and 8, we use the same frozen backbone that is pre-trained on unlabeled ImageNet as the feature extractor to compare different sampling strategies on ImageNet-LT. Initial pools of iterative methods and multi K-means are 0.8% randomly and K-means selected ImageNet-LT, respectively. Uniform sampling chooses equal number of

examples per class as long as it does not surpass the class size. Table 7 shows that K-means strategies are strong active learning methods in both evaluation metrics with no annotation information. Also, in the appendix A.2, we report top-1 linear and nearest neighbor classification results of different strategies on ImageNet-LT using an ImageNet-LT pre-trained backbone, as a realistic feature extractor to show that K-means is insensitive to category distribution of unlabeled training set.

Table 1: **Top-1 linear (LIN) and nearest neighbor (NN) classification results of different strategies on CIFAR-10.** Both K-means strategies outperform all methods in nearest neighbor classification. In linear classification, K-means outperforms Max-Entropy, Core-set, VAAL, and Random, and is on par with Uniform. Although in contrast to Uniform, K-means does not have equal number of examples per class, this strategy performs better than Uniform in 0.02%, 0.04%, and 0.6% budgets. In both evaluation benchmarks, K-means is consistently better than multi K-means.

Method	LIN	NN	Budgets 10 0.02%	20 0.04%	50 0.1%	70 0.14%	100 0.2%	200 0.4%	300 0.6%
Uniform	✓		20.4 ± .2	28.6 ± .0	**39.9** ± .1	**43.9** ± .1	**44.9** ± .1	**48.9** ± .0	51.9 ± .0
Random	✓		21.6 ± .3	28.7 ± .1	32.8 ± .1	36.2 ± .3	43.8 ± .1	48.1 ± .0	50.6 ± .1
Max-Entropy	✓		21.6 ± .3	26.4 ± .1	34.5 ± .2	38.6 ± .0	40.3 ± .1	44.6 ± .1	47.7 ± .0
Core-set	✓		21.6 ± .3	27.6 ± .2	34.9 ± .3	35.1 ± .1	38.8 ± .0	43.2 ± .0	46.5 ± .0
VAAL	✓		21.6 ± .3	26.4 ± .1	34.7 ± .0	38.8 ± .0	40.8 ± .1	44.6 ± .1	47.7 ± .0
Multi K-means	✓		**28.5** ± .2	28.6 ± .0	35.9 ± .1	39.2 ± .2	41.5 ± .0	48.1 ± .0	51.5 ± .1
K-means	✓		**28.5** ± .2	**33.3** ± .2	37.6 ± .1	43.7 ± .1	44.1 ± .0	48.8 ± .2	**52.1** ± .2
Uniform		✓	19.8 ± .3	25.7 ± .1	32.2 ± .1	32.9 ± .2	34.2 ± .0	36.4 ± .0	38.3 ± .1
Random		✓	23.7 ± .2	28.5 ± .1	29.4 ± .2	29.7 ± .1	32.5 ± .1	35.5 ± .0	37.5 ± .0
Max-Entropy		✓	23.7 ± .2	24.5 ± .1	28.1 ± .0	29.1 ± .0	27.8 ± .1	28.6 ± .1	30.1 ± .0
Core-set		✓	23.7 ± .2	25.3 ± .1	26.6 ± .1	27.7 ± .0	29.5 ± .0	30.4 ± .1	32.6 ± .0
VAAL		✓	23.7 ± .2	25.1 ± .1	29.1 ± .1	29.8 ± .0	30.1 ± .1	34.5 ± .1	35.8 ± .1
Multi K-means		✓	**29.5** ± .1	29.7 ± .0	34.2 ± .1	35.1 ± .0	36.1 ± .2	39.4 ± .1	40.2 ± .1
K-means		✓	**29.5** ± .1	**33.4** ± .2	**34.7** ± .1	**38.3** ± .0	**38.9** ± .1	**40.6** ± .1	**42.4** ± .0

Table 2: **CIFAR-10 category coverage of selected examples.** 0.02% K-means selected instances cover 80% of all CIFAR-10 categories.

Budgets	10 0.02%	20 0.04%	50 ≥ 0.1%
Uniform	**100** ± 0	**100** ± 0	100 ± 0
Random	56.7 ± 4.7	86.7 ± 4.7	100 ± 0
Max-Entropy	56.7 ± 4.7	68.0 ± 2.0	100 ± 0
Core-set	56.7 ± 4.7	86.7 ± 4.7	100 ± 0
VAAL	56.7 ± 4.7	71.0 ± 1.0	100 ± 0
Multi K-means	80.0 ± 0.0	**100** ± 0.0	100 ± 0
K-means	80.0 ± 0.0	90.0 ± 0.0	100 ± 0

Table 3: **CIFAR-100 category coverage of selected examples.** After Uniform sampling, both K-means methods have a better category coverage than iterative methods and are on par with Random.

Budgets	100 0.2%	300 0.6%	500 1%	1000 ≥ 2%
Uniform	**100** ± 0	**100** ± 0	**100** ± 0	100 ± 0
Random	60.9 ± 3.2	96.7 ± .9	100 ± 0	100 ± 0
Max-Entropy	60.9 ± 3.2	87.7 ± .9	97.0 ± 1.4	100 ± 0
Core-set	60.9 ± 3.2	89.0 ± .0	97.3 ± 1.2	100 ± 0
VAAL	60.9 ± 3.2	86.3 ± .9	95.7 ± 0.9	100 ± 0
Multi K-means	68.0 ± 0.0	93.0 ± .0	98.0 ± 0.0	100 ± 0
K-means	68.0 ± 0.0	95.0 ± .0	100 ± 0	100 ± 0

Table 4: **Top-1 linear (LIN) and nearest neighbor (NN) classification results of different strategies on CIFAR-100.** In both evaluation benchmarks, K-means strategies outperform Max-Entropy, Core-set, VAAL, and Random and are on par with Uniform sampling. Despite having equal number of examples per category, Uniform outperforms K-means in linear classification only on budgets larger than 4%. K-means and multi K-means are competitive on CIFAR-100.

Method	LIN	NN	Budgets								
			100	300	500	1000	2000	2500	4000	5000	7500
			0.2%	0.6%	1%	2%	4%	5%	8%	10%	15%
Uniform	✓		10.2 ± .2	18.7 ± .1	21.3 ± .1	27.2 ± .0	**29.9** ± .1	30.9 ± .1	**32.7** ± .0	**32.9** ± .0	**33.6** ± .0
Random	✓		10.4 ± .3	16.5 ± .1	20.7 ± .1	24.6 ± .4	29.3 ± .1	29.5 ± .5	30.8 ± .1	31.7 ± .1	32.8 ± .1
Max-Entropy	✓		10.4 ± .3	14.6 ± .1	17.2 ± .0	20.8 ± .1	21.9 ± .2	24.6 ± .1	26.1 ± .0	27.6 ± .0	28.8 ± .0
Core-set	✓		10.4 ± .3	15.1 ± .0	17.4 ± .1	22.2 ± .1	25.9 ± .0	26.9 ± .0	27.8 ± .1	28.6 ± .0	29.3 ± .1
VAAL	✓		10.4 ± .3	15.9 ± .1	19.1 ± .0	24.1 ± .0	28.4 ± .1	29.9 ± .1	30.9 ± .0	31.6 ± .1	33.1 ± .0
Multi K-means	✓		**13.4** ± .1	**20.0** ± .1	22.2 ± .0	26.1 ± .0	29.5 ± .0	30.2 ± .0	31.5 ± .0	31.5 ± .0	32.8 ± .0
K-means	✓		**13.4** ± .1	18.8 ± .1	**23.7** ± .1	**27.1** ± .0	29.4 ± .0	**31.1** ± .1	32.5 ± .0	32.8 ± .1	33.4 ± .0
Uniform		✓	10.1 ± .3	13.8 ± .1	15.3 ± .1	17.8 ± .0	20.2 ± .0	21.9 ± .0	24.1 ± .1	**25.8** ± .1	**30.0** ± .1
Random		✓	8.3 ± .1	12.8 ± .1	15.0 ± .2	16.9 ± .1	19.7 ± .1	20.8 ± .0	22.0 ± .1	23.2 ± .0	25.5 ± .0
Max-Entropy		✓	8.3 ± .1	10.1 ± .0	10.9 ± .0	12.1 ± .1	12.5 ± .1	12.7 ± .1	12.9 ± .0	13.1 ± .0	13.6 ± .1
Core-set		✓	8.3 ± .1	10.4 ± .1	10.9 ± .0	13.3 ± .0	16.4 ± .1	16.8 ± .0	18.2 ± .0	18.9 ± .0	20.7 ± .1
VAAL		✓	8.3 ± .1	12.1 ± .0	13.7 ± .1	16.7 ± .0	19.2 ± .1	20.4 ± .0	22.1 ± .1	23.1 ± .0	24.9 ± .0
Multi K-means		✓	**13.7** ± .2	**17.2** ± .0	18.0 ± .1	20.4 ± .1	22.9 ± .0	23.5 ± .0	24.2 ± .1	25.1 ± .0	27.2 ± .0
K-means		✓	**13.7** ± .2	17.1 ± .0	**19.4** ± .1	**21.1** ± .1	**23.1** ± .0	**23.6** ± .1	**24.5** ± .0	24.9 ± .1	26.1 ± .1

Table 5: **Top-1 linear (LIN) and nearest neighbor (NN) classification results of different strategies on ImageNet.** K-means outperforms all non-uniform sampling methods. Linear classification results of Uniform sampling that takes advantage of equal number of examples per class are better than K-means only at 5% and 10% by a small margin.

Method	LIN	NN	Budgets							
			1K	3K	7K	13K	26K	64K	128K	192K
			0.08%	0.2%	0.5%	1%	2%	5%	10%	15%
Uniform	✓		19.2 ± .3	31.9 ± .3	41.0 ± .3	46.0 ± .1	49.9 ± .0	**54.2** ± .1	**56.7** ± .1	57.9 ± .1
Random	✓		15.8 ± .0	28.0 ± .4	39.2 ± .3	45.1 ± .1	49.7 ± .1	54.0 ± .1	56.6 ± .1	57.9 ± .1
Max-Entropy	✓		15.8 ± .0	19.4 ± .0	25.6 ± .0	33.7 ± .0	41.3 ± .0	48.9 ± .1	51.9 ± .1	54.3 ± .1
Core-set	✓		15.8 ± .0	25.6 ± .1	33.3 ± .0	39.6 ± .0	45.7 ± .1	51.3 ± .0	54.9 ± .1	56.6 ± .1
VAAL	✓		15.8 ± .0	27.7 ± .1	34.9 ± .2	42.8 ± .1	49.2 ± .2	53.6 ± .1	56.0 ± .1	57.4 ± .0
Multi K-means	✓		**24.6** ± .0	34.1 ± .1	41.1 ± .0	45.3 ± .0	49.5 ± .0	53.9 ± .0	56.3 ± .0	57.5 ± .1
K-means	✓		**24.6** ± .0	**35.7** ± .0	**42.6** ± .1	**46.9** ± .1	**50.7** ± .0	54.0 ± .1	56.6 ± .0	**58.0** ± .1
Uniform		✓	29.5 ± .1	35.7 ± .1	38.9 ± .2	41.1 ± .1	43.2 ± .0	45.6 ± .0	47.6 ± .2	48.6 ± .0
Random		✓	22.8 ± .2	33.2 ± .7	38.4 ± .2	40.8 ± .0	42.2 ± .1	45.4 ± .1	47.3 ± .1	48.3 ± .1
Max-Entropy		✓	22.8 ± .2	24.5 ± .1	27.2 ± .1	30.3 ± .0	33.3 ± .1	36.2 ± .1	37.6 ± .0	38.6 ± .0
Core-set		✓	22.8 ± .2	30.7 ± .0	34.8 ± .1	37.5 ± .1	39.7 ± .1	42.0 ± .1	43.7 ± .2	44.6 ± .1
VAAL		✓	22.8 ± .2	32.8 ± .1	36.2 ± .0	39.7 ± .1	42.6 ± .0	45.3 ± .0	46.7 ± .1	47.9 ± .0
Multi K-means		✓	**31.6** ± .1	38.2 ± .0	41.4 ± .0	43.3 ± .0	45.2 ± .0	47.2 ± .0	**48.6** ± .0	**49.4** ± .0
K-means		✓	**31.6** ± .1	**39.9** ± .0	**42.7** ± .0	**44.0** ± .1	**45.5** ± .0	**46.8** ± .1	48.1 ± .1	48.8 ± .0

Table 6: **Category distribution of sampled ImageNet.** (a) K-means covers almost 98% of all classes with only $3,000$ examples while iterative methods require at least $7,000$ examples to reach this coverage; (b) Compared to Random selection, K-means has a sharper distribution around 3 which means selected images are distributed more uniformly across the categories.

(a) **ImageNet category coverage of selected examples.**

Budgets	1K 0.08%	3K 0.2%	7K 0.5%	13K $\geq$ 1%
Uniform	**100** $\pm$ 0	**100** $\pm$ 0	**100** $\pm$ 0	**100** $\pm$ 0
Random	62.9 $\pm$ .2	94.6 $\pm$ .4	100 $\pm$ 0	100 $\pm$ 0
Max-Entropy	62.9 $\pm$ .2	84.3 $\pm$ .5	94.8 $\pm$ .2	100 $\pm$ 0
Core-set	62.9 $\pm$ .2	87.9 $\pm$ .1	97.0 $\pm$ .5	100 $\pm$ 0
VAAL	62.9 $\pm$ .2	94.6 $\pm$ .1	98.1 $\pm$ .3	100 $\pm$ 0
Multi K-means	72.2 $\pm$ .1	97.0 $\pm$ .0	99.8 $\pm$ .0	100 $\pm$ 0
K-means	72.2 $\pm$ .1	97.8 $\pm$ .2	99.9 $\pm$ .0	100 $\pm$ 0

(b) **Category distribution of 3,000 samples of ImageNet (0.2%).**

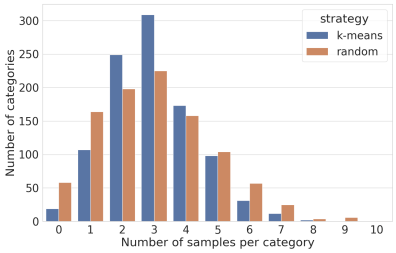

Table 7: **Top-1 linear (LIN) and nearest neighbor (NN) classification results of different strategies on ImageNet-LT.** We use the same frozen backbone pre-trained on unlabeled ImageNet as the feature extractor. With no labels information, K-means is a strong selection baseline in low budgets in both linear and nearest neighbor classification compared to prior works.

Method	**LIN**	**NN**	**Budgets** 1K 0.8%	3K 3%	5K 4%	7K 6%	9K 8%	12K 10%
Uniform	✓		**19.4** $\pm$ .1	**31.8** $\pm$ .1	**37.2** $\pm$ .2	**40.5** $\pm$ .0	**42.7** $\pm$ .1	**44.8** $\pm$ .0
Random	✓		13.0 $\pm$ .2	21.6 $\pm$ .1	26.7 $\pm$ .1	29.8 $\pm$ .0	32.1 $\pm$ .0	34.3 $\pm$ .0
Max-Entropy	✓		13.0 $\pm$ .2	16.3 $\pm$ .0	19.8 $\pm$ .1	22.9 $\pm$ .0	25.6 $\pm$ .1	28.6 $\pm$ .1
Core-set	✓		13.0 $\pm$ .2	22.7 $\pm$ .0	26.9 $\pm$ .0	30.3 $\pm$ .1	32.1 $\pm$ .0	34.9 $\pm$ .0
VAAL	✓		13.0 $\pm$ .2	21.8 $\pm$ .0	25.9 $\pm$ .0	28.7 $\pm$ .1	30.8 $\pm$ .0	33.4 $\pm$ .0
Multi K-means	✓		18.1 $\pm$ .0	24.7 $\pm$ .0	27.4 $\pm$ .1	29.4 $\pm$ .0	30.8 $\pm$ .0	33.5 $\pm$ .0
K-means	✓		18.1 $\pm$ .0	25.9 $\pm$ .1	29.4 $\pm$ .0	31.7 $\pm$ .2	33.6 $\pm$ .0	35.7 $\pm$ .1
Uniform		✓	**29.2** $\pm$ .0	**35.9** $\pm$ .1	**37.9** $\pm$ .1	**38.8** $\pm$ .1	**39.5** $\pm$ .0	**40.4** $\pm$ .0
Random		✓	19.3 $\pm$ .1	27.6 $\pm$ .0	31.3 $\pm$ .0	33.1 $\pm$ .0	34.4 $\pm$ .0	35.7 $\pm$ .0
Max-Entropy		✓	19.3 $\pm$ .1	22.3 $\pm$ .1	24.2 $\pm$ .1	25.9 $\pm$ .0	27.0 $\pm$ .0	28.8 $\pm$ .0
Core-set		✓	19.3 $\pm$ .1	29.3 $\pm$ .0	31.9 $\pm$ .0	33.7 $\pm$ .0	34.6 $\pm$ .0	35.9 $\pm$ .0
VAAL		✓	19.3 $\pm$ .1	26.9 $\pm$ .0	30.4 $\pm$ .0	32.2 $\pm$ .0	33.4 $\pm$ .1	34.7 $\pm$ .0
Multi K-means		✓	24.3 $\pm$ .0	29.6 $\pm$ .0	31.2 $\pm$ .0	32.2 $\pm$ .1	32.9 $\pm$ .1	34.9 $\pm$ .0
K-means		✓	24.3 $\pm$ .0	31.5 $\pm$ .0	34.3 $\pm$ .1	35.3 $\pm$ .0	36.3 $\pm$ .0	36.9 $\pm$ .0

Table 8: **ImageNet-LT category coverage of selected examples.** Uniform selects equal number of samples per class as long as that class contains examples.

Budgets	1K 0.8%	3K 3%	5K 4%	7K 6%	9K 8%	12K 10%
Uniform	**100** $\pm$ 0	**100** $\pm$ 0	**100** $\pm$ 0	**100** $\pm$ 0	**100** $\pm$ 0	**100** $\pm$ 0
Random	48.6 $\pm$ .3	76.1 $\pm$ 1.0	84.9 $\pm$ .5	89.6 $\pm$ .3	92.2 $\pm$ .4	94.8 $\pm$ .3
Max-Entropy	48.6 $\pm$ .3	69.6 $\pm$ .2	78.8 $\pm$ .1	84.2 $\pm$ .1	89.3 $\pm$ .0	92.0 $\pm$ .0
Core-set	48.6 $\pm$ .3	77.5 $\pm$ .4	86.7 $\pm$ .4	91.1 $\pm$ .8	93.5 $\pm$ .0	95.8 $\pm$ .3
VAAL	48.6 $\pm$ .3	73.9 $\pm$ .3	83.1 $\pm$ .1	87.6 $\pm$ .1	90.8 $\pm$ .1	93.6 $\pm$ .1
Multi K-means	51.8 $\pm$ .0	71.8 $\pm$ .0	79.7 $\pm$ .0	84.3 $\pm$ .0	88.3 $\pm$ .0	92.1 $\pm$ .0
K-means	51.8 $\pm$ .0	75.8 $\pm$ .0	86.8 $\pm$ .0	90.8 $\pm$ .0	92.2 $\pm$ .0	95.1 $\pm$ .0

4.4 Evaluation on Fine-grained Tasks

In Table 9, we analyze K-means clustering on datasets that are finer-grained compared to ImageNet. This table shows that the linear classification results of K-means on fine-grained datasets are consistently better than Random and are on par with Uniform.

Table 9: **Evaluation on fine-grained tasks.** The same frozen backbone that is pre-trained on unlabeled ImageNet is used to extract feature embeddings. For each dataset, a small subset of training data is annotated and is used to train a linear classifier on top of the frozen backbone. With no label information, K-means consistently outperforms Random and is on par with Uniform.

Methods	Flowers-102		DTD-47		Aircraft-100	
	1×102 10%	4×102 40%	1×47 2.5%	4×47 10%	1×100 2%	4×100 7.5%
Uniform	81.56 ±1.1	82.70 ±0.0	64.23 ±1.6	65.87 ±1.1	36.73 ±0.3	37.43 ±0.3
Random	80.56 ±1.6	82.06 ±1.3	53.86 ±0.6	61.06 ±0.7	36.10 ±1.7	37.36 ±0.1
K-means	82.20 ±1.0	82.76 ±1.2	64.30 ±0.0	65.96 ±0.7	37.20 ±0.0	38.67 ±0.2

4.5 Semi-Supervised Learning Evaluation

Table 10 shows semi-supervised evaluation results of FixMatch [51] when the labeled set is selected *with* label information, by Uniform and Uniform(K-means), or *without* it, by Random and K-means. In this table, Uniform(K-means) uses number of classes to cluster the dataset and selects equal number of examples per cluster. As shown in Table 10, K-means outperforms Random over all datasets with no label information. Taking the advantage of annotations, Uniform(K-means) performs better than Uniform in low budgets.

Table 10: **Semi-supervised evaluation with FixMatch [51].** The scores are top-1 accuracies (in %) of the model on the test set. In contrast to Random and K-means, Uniform and Uniform(K-means) take advantage of annotation information to sample a labeled set. With no labels, K-means performs consistently better than Random. Using labels, Uniform(K-means) outperforms Uniform in low budgets. The results are from 1 repetition of the experiments.

Methods	CIFAR-10			Flowers-102			DTD-47		
	1×10 0.02%	4×10 0.08%	10×10 0.2%	1×102 10%	3×102 30%	4×102 40%	1×47 2.5%	3×47 7.5%	4×47 10%
With Labels									
Uniform	54.79	88.76	**91.20**	15.22	34.33	**42.54**	8.88	18.40	**23.19**
Uniform(K-*means*)	**57.53**	**89.11**	89.40	**18.08**	**37.79**	41.60	**16.49**	**22.29**	22.39
Without Labels									
Random	38.69	60.21	86.73	13.81	31.42	42.51	8.30	16.86	21.65
K-means	**43.37**	**84.46**	**86.96**	**19.87**	**37.60**	**46.09**	**14.36**	**19.04**	**23.56**

5 Ablation Study

In this section, we perform ablation study on the initial labeled pool and the feature extraction backbone.

5.1 Effect of Initial Pool

Interestingly, Random sampling performs better than iterative active learning baselines at both accuracy and category coverage in some experiments. This may happen since a very small random initial labeled pool might not be representative of all categories and cannot be a strong starting point for iterative methods. To examine this hypothesis, we investigate how the strategy of sampling and size of the initial pool affect the performance of iterative methods.

Table 11: **Effect of a larger (2%) initial labeled set on ImageNet linear classification results.** VAAL and Core-set perform better than K-means and Random sampling using 10% of the unlabeled data. Results of Random and K-means are repeated from Table 5.

Budget	26K 2%	64K 5%	128K 10%
Max-Entropy	49.7 ± .1	51.8 ± .0	56.6 ± .0
Core-set	49.7 ± .1	52.5 ± .1	56.7 ± .0
VAAL	49.7 ± .1	53.8 ± .1	**56.8** ± .1
Random	49.7 ± .1	**54.0** ± .1	56.6 ± .1
K-means	**50.7** ± .0	**54.0** ± .1	56.6 ± .0

Table 12: **Effect of 0.08% K-means selected initial set of ImageNet on linear classification results.** Despite an improvement in the performance of iterative methods, K-means still outperforms them in low budgets. Results of K-means are repeated from Table 5.

Budget	1K 0.08%	3K 0.2%	7K 0.5%
Max-Entropy	24.6 ± .0	26.2 ± .1	30.0 ± .0
Core-set	24.6 ± .0	28.2 ± .0	33.6 ± .0
VAAL	24.6 ± .0	28.9 ± .1	35.0 ± .0
K-means	24.6 ± .0	**35.7** ± .0	**42.6** ± .1

Effect of size. We repeat the experiments with the same setting as Section 4.2 for a larger randomly sampled initial set of 2%. By comparing linear classification results of Tables 5 and 11, we find that all three iterative methods perform better using a larger initial pool. As a result, iterative methods are suitable options when large initial pools are available.

Effect of sampling strategy. We repeat the analyses in Section 4.2 with K-means selected instead of randomly selected 0.08% of ImageNet as the initial pool. The linear classification results on 0.2% and 0.5% of ImageNet are shown in Table 12. Comparing Tables 5 and 12 demonstrates that although the performance of iterative methods are improved with a richer initialization, there is still a great gap with K-means in low budgets.

5.2 Ablating Feature Extraction Backbone

We perform ablation on *i*) sampling and *ii*) classification backbone initialization and investigate their contributions to K-means performance on ImageNet in Table 13.

Table 13: **Fine-tuning and linear classification results of ablation study on sampling ImageNet using K-means.** SSL refers to using SSL pre-trained weights and Rand refers to using randomly initialized weights for backbones. Initializing both selection and classification backbones with SSL pre-trained or random features is an empirical higher and lower bound for our experiments, respectively. In the fine-tuning process, forgetting happens and causes the gap with the linear classification counterpart.

Selection backbone		**Classification backbone**		**Fine-tuning**	**Budgets**							
SSL	Rand	SSL	Rand		1K 0.08%	3K 0.2%	7K 0.5%	13K 1%	26K 2%	64K 5%	128K 10%	256K 20%
	✓		✓	No	0.38 ± .0	0.75 ± .0	1.07 ± .0	1.41 ± .1	2.05 ± .0	3.46 ± .0	4.26 ± .1	5.08 ± .1
	✓	✓		No	15.1 ± .0	26.5 ± .0	36.6 ± .1	43.8 ± .0	49.1 ± .0	54.0 ± .0	56.6 ± .0	58.7 ± .0
✓			✓	No	0.47 ± .0	0.87 ± .1	1.26 ± .1	1.79 ± .1	2.37 ± .1	3.35 ± .1	4.19 ± .1	5.01 ± .1
✓		✓		No	**24.6** ± .0	**35.7** ± .0	**42.6** ± .1	**46.9** ± .1	**50.7** ± .0	**54.0** ± .1	**56.6** ± .0	58.9 ± .0
	✓		✓	Yes	0.73 ± .0	1.50 ± .0	3.49 ± .1	8.43 ± .3	17.7 ± .3	35.3 ± .6	48.4 ± .2	58.3 ± .2
	✓	✓		Yes	12.2 ± .1	22.7 ± .2	32.2 ± .1	38.7 ± .1	44.9 ± .1	51.3 ± .0	55.4 ± .2	59.9 ± .1
✓			✓	Yes	1.22 ± .0	2.43 ± .0	5.63 ± .2	11.4 ± .2	19.9 ± .3	36.2 ± .3	48.9 ± .0	58.4 ± .1
✓		✓		Yes	19.5 ± .2	30.3 ± .2	37.3 ± .1	41.8 ± .2	46.5 ± .2	51.5 ± .2	55.6 ± .1	**60.0** ± .1

Ablating the sampling backbone. In Table 13, we find that by changing the selection backbone weights to random and keeping the evaluation setting the same, K-means performance drops in low budgets.

Ablating the classification backbone. Table 13 also presents the key role of SSL pre-trained classification backbone in achieving strong accuracy. When not fine-tuning the randomly initialized backbone, we train a linear layer on the top of frozen random features.

Effect of fine-tuning. In Table 13, we also report the fine-tuning results of all ablation variants. For randomly initialized backbones, we train both feature extractor and linear classifier using SGD optimizer for 100 epochs with the same learning rate of 0.1, which is multiplied by 0.1 in epochs 30, 60, and 90. For SSL pre-trained variants, we apply mean and standard deviation normalization at features before feeding to the linear layer. The optimizer is Adam and a lower learning rate is used for the backbone compared to the linear layer (10^{-4} vs. 10^{-2}). It is shown in Table 13 that back-propagating on a pre-trained model with a new objective causes the model to forget previously learned features and a drop in the performance happens.

5.3 Effect of the Network Architecture and Self-Supervised Pre-training Method

We change both selection and classification backbone architectures to ResNet-50 and pre-train them on ImageNet with MoCo-v2 [12] for 800 epochs. We report the experiments in Section 4.2 with the new setting in Table 14. This table shows that the superiority of K-means sampling to other active learning methods in low budgets is not sensitive to the choice of architecture or SSL pre-training method.

Table 14: **Effect of MoCo-v2 pre-trained R50 on top-1 linear (LIN) and nearest neighbor (NN) classification results of ImageNet.** The superiority of K-means is insensitive to the choice of network architecture and SSL pre-training method. The results are from 1 repetition of the experiments.

Method	LIN	NN	Budgets					
			1K 0.08%	3K 0.2%	7K 0.5%	13K 1%	26K 2%	64K 5%
Uniform	✓		30.0	40.2	46.8	50.8	54.9	59.5
Random	✓		23.4	35.4	45.2	49.9	54.5	59.5
Max-Entropy	✓		23.4	26.5	34.8	40.7	45.6	51.8
Core-set	✓		23.4	32.1	37.6	41.6	46.5	53.8
VAAL	✓		23.4	35.9	41.8	48.0	54.0	59.4
K-means	✓		**33.3**	**42.5**	**47.9**	**51.8**	**55.7**	**60.2**
Uniform		✓	30.8	37.5	40.8	43.1	45.2	48.1
Random		✓	23.7	33.9	40.2	42.9	45.2	48.0
Max-Entropy		✓	23.7	25.1	29.5	31.6	32.5	33.4
Core-set		✓	23.7	30.3	32.5	33.7	35.3	38.2
VAAL		✓	23.7	34.3	38.2	41.9	44.8	47.8
K-means		✓	**33.6**	**41.2**	**44.2**	**46.0**	**47.7**	**49.8**

6 Discussion

In general, we expect multi-batch active learning algorithms to perform better than single-batch ones since having machine learning and human in the loop reduces the redundancy in the annotated data. However, in our case, single-batch performs better than multi-batch. We hypothesize this happens since our single-batch method finds a large number of clusters that equals the total budget, which causes to represent very small clusters as well. One may improve the multi-batch method by encouraging diversity between iterations of the sampling.

We believe strong performance of K-means, especially in low budgets, happens since selected examples by K-means represent the categories even better than random examples in Uniform. As the budget size increases, all selection methods converge to the same performance results. Thus, with no annotation information, K-means clustering is a strong active learning baseline to achieve an accurate image classifier in very low budgets.

7 Conclusion

Most active learning benchmarks assume they have access to a large budget and large labeled seed pool. We believe there is practical need for active learning with smaller budgets. However, the problem is challenging as some categories in image classification may not be presented in the seed. We introduce a very simple baseline for this problem and show that it outperforms state-of-the-art active learning methods in low budgets. Our method leverages the recent progress in self-supervised learning along with simple K-means clustering for selecting the images that need to be annotated.

Acknowledgment: This material is based upon work partially supported by the United States Air Force under Contract No. FA8750-19-C-0098, funding from SAP SE, and also NSF grant numbers 1845216 and 1920079. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the United States Air Force, DARPA, or other funding agencies.

References

1. Abbasi Koohpayegani, S., Tejankar, A., Pirsiavash, H.: Compress: Self-supervised learning by compressing representations. In: Advances in Neural Information Processing Systems. vol. 33, pp. 12980–12992 (2020) 2, 4, 6
2. Aggarwal, U., Popescu, A., Hudelot, C.: Optimizing active learning for low annotation budgets. arXiv preprint arXiv:2201.07200 (2022) 3
3. Andrychowicz, M., Denil, M., Gómez, S., Hoffman, M.W., Pfau, D., Schaul, T., Shillingford, B., de Freitas, N.: Learning to learn by gradient descent by gradient descent. In: Advances in Neural Information Processing Systems. vol. 29 (2016) 3
4. Ash, J.T., Zhang, C., Krishnamurthy, A., Langford, J., Agarwal, A.: Deep batch active learning by diverse, uncertain gradient lower bounds. In: International Conference on Learning Representations (ICLR) (2020) 4
5. Beluch, W.H., Genewein, T., Nürnberger, A., Köhler, J.M.: The power of ensembles for active learning in image classification. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 9368–9377 (2018) 1
6. Berthelot, D., Carlini, N., Goodfellow, I., Papernot, N., Oliver, A., Raffel, C.A.: Mixmatch: A holistic approach to semi-supervised learning. In: Advances in Neural Information Processing Systems. vol. 32 (2019) 4
7. Brinker, K.: Incorporating diversity in active learning with support vector machines. In: Proceedings of the 20th international conference on machine learning (ICML-03). pp. 59–66 (2003) 3
8. Caron, M., Misra, I., Mairal, J., Goyal, P., Bojanowski, P., Joulin, A.: Unsupervised learning of visual features by contrasting cluster assignments. In: Proceedings of Advances in Neural Information Processing Systems (NeurIPS) (2020) 4
9. Chandra, A.L., Desai, S.V., Devaguptapu, C., Balasubramanian, V.N.: On initial pools for deep active learning. arXiv preprint arXiv:2011.14696 (2020) 4
10. Chen, T., Kornblith, S., Norouzi, M., Hinton, G.: A simple framework for contrastive learning of visual representations. In: International conference on machine learning. pp. 1597–1607. PMLR (2020) 2, 4, 6
11. Chen, T., Kornblith, S., Swersky, K., Norouzi, M., Hinton, G.E.: Big self-supervised models are strong semi-supervised learners. In: Advances in Neural Information Processing Systems. vol. 33, pp. 22243–22255 (2020) 4
12. Chen, X., Fan, H., Girshick, R., He, K.: Improved baselines with momentum contrastive learning. arXiv preprint arXiv:2003.04297 (2020) 2, 6, 13
13. Cimpoi, M., Maji, S., Kokkinos, I., Mohamed, S., Vedaldi, A.: Describing textures in the wild. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 3606–3613 (2014) 6, 7, 19
14. Cohn, D.A., Ghahramani, Z., Jordan, M.I.: Active learning with statistical models. Journal of artificial intelligence research **4**, 129–145 (1996) 1
15. Ducoffe, M., Precioso, F.: Adversarial active learning for deep networks: a margin based approach. arXiv preprint arXiv:1802.09841 (2018) 4

16. Ebrahimi, S., Gan, W., Salahi, K., Darrell, T.: Minimax active learning. arXiv preprint arXiv:2012.10467 (2020) 1, 4
17. Emam, Z.A.S., Chu, H.M., Chiang, P.Y., Czaja, W., Leapman, R., Goldblum, M., Goldstein, T.: Active learning at the imagenet scale. arXiv preprint arXiv:2111.12880 (2021) 4
18. François, D.: High-dimensional data analysis. From Optimal Metric to Feature Selection pp. 54–55 (2008) 4
19. Gal, Y., Ghahramani, Z.: Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In: international conference on machine learning. pp. 1050–1059. PMLR (2016) 3
20. Gao, M., Zhang, Z., Yu, G., Arık, S.Ö., Davis, L.S., Pfister, T.: Consistency-based semi-supervised active learning: Towards minimizing labeling cost. In: European Conference on Computer Vision. pp. 510–526. Springer (2020) 2, 4
21. Gidaris, S., Komodakis, N.: Dynamic few-shot visual learning without forgetting. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 4367–4375 (2018) 3
22. Gidaris, S., Singh, P., Komodakis, N.: Unsupervised representation learning by predicting image rotations. In: International Conference on Learning Representations (2018) 4
23. Grandvalet, Y., Bengio, Y.: Semi-supervised learning by entropy minimization. In: Advances in Neural Information Processing Systems. vol. 17 (2005) 4
24. Grill, J.B., Strub, F., Altché, F., Tallec, C., Richemond, P., Buchatskaya, E., Doersch, C., Avila Pires, B., Guo, Z., Gheshlaghi Azar, M., et al.: Bootstrap your own latent-a new approach to self-supervised learning. Advances in Neural Information Processing Systems **33**, 21271–21284 (2020) 6, 20
25. He, K., Fan, H., Wu, Y., Xie, S., Girshick, R.: Momentum contrast for unsupervised visual representation learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 9729–9738 (2020) 4
26. Hoi, S.C., Jin, R., Zhu, J., Lyu, M.R.: Batch mode active learning and its application to medical image classification. In: Proceedings of the 23rd international conference on Machine learning. pp. 417–424 (2006) 4
27. Johnson, J., Douze, M., Jégou, H.: Billion-scale similarity search with gpus. arXiv preprint arXiv:1702.08734 (2017) 6
28. Kapoor, A., Grauman, K., Urtasun, R., Darrell, T.: Active learning with gaussian processes for object categorization. In: 2007 IEEE 11th International Conference on Computer Vision. pp. 1–8. IEEE (2007) 3
29. Kirsch, A., van Amersfoort, J., Gal, Y.: Batchbald: Efficient and diverse batch acquisition for deep bayesian active learning. In: Advances in Neural Information Processing Systems. vol. 32 (2019) 4
30. Krizhevsky, A.: Learning multiple layers of features from tiny images. Tech. rep., Citeseer (2009) 5
31. Kuo, C.W., Ma, C.Y., Huang, J.B., Kira, Z.: Featmatch: Feature-based augmentation for semi-supervised learning. In: European Conference on Computer Vision. pp. 479–495 (2020) 4
32. Kuo, W., Häne, C., Yuh, E., Mukherjee, P., Malik, J.: Cost-sensitive active learning for intracranial hemorrhage detection. In: International Conference on Medical Image Computing and Computer-Assisted Intervention. pp. 715–723. Springer (2018) 1
33. Laine, S., Aila, T.: Temporal ensembling for semi-supervised learning. In: ICLR (Poster) (2017) 4

34. Li, X., Guo, Y.: Adaptive active learning for image classification. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2013) 4
35. Liu, Z., Miao, Z., Zhan, X., Wang, J., Gong, B., Yu, S.X.: Large-scale long-tailed recognition in an open world. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 2537–2546 (2019) 5
36. Mahmood, R., Fidler, S., Law, M.T.: Low budget active learning via wasserstein distance: An integer programming approach. arXiv preprint arXiv:2106.02968 (2021) 4
37. Maji, S., Rahtu, E., Kannala, J., Blaschko, M., Vedaldi, A.: Fine-grained visual classification of aircraft. arXiv preprint arXiv:1306.5151 (2013) 6, 19
38. Mottaghi, A., Yeung, S.: Adversarial representation active learning. arXiv preprint arXiv:1912.09720 (2019) 4
39. Nguyen, H.T., Smeulders, A.: Active learning using pre-clustering. In: Proceedings of the twenty-first international conference on Machine learning. p. 79 (2004) 3
40. Nilsback, M.E., Zisserman, A.: Automated flower classification over a large number of classes. In: 2008 Sixth Indian Conference on Computer Vision, Graphics & Image Processing. pp. 722–729. IEEE (2008) 6, 7, 19
41. Noroozi, M., Favaro, P.: Unsupervised learning of visual representations by solving jigsaw puzzles. In: European conference on computer vision. pp. 69–84. Springer (2016) 4
42. Noroozi, M., Pirsiavash, H., Favaro, P.: Representation learning by learning to count. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 5898–5906 (2017) 4
43. Roy, N., McCallum, A.: Toward optimal active learning through monte carlo estimation of error reduction. ICML, Williamstown pp. 441–448 (2001) 3
44. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., et al.: Imagenet large scale visual recognition challenge. International journal of computer vision **115**(3), 211–252 (2015) 1, 5
45. Sajjadi, M., Javanmardi, M., Tasdizen, T.: Regularization with stochastic transformations and perturbations for deep semi-supervised learning. In: Advances in Neural Information Processing Systems. vol. 29 (2016) 4
46. Sener, O., Savarese, S.: Active learning for convolutional neural networks: A core-set approach. In: International Conference on Learning Representations (2018) 2, 3, 4, 6
47. Shui, C., Zhou, F., Gagné, C., Wang, B.: Deep active learning: Unified and principled method for query and training. In: Chiappa, S., Calandra, R. (eds.) Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics. vol. 108, pp. 1308–1318. PMLR (26–28 Aug 2020) 4
48. Siméoni, O., Budnik, M., Avrithis, Y., Gravier, G.: Rethinking deep active learning: Using unlabeled data at model training. arXiv preprint arXiv:1911.08177 (2019) 4
49. Sinha, S., Ebrahimi, S., Darrell, T.: Variational adversarial active learning. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) (2019) 1, 2, 4, 6
50. Snell, J., Swersky, K., Zemel, R.: Prototypical networks for few-shot learning. In: Advances in Neural Information Processing Systems. vol. 30 (2017) 3
51. Sohn, K., Berthelot, D., Carlini, N., Zhang, Z., Zhang, H., Raffel, C.A., Cubuk, E.D., Kurakin, A., Li, C.L.: Fixmatch: Simplifying semi-supervised learning with consistency and confidence. In: Advances in Neural Information Processing Systems. vol. 33, pp. 596–608 (2020) 4, 7, 11

52. Tong, S., Koller, D.: Support vector machine active learning with applications to text classification. Journal of machine learning research **2**(Nov), 45–66 (2001) 3
53. Wang, D., Shang, Y.: A new active labeling method for deep learning. In: 2014 International Joint Conference on Neural Networks (IJCNN). pp. 112–119 (2014) 3, 6
54. Xie, Q., Dai, Z., Hovy, E., Luong, T., Le, Q.: Unsupervised data augmentation for consistency training. In: Advances in Neural Information Processing Systems. vol. 33, pp. 6256–6268 (2020) 4
55. Yang, L., Zhang, Y., Chen, J., Zhang, S., Chen, D.Z.: Suggestive annotation: A deep active learning framework for biomedical image segmentation. In: International conference on medical image computing and computer-assisted intervention. pp. 399–407. Springer (2017) 1
56. Zhang, B., Li, L., Yang, S., Wang, S., Zha, Z.J., Huang, Q.: State-relabeling adversarial active learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 8756–8765 (2020) 4

A Appendix

Here, we provide additional details about Sections 4.2 through 4.5. Table 15 shows the details of fine-grained datasets used in semi-supervised learning and fine-grained evaluation tasks described in Sec 4.4 and 4.5.

Table 15: **Fine-grained datasets details.** Training, val, and test split details of the fine-grained datasets used in semi-supervised learning and fine-grained evaluation tasks are listed. For DTD and Flowers, we use the provided val sets. For Aircraft, we sample 20% of samples per class.

Dataset	**Classes**	**Train size**	**Val size**	**Test size**	Accuracy measure
DTD [13]	47	1,880	1,880	1,880	Top-1
Aircraft [37]	100	5,367	1,300	3,333	Mean per-class
Flowers [40]	102	1,020	1,020	6,149	Mean per-class

A.1 Category Distribution of Sampled ImageNet

In Figure 3, we add the category distribution results of Core-set, VAAL, and Max-Entropy to Table 6b. This figure illustrates the distribution of ImageNet categories over their number of occurrences in 3,000 (0.2%) samples of the unlabeled training data. While we prefer the category distribution of selected data points to have a peak at 3 and be zero otherwise, which will happen with Uniform sampling, it is shown in Figure 3 that Core-set, VAAL, and Max-Entropy are far from this peaky distribution. We believe this happens since these iterative sampling methods start from a random initial pool that may not be representative of all categories. This incomplete category coverage also propagates to model knowledge and selected batches in next sampling rounds. However, K-means sampled examples have sharper category distribution around 3 which means selected images are distributed more uniformly across the categories.

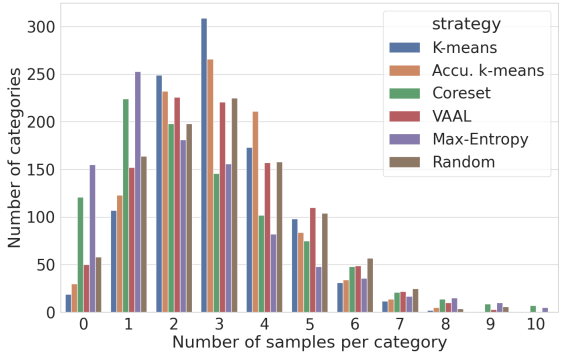

Fig. 3: **Category distribution of 3,000 samples of ImageNet (0.2%).** Non-peaky category distribution of sampled examples at 3 for iterative active learning methods that start from random initial pool happens since a random initial pool may not cover all categories and does not provide a strong starting point.

A.2 Performance Analysis on ImageNet-LT

In Section 4.3, we use the same frozen backbone pre-trained on unlabeled ImageNet as the feature extractor and show that K-means strategies are strong baselines on ImageNet-LT dataset in low budgets. To assure that K-means sampling performance on ImageNet-LT is insensitive to category distribution of the unlabeled training set used for pre-training the backbone, we repeat the experiment in Section 4.3 with a ResNet-50 that is pre-trained on ImageNet-LT using BYOL [24] and report the results in Table 16. As shown in this table, despite a drop in linear and nearest neighbor classification results of all methods, K-means still performs better than non-uniform sampling methods in low budgets in both evaluation metrics without taking advantage of annotations.

We hypothesize that pre-training a self-supervised model on an imbalance training set causes a drop in overall accuracies. Such problem is actively being studied in the community and is out of the scope of this paper.

Table 16: **Top-1 linear (LIN) and nearest neighbor (NN) classification results of different sampling methods on ImageNet-LT.** We use a ResNet-50 that is pre-trained on ImageNet-LT using BYOL [24] to assure the insensitivity of K-means to category distribution of unlabeled training set. With no label information, K-means performs better than non-uniform sampling methods in both linear and nearest neighbor classification.

Method	LIN	NN	Budgets					
			1K 0.8%	3K 3%	5K 4%	7K 6%	9K 8%	12K 10%
Uniform	✓		5.34 ± .1	**10.6** ± .1	**13.5** ± .2	**15.6** ± .0	**17.6** ± .1	**19.5** ± .0
Random	✓		5.04 ± .2	8.60 ± .1	11.4 ± .1	12.8 ± .0	14.1 ± .0	15.6 ± .0
Max-Entropy	✓		5.04 ± .2	7.41 ± .0	9.29 ± .1	10.7 ± .0	11.9 ± .1	13.6 ± .1
Core-set	✓		5.04 ± .2	7.77 ± .0	9.66 ± .0	10.8 ± .1	12.0 ± .0	13.7 ± .0
VAAL	✓		5.04 ± .2	8.58 ± .0	10.6 ± .0	12.2 ± .1	13.4 ± .0	14.9 ± .0
Multi k-means	✓		**6.01** ± .0	9.69 ± .0	11.4 ± .1	12.7 ± .0	13.9 ± .0	15.4 ± .0
K-means	✓		**6.01** ± .0	9.60 ± .1	11.7 ± .0	13.1 ± .2	14.4 ± .0	15.9 ± .1
Uniform		✓	4.81 ± .0	7.03 ± .1	8.21 ± .1	9.02 ± .1	**9.95** ± .0	**10.8** ± .0
Random		✓	4.42 ± .1	6.40 ± .0	7.64 ± .0	8.22 ± .0	8.85 ± .0	9.55 ± .0
Max-Entropy		✓	4.42 ± .1	4.45 ± .1	4.56 ± .1	4.76 ± .0	5.02 ± .0	5.36 ± .0
Core-set		✓	4.42 ± .1	5.58 ± .0	6.34 ± .0	7.03 ± .0	7.65 ± .0	8.40 ± .0
VAAL		✓	4.42 ± .1	6.33 ± .0	7.26 ± .0	7.95 ± .0	8.51 ± .1	9.12 ± .0
Multi k-means		✓	**5.48** ± .0	7.58 ± .0	8.50 ± .0	**9.20** ± .1	9.74 ± .1	10.4 ± .0
K-means		✓	**5.48** ± .0	**7.61** ± .0	**8.64** ± .1	9.05 ± .0	9.74 ± .0	10.2 ± .0