LabelBench: A Comprehensive Framework for Benchmarking Label-Efficient Learning

Jifan Zhang [*1] **Yifang Chen** [*2] **Gregory Canal** [1] **Stephen Mussmann** [2] **Yinglun Zhu** [1] **Simon S. Du** [2]
Kevin Jamieson [2] **Robert D. Nowak** [1]

Abstract

Labeled data are critical to modern machine learning applications, but obtaining labels can be expensive. To mitigate this cost, machine learning methods, such as transfer learning, semi-supervised learning and active learning, aim to be *label-efficient*: achieving high predictive performance from relatively few labeled examples. While obtaining the best label-efficiency in practice often requires combinations of these techniques, existing benchmark and evaluation frameworks do not capture a concerted combination of all such techniques. This paper addresses this deficiency by introducing LabelBench, a new computationally-efficient framework for joint evaluation of multiple label-efficient learning techniques. As an application of LabelBench, we introduce a novel benchmark of state-of-the-art active learning methods in combination with semi-supervised learning for fine-tuning pretrained vision transformers. Our benchmark demonstrates better label-efficiencies than previously reported in active learning. LabelBench's modular codebase will be open-sourced for the broader community to contribute label-efficient learning methods and benchmarks.

1. Introduction

Recently, large pretrained models have provided practitioners strong starting points in developing machine-learning-powered applications (Radford et al., 2021; Yu et al., 2022; Kirillov et al., 2023). While zero-shot and few-shot predictions can provide solid baselines, linear probing (which freezes the model and trains a layer on top) and fine-tuning based on human annotation yield significantly better performance (Radford et al., 2021; Yu et al., 2022). Label-efficient learning, the objective of which is to achieve high predictive performance with fewer labels, has received much attention lately due to the high annotation cost of labeling large-scale datasets.

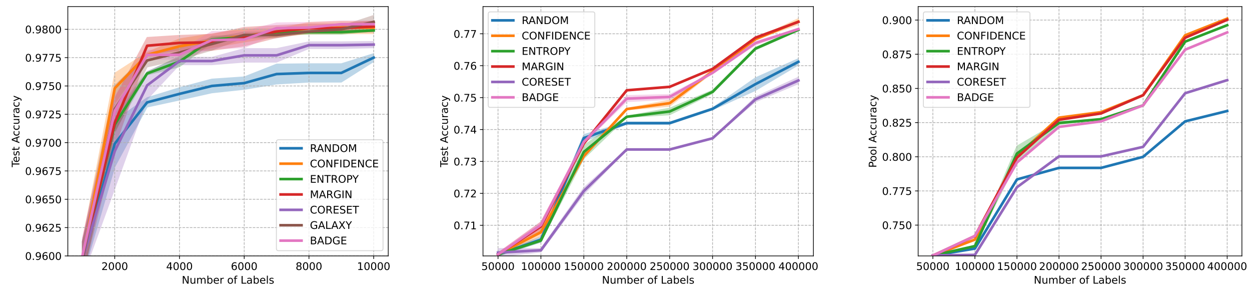
Transfer learning, semi-supervised learning (SSL) and active learning (AL) all study different aspects of label-efficient learning. Modern transfer learning leverages large general-purpose models pretrained on web-scale data and fine-tunes the model to fit application-specific examples. Semi-supervised learning utilizes a large set of unlabeled examples to estimate the underlying data distribution and more efficiently learn a good model. Active learning incrementally and adaptively annotates only those examples deemed to be informative by the model. To date, however, no existing literature has studied the above methods under a single unified framework for fine-tuning large pretrained models.

In this paper, we present LabelBench, a comprehensive benchmarking framework for *label-efficient* learning. Additionally, our framework tackles computational efficiency problems that arise when scaling these techniques to large neural network architectures. Specifically, incorporating active learning involves periodically re-training the model based on the latest labeled examples. While repeatedly training small convolutional neural networks is practically feasible (Sener & Savarese, 2017; Ash et al., 2019; 2021; Beck et al., 2021; Zhan et al., 2022; Lüth et al., 2023), retraining large-scale models is extremely compute intensive, which could be computationally prohibitive for conducting large scale experiments on active learning. Inspired by selection-via-proxy (Coleman et al., 2019), we propose lightweight retraining schemes (based on freezing all but the last layer of large pretrained models) for the purpose of data selection and labeling, but evaluate final model performance with a single end-to-end fine-tuning. This technique yields a ten-fold reduction in training cost, but reaps all the label-efficiency gains of using active learning.

To showcase the power of our framework, we conduct ex-

[*]Equal contribution [1]University of Wisconsin, Madison, WI, USA [2]University of Washington, Seattle, WA, USA. Correspondence to: Jifan Zhang <jifan@cs.wisc.edu>, Yifang Chen <yifangc@cs.washington.edu>.

Proceedings of the 40^{th} International Conference on Machine Learning DMLR Workshop, Honolulu, Hawaii, USA. Copyright 2023 by the author(s).



(a) Generalization Accuracy on CIFAR-10 (b) Generalization Accuracy on ImageNet (c) Pool Accuracy on ImageNet

Figure 1. Performance of active learning + FlexMatch re-training + CLIP ViT-B32 when given different annotation budgets. Generalization accuracy refers to the model's Top-1 test accuracy. Pool accuracy measures the labeling accuracy on the pool of examples to be labeled (see Section 4.2 for more details). Each curve of CIFAR-10 is averaged over 4 trials and each curve of ImageNet is averaged over two trials. The confidence intervals are based on standard error. The AL gains over passive presented here are significantly larger than typical gains observed in previous AL work where SSL and pretrained models are not considered.

periments that benchmark multiple deep active learning algorithms in combination with semi-supervised learning and large pretrained models; compared to existing active learning literature, our experiments yield state-of-the-art label-efficiency. To highlight some of our results, we observe a more than four-fold reduction (75% savings) in annotation cost over random sampling on CIFAR-10 (Figure 1(a)), a dataset known to be particularly challenging for active learning [1]. This improvement is further demonstrated in our experiments on ImageNet (Figure 1(b, c)). Under any fixed annotation budget, our experiments suggest active learning algorithms can consistently boost test accuracy by more than 1.2% and pool accuracy (accuracy of predictions on the pool of unlabeled training data defined in Section 4.2) by more than 5%. Compared to the previous best results in this setting (Emam et al., 2021), our results yield at least 10% higher test accuracy. Overall, LabelBench provides a light-weight experiment framework for researchers to test their algorithms on under more realistic and large-scale scenarios.

2. Related Work

Large pretrained models have demonstrated a wide range of emergent generalization abilities on downstream language and vision tasks. Most of these models are trained on web-scale data with supervised (Kolesnikov et al., 2019; Dosovitskiy et al., 2020; Zhai et al., 2021) or self-supervised techniques (Radford et al., 2021; Jia et al., 2021; Yuan et al., 2021; Singh et al., 2021; Yao et al., 2021; Wang et al., 2022a; Yu et al., 2022). While these models are powerful by themselves, adapting them to applications often requires transfer learning by fine-tuning on human annotated examples. Below we survey existing literature on label-efficient learning with an emphasis on the interplay among large pretrained models, semi-supervised learning and active learning.

2.1. Semi-supervised Training

While in traditional supervised learning the model is only trained on the set of *labeled* examples, in semi-supervised learning (SSL) the model is additionally trained on the remaining *unlabeled* examples in the pool, with the intention of taking the underlying sample distribution into account for more label-efficient training. Intuitively, SSL leverages the assumption that examples lying “nearby” to one another should belong to the same class, and therefore during training the model is encouraged to produce the same model output for these examples (for an overview of SSL we refer the interested reader to (Zhu, 2005; van Engelen & Hoos, 2020; Ouali et al., 2020)). Broadly speaking, modern SSL methods implement this principle using a combination of *Consistency Regularization* — where model outputs of neighboring examples are regularized to be similar — and *Pseudo Labeling* — where unlabeled examples that the model is confident on are assigned artificial labels to supplement supervised training (Sohn et al., 2020; Berthelot et al., 2020). In our pipeline we implement one such SSL method called FlexMatch (Zhang et al., 2021), due to its simplicity and effectiveness.

Semi-supervised Training of Large Pretrained Models. The application of SSL to fine-tuning large pretrained models is a nascent area of research. (Cai et al., 2022) pioneered the application of SSL methods to large-scale vision transformers by using a multi-stage pipeline of pretraining followed by supervised fine-tuning and finally semi-supervised fine-tuning. (Lagunas et al., 2023) apply this pipeline to a fine-grained classification e-commerce task and demonstrate

[1] As reported in seminal papers and common benchmarks such as Ash et al. (2019) (Figure 16), Ash et al. (2021) (Figure 10), Beck et al. (2021) (Figure 1) and Lüth et al. (2023) (Figure 6-8), they see less than two-fold reductions in annotation cost.

improved performance compared to standard supervised training. SSL training on transformer architectures has also been successfully applied to video action recognition (Xing et al., 2023). USB (Wang et al., 2022b) is a benchmark that includes SSL evaluations on large pretrained models such as ViT; however, it does not incorporate AL into its pipeline, as we do here.

2.2. Active Learning

Suppose we have a large pool of unlabeled examples and a limited labeling budget. We might study how to choose an informative subset for label annotation so that a learner yields strong performance. While experimental design (Pukelsheim, 2006) studies the setting where the subset is chosen before any annotations are observed, pool-based active learning (Settles, 2009) examines iterative adaptive annotation: labels from previously annotated examples can be used to determine which examples to choose for annotation in the next iteration. Active learning algorithms are generally designed to maximize one or both of the intuitive concepts of *uncertainty* and *diversity*. Uncertainty, measured in a variety of ways (Settles, 2009), refers to the uncertainty of a trained model for the label of a given point (Lewis, 1995; Scheffer et al., 2001), while diversity refers to selecting points with different features (Sener & Savarese, 2017). Many algorithms maximize a combination of these two concepts (Ash et al., 2019; 2021; Citovsky et al., 2021; Zhang et al., 2022).

Active Learning for Fine-Tuning Large Pretrained Models. Recent literature in deep active learning has started to utilize large pretrained models for large-scale datasets. Coleman et al. (2022) proposes a computational efficient method to annotate billion-scale datasets by actively labeling examples only in the neighborhood of labeled examples in the SimCLR (Chen et al., 2020) embedding space. Tamkin et al. (2022) studies the emergent property of uncertainty sampling when using large pretrained models. LabelBench serves as a more comprehensive large-scale benchmark for these studies, where we combine SSL training in our framework. We further take into account the expensive cost of fine-tuning large pretrained models at every iteration of active data collection.

In addition, numerous papers have utilized self-supervised or unsupervised learning methods to initilaize their models (Siméoni et al., 2021; Chan et al., 2021; Wen et al., 2022; Lüth et al., 2023) on the unlabeled datasets. However, their methods do not utilize existing large pretrained models.

Active Learning with Semi-supervised Training. Since AL and SSL seek to maximize model performance using only a minimal budget of labeled points, it is natural to combine both techniques to maximize label efficiency. This practice dates back to (Zhu et al., 2003), which labels examples that minimize expected classification error in a Gaussian Field SSL model. In the context of deep learning, (Lüth et al., 2023; Chan et al., 2021; Mittal et al., 2019; Siméoni et al., 2021) benchmark various AL methods in SSL settings. (Huang et al., 2021) develops a hybrid AL/SSL approach for computer vision tasks, and (Gao et al., 2020) develops a consistency-based AL selection strategy that is naturally compatible with SSL methods. (Borsos et al., 2021) approaches AL in the context of SSL as a problem of dataset summarization, and demonstrates improved performance on keyword detection tasks. (Hacohen et al., 2022; Yehuda et al., 2022) both use FlexMatch as a baseline SSL method in their AL experiments, further corroborating our choice to implement FlexMatch in our own pipeline.

3. Label Efficient Fine-tuning Framework

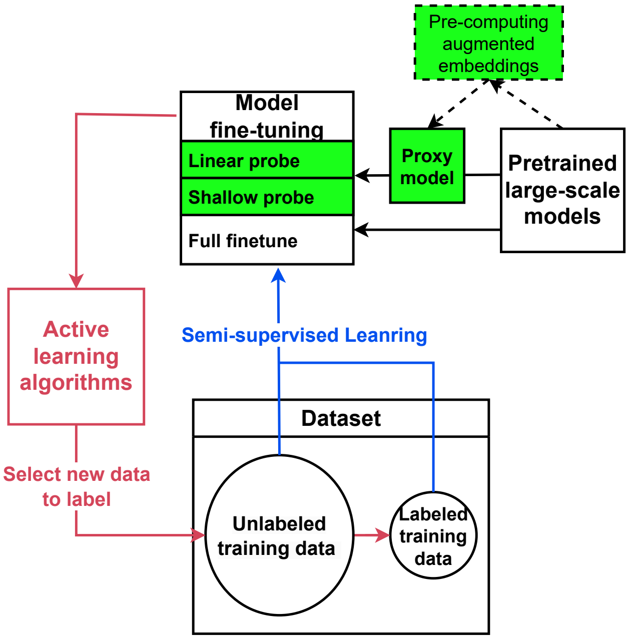


Figure 2. A modular framework consisted of pre-trained models, SSL trainer and AL strategies.

We propose a framework for label-efficient learning consisting of three widely-adopted components in modern deep learning: initialization from a large pretrained model, data annotation, and fine-tuning on downstream tasks. Our framework bears much resemblance to traditional active learning, but also takes advantage of large pretrained models and semi-supervised learning in obtaining the optimal label-efficiency. As shown in Figure 2, our framework starts with a large pretrained model as initialization. Data annotation follows a closed-loop procedure, where one starts with a pool of unlabeled examples in the beginning and iteratively gathers more human annotations. At any iteration, given a partially labeled pool we utilize semi-supervised training to obtain the best performing model. Informed by this trained model, an active learning strategy selects unlabeled examples it deems the most informative. The selected examples are sent to human annotators for labeling. At the end of the iteration, the newly annotated labels are recorded into the dataset.

The greatest challenge in implementing this framework comes from incorporating large-scale model training while meeting a limited computational budget. Unlike classical deep AL literature (Sener & Savarese, 2017; Ash et al., 2019; 2021) that utilizes smaller neural network architectures (e.g., ResNet-18), computational cost of fine-tuning large pretrained models at every iteration of the data collection loop is significant. To address this challenge, we propose a *selection-via-proxy* approach (Section 3.1), along with additional code optimization to improve the computational and memory efficiencies for large-scale datasets. In addition, our codebase is modular, allowing contributors to easily work on isolated components of the framework (Section 3.2).

3.1. Selection via Proxy

During each iteration of data collection, there are three potential strategies in fine-tuning the large pre-trained model: fine-tuning the model end-to-end, training only a linear probe, and training a nonlinear probe with shallow neural network. In the latter two strategies, the learner freezes the pretrained image encoder and attaches it with the less computationally intensive model (i.e. linear classifier, shallow network). In traditional active learning, the same model is utilized for both informing the selection of examples and acts as a deployable predictive model after label budget has been exhausted. Consequently, end-to-end fine-tuning leads to expensive retraining costs, while probing methods, although computationally efficient, performs much worse at test time due to the limitation on model capacity.

To better trade-off between retraining/inference cost and the final model performance, we present the *selection-via-proxy* approach, which is inspired by (Coleman et al., 2019). In the referenced work, a less computationally intensive proxy is created by carefully scaling down the original model architecture and training for fewer epochs. In our framework, we exploit a more straightforward approach by employing the linear probe and shallow network models as potential proxies. During every iteration of the data annotation loop, the learner only retrains the proxy model, which informs the selection of unlabeled examples to be annotated. After collecting a sufficient amount of labeled examples or reaching the labeling budget limits, the learner then switches to end-to-end fine-tuning at the last batch to further boost the performance of the final model. As a result, selection-by-proxy significantly reduces the cost of back-propagation.

	End-to-end Fine-Tune		Shallow Network (proxy)	
Training Stage	GPU Hours	AWS Dollars	GPU Hours	AWS Dollars
Precomputation	0	$0	5	$15
Retraining	1900	$5700	57	$180
Final Model	100	$300	100	$300
Total	2000	$6300	162	$495

Table 1. Estimated cost of neural network training for ImageNet experiments when collecting 600,000 labels with 20 iterations (batches of 30,000 labels per iteration). Here we display the total cost of running 12 trials with CLIP ViT-B32 and FlexMatch semi-supervised training (Zhang et al., 2021). All AWS dollars are based on on-demand rates of EC2 P3 instances.

```
# Add a new dataset.
@register_dataset(”my_dataset”,
                  MULTI_CLASS)
def get_dataset(...):
    ...

# Add a new SSL Algorithm.
Class MyTrainer(PyTorchSemiTrainer):
    def train_step(img, aug_img, ...):
        ...
```

Figure 3. Our modular codebase allows one to work solely in one directory without a thorough knowledge of the entire codebase. Implementing a new dataset or semi-supervised learning trainer is as easy as implementing a single function.

We further diminish the forward inference cost by precomputing and saving embeddings of each dataset in advance. To account for random image augmentations during training, we precompute five sets of embeddings on randomly augmented images using different random seeds. Our dataloader loops through these sets of embeddings over different epochs. As shown in Table 1, we highlight the reduction in experimentation cost on the ImageNet dataset. In particular, selection-via-proxy reduces the GPU time and training induced cost by more than ten-fold.

3.2. Codebase

Our codebase consists of five components: datasets, model, training strategy (for suprevised and semi-supervised training), active learning strategy and metrics. We would like to highlight the following advantages of our implementation:

- **Modularity.** As shown in Figure 3, adding any new instance, such as a new dataset or training strategy, simply involves implementing a new function. This allows future contributors to solely focus on any isolated component without a thorough understanding of the entire repository.

- **Self-report mechanism.** We include configuration files of all experiment setups. In addition, we keep track of all experiment results in the results directory for fair comparisons. Researchers are encouraged to self-report their research findings by submitting pull requests to our repository. This significantly reduces the unnecessary overhead of replicating existing experiments for the research community.
- **Significant speed-up of existing AL implementation.** Running some AL algorithms can be time-prohibitive when scaled to large datasets with large numbers of classes. In our implementation, we speed up popular active learning algorithms such as BADGE (Ash et al., 2019) by orders of magnitude in comparison to existing implementations (see details in Appendix C).

4. Benchmarking Active Learning Algorithms

As a demonstration of the utility of our framework, we conduct experiments in comparing popular deep active learning strategies in combination with large pretrained models and semi-supervised training. Our results presented in Section 4.4 shows better label efficiencies than state-of-the-art deep AL literature. Moreover, we discuss the accuracy gap by using selection-via-proxy under different settings.

4.1. Experiment Setup

Here we detail our benchmark's specific choices of AL strategies, large pretrained models, and semi-supervised training methods. It is important to note that settings beyond the ones discussed here can also be easily integrated into our general framework and codebase. We leave more detailed discussions on potential future directions to Section 5.

Our benchmark studies the following annotation procedure:

1. **Initial large pre-trained model.** We use pretrained CLIP (Radford et al., 2021) and CoCa (Yu et al., 2022) with the ViT-B32 architecture as image encoders. For end-to-end fine-tuning, we attach the image encoder with a dataset-dependent zero-shot head. On the other hand, proxy models are initialized randomly. Throughout our experiments, shallow networks have a single hidden layer with the same dimension as the embeddings.
2. **Initial batch of labels.** We collect the first batch of labels by choosing examples uniformly at random.
3. **Adaptive annotation loop.** We iterate over the following steps to annotate batches of examples.

   Model training. At the beginning of each iteration, the dataset is partially labeled. We use the semi-supervised training technique FlexMatch (Zhang et al., 2021) to fine-tune the vision transformer or train the proxy model from scratch. FlexMatch minimizes the sum of *supervised training loss* on labeled examples and *unsupervised losses* on unlabeled examples, including the loss of pseudo labeled examples and the regularization term capturing the input distribution properties.

   Data selection. Given the trained model, we use a data selection strategy to select unlabeled examples for annotation. We benchmark against prevalent active learning algorithms such as confidence sampling (Lewis, 1995), margin sampling (Scheffer et al., 2001), entropy sampling (Settles, 2009), BADGE (Ash et al., 2019) and GALAXY (Zhang et al., 2022) (see Section 2 and Appendix A for details). These algorithms make decisions based on the model's properties and its prediction on the pool of unlabeled examples (e.g. the confidence/entropy score, the gradient of the linear head).

   Annotate. Based on the strategy's selection, we reveal the ground-truth labels and update the dataset.
4. **Final Model.** After the annotation budget is exhausted, we fine-tune the pretrained CLIP or CoCa model end-to-end regardless if we are using proxy model for selection. Similar to the above, we use FlexMatch to fine-tune on the collected labeled examples as well as the remaining unlabeled examples.

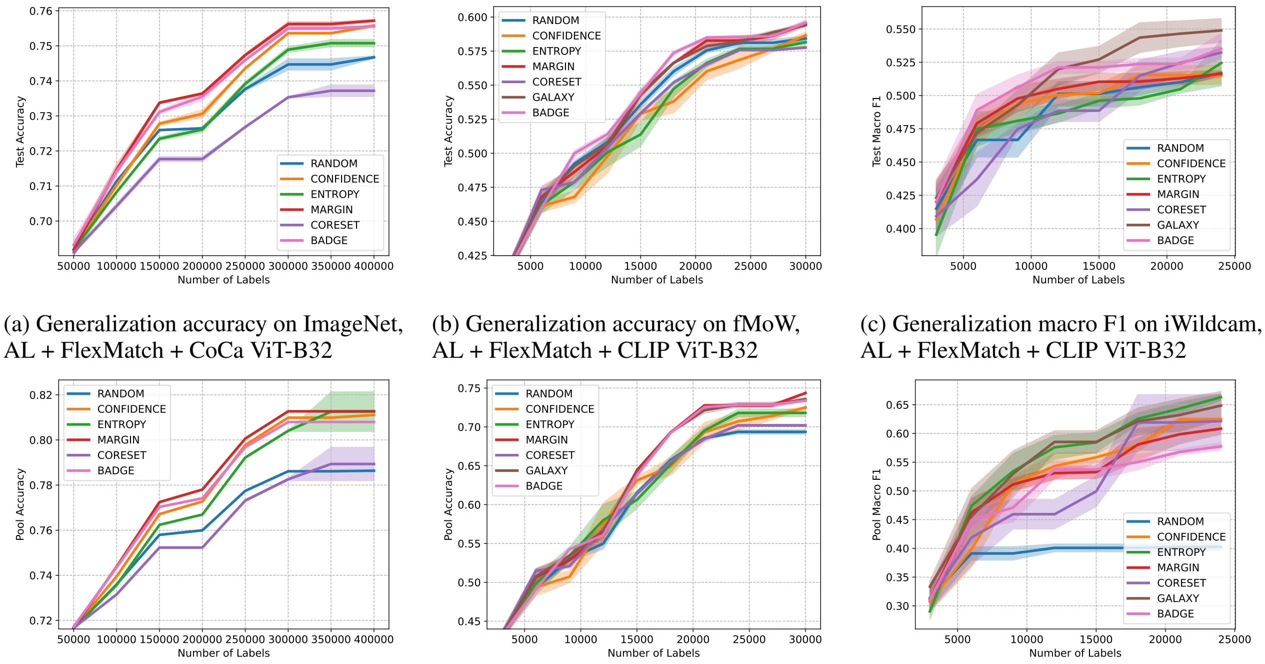
Appendix B details our hyper-parameter tuning procedure.

4.2. Performance Metrics

We report results on the following two tasks of label-efficient learning.

- **Label-efficient generalization** aims to learn accurate models that generalize beyond examples in the pool while spending limited budget on oracle annotation, such as human labeling. We refer to the models' performances on test data as *generalization performance*. In this paper, we report performances on in-distribution test data (drawn from the same distribution as the pool). As will be mentioned in Section 5, one may further develop benchmarks on label-efficient learning under distribution shifts.
- **Label-efficient annotation** aims to annotate all examples in the pool with limited budget. When the dataset is partially labeled by human, a model trained based on existing annotations can serve as a pseudo annotation tool that labels the rest of the unlabeled examples. We refer to the percentage of labels (both human annotated and pseudo labels) that agree with ground-truth labels as the *pool performance*. Examples of label-efficient annotation applications include product cataloging, categorizing existing userbases, etc.

To quantify performance, we use the standard accuracy for (near) balanced datasets, and balanced accuracy and macro F1 score for imbalanced datasets. Balanced accuracy and macro F1 score are measured as unweighted averages of per-class recall accuracies and per-class F1 scores, respectively.



(a) Generalization accuracy on ImageNet, AL + FlexMatch + CoCa ViT-B32

(b) Generalization accuracy on fMoW, AL + FlexMatch + CLIP ViT-B32

(c) Generalization macro F1 on iWildcam, AL + FlexMatch + CLIP ViT-B32

(d) Pool accuracy on ImageNet, AL + FlexMatch + CoCa ViT-B32

(e) Pool accuracy on fMoW, AL + FlexMatch + CLIP ViT-B32

(f) Pool macro F1 on iWildcam, AL + FlexMatch + CLIP ViT-B32

Figure 4. Performances of different data selection strategies on ImageNet, fMoW and iWildcam. The ImageNet results differ from Figure 1 by using a different pretrained model, CoCa ViT-B32. Each result of fMoW and iWildcam is averaged over four trials and each results of ImageNet is over two trials due to the limitation of computing resources. The confidence intervals are based on standard error.

4.3. Datasets

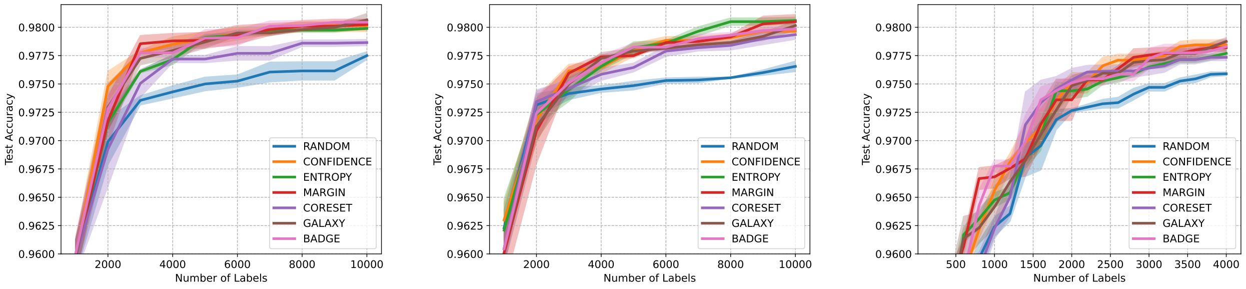
We first test on CIFAR-10 and ImageNet, both of which are standard datasets used in previous AL and SSL papers. To further evaluate LabelBench on more realistic datasets, we also test on iWildCam (Beery et al., 2021) and fMoW (Christie et al., 2018), as parts of WILDS benchmark (Koh et al., 2021). To the best of our knowledge, only a handful of existing studies, such as (Tamkin et al., 2022; Mussmann et al., 2022; Bartlett et al., 2022), have evaluated label-efficient algorithms on these datasets, albeit under different experimental setups. This benchmark is originally intended to represent distribution shifts faced in the wild (i.e., OOD test sets), here we limit our evaluation to in-domain (ID) test set performance as an initial exploratory step. Using these datasets provides several advantages:: 1) Both of them are highly imbalanced. 2) Fine-tuning pre-trained large-scale models on them is more challenging than on imageNet (e.g., ID test acc on fMoW is 73.3% (Wortsman et al., 2022) when fine-tuning ViT-L14 end-to-end). 3) Unlike imagenet whose examples are gathered by querying search engines and passing candidate images through a validation step on Amazon Mechanical Turk, iWildCam and fMoW gather labels directly from human annotators, which aligns more closely with our aim to enhance label efficiency.

4.4. Results and Discussion

In this section we present a summary of performance evaluations on various combinations of models and AL strategies.

End-to-end Fine-tuning First, we summarize our results when end-to-end fine-tuning the large pretrained model at every iteration of the data collection loop. When comparing the results of AL strategies to random sampling, we consistently see label efficiency gains across all datasets (Figures 1 and 4). Such label efficiency gain is especially significant on pool performances, with active learning strategies saving up to 50% of the annotation budget for ImageNet (Figure 4(d)). Notably, these gains are not confined to CLIP models. As shown in Figures 4(a,d), we observe consistent gains in accuracies also with pretrained CoCa model. In general, when comparing performance of different AL strategies on (near) balanced datasets (ImageNet, CIFAR-10 and fMoW), margin sampling surprisingly performs among the top in terms of both generalization and pool accuracy. On imbalanced dataset like iWildcam, GALAXY demonstrates a clear advantage in terms of generalization and pool macro F1 scores. These findings underscore the importance of further evaluating AL strategies on more realistic dataset.

Lastly, comparing to existing literature of AL + SSL (Lüth et al., 2023; Chan et al., 2021; Mittal et al., 2019; Siméoni



(a) Selection and evaluation with end-to-end fine-tuning, batch size of 1000

(b) Selection with shallow network, evaluation on fine-tuning, batch size of 1000

(c) Selection with shallow network, evaluation on fine-tuning, batch size of 200

Figure 5. Generalization performance on CIFAR-10 when using different proxy models for data selection. Each result is averaged over four trials and the confidence intervals are based on standard error.

	Test Accuracy		Pool Accuracy	
	Fine-tune	Shallow Network	Fine-tune	Shallow Network
Confidence	97.84 ± .07	97.85 ± .05	99.92 ± .02	99.67 ± .02
Entropy	97.89 ± .08	97.87 ± .14	**99.93 ± .01**	99.65 ± .02
Margin	**97.97 ± .12**	97.88 ± .17	**99.93 ± .01**	**99.68 ± .01**
Coreset	97.79 ± .06	97.81 ± .19	99.48 ± .02	98.94 ± .03
GALAXY	97.94 ± .20	**97.98 ± .12**	99.90 ± .01	99.66 ± .02
BADGE	97.95 ± .08	97.84 ± .10	**99.93 ± .01**	99.61 ± .02
Random	97.59 ± .22	97.59 ± .22	98.18 ± .05	98.18 ± .05
Best Overall	97.97 ± .12	97.98 ± .10	99.93 ± .01	99.68 ± .01

Table 2. Selection-via-proxy results of CIFAR-10 using CLIP ViT-B32. The results are evaluated with 10,000 labels. Confidence intervals are standard errors based on four trials.

et al., 2021) and AL + large pretrained models (Tamkin et al., 2022), our experiment yields the largest percentage of annotation cost savings to reach the same level of accuracy as random sampling. This reinforce the importance of studying the combination of active learning, semi-supervised learning and pretrained foundation models under an unified framework.

Selection-via-proxy. We also study the effectiveness and drawbacks of selection-via-proxy where we only retrain shallow neural networks (proxy models) for data selection. We compare it against *selection with fine-tuning*, where one fine-tunes the entire model during the data collection process. Note that despite using different models for data selection, our evaluation results of both strategies are reported based on fine-tuning pretrained models end-to-end on the selected examples. As shown in Table 2, 3, 4, selection-via-proxy performs similarly to selection with end-to-end fine-tuned models in terms of *test accuracy*. On the other hand, we found that selection-via-proxy is less effective than selection with fine-tuning in terms of *pool accuracy* - there is an approximately 3% reduction in performance in fMoW and ImageNet experiments.

To further investigate the label-efficiency tradeoff of the two methods, in Figures 5(a,b), we plot their performances respectively after collecting every batch of labels. The gap between selection-via-proxy and selection with fine-tuning diminishes quickly with more iterations of data selection. As shown in Figure 5(c), we can further close the gap in lower-budget settings by collecting more rounds of annotations with smaller batches. Indeed, to achieve 97.75% accuracy (random sampling's accuracy with 10,000 labels), selection-via-proxy only requires 2750 labels (with batch size of 200), comparable to selection with fine-tuning's label-efficiency in Figure 5(a). We note that smaller batches are only computationally feasible for selection-via-proxy, as one can only end-to-end fine-tune a small number of times under a limited budget.

5. Call for Contribution and Future Work

We call on the broader community to further develop different components of LabelBench: below we provide suggested contributions for each directory of framework. Our codebase is modular, so one can easily start working on a single component without a thorough understanding of the

	Test Accuracy		Pool Accuracy	
	Fine-tune	Shallow Network	Fine-tune	Shallow Network
Confidence	58.66 ± .49	57.82 ± .37	72.47 ± .32	70.91 ± .41
Entropy	58.14 ± .75	57.75 ± .35	71.02 ± 1.40	70.87 ± .27
Margin	59.51 ± .37	58.80 ± .06	**74.36 ± .19**	**71.63 ± .19**
Coreset	57.71 ± .26	57.35 ± .07	68.43 ± .42	66.50 ± .40
GALAXY	59.41 ± .22	58.91 ± .19	73.56 ± .43	71.32 ± .76
BADGE	**59.59 ± .47**	**59.25 ± .27**	73.30 ± .16	70.92 ± .05
Random	58.40 ± .34	58.40 ± .34	68.46 ± .13	68.46 ± .13
Best Overall	59.59 ± .47	59.25 ± .27	74.36 ± .19	71.63 ± .19

Table 3. Selection-via-proxy results of fMoW using CLIP ViT-B32. The results are evaluated with 30,000 labels. Confidence intervals are standard errors based on four trials.

	Test Accuracy		Pool Accuracy	
	Fine-tune	Shallow Network	Fine-tune	Shallow Network
Confidence	**77.38 ± .13**	76.96 ± .12	**90.11 ± .01**	**88.93 ± .01**
Entropy	77.12 ± .04	76.63 ± .11	89.62 ± .01	88.33 ± .02
Margin	77.37 ± .04	**77.15 ± .01**	90.02 ± .03	88.75 ± .03
Coreset	75.54 ± .15	75.33 ± .17	85.60 ± .01	84.84 ± .03
BADGE	77.15 ± .02	76.83 ± .04	89.10 ± .04	87.64 ± .02
Random	76.12 ± .14	76.12 ± .14	83.35 ± .01	83.35 ± .01
Best Overall	77.38 ± .13	77.15 ± .01	90.11 ± .01	88.93 ± .01

Table 4. Selection-via-proxy results of ImageNet using CLIP ViT-B32. The results are evaluated with 400,000 labels. Confidence intervals are standard errors based on two trials.

entire codebase.

Trainer. Our experiments demonstrate the potential label savings provided by combining active learning with FlexMatch. To expand upon these results, it would be valuable to develop a benchmark of additional semi-supervised methods, evaluated in combination with active learning and large pretrained models. To do so, one could instantiate specific SSL trainer classes that inherit our template SSL trainer.

Active Learning Strategy. As exhibited by our results, combining active learning with SSL and large pretrained models results in highly accurate and label-efficient models that demonstrate clear gains over passive learning. Therefore, we call on the active learning community to evaluate their active selection algorithms under our more comprehensive and up-to-date benchmark. Additionally, while we have implemented several existing baseline methods in our active benchmark experiments, there exists a large suite of active selection methods in the literature that could potentially be implemented and evaluated (Ren et al., 2021).

Datasets and Metrics. In future benchmarks built on top of LabelBench, we plan on incorporating datasets with distribution shift evaluation data. We believe this is a valuable future direction that aligns with many real-world scenarios. Introducing tasks beyond image classification is also an important next step, e.g., natural language processing tasks, vision tasks such as object detection and segmentation, and generative modeling in both vision and language applications.

Models. With the computational speed-ups afforded by selection-via-proxy and our pre-computation steps, we can scale the model to ViT-L and ViT-H architectures (Dosovitskiy et al., 2020) without incurring significant computational costs. Moreover, in developing benchmarks that better reflect real-world applications, contributors could implement a blend of selection-via-proxy and end-to-end fine-tuning during example selection, instead of fine-tuning at every iteration or only once at the end of labeling.

6. Conclusion

In this paper, we present LabelBench, a comprehensive and computationally efficient framework for evaluating label-efficient learning. LabelBench puts label-efficient learning under the spotlight of fine-tuning large pretrained model. A pivotal realization from our experiments is the necessity to re-calibrate our focus. Beyond algorithm development in isolated research areas, it is crucial to study how existing tools – such as pre-trained models, semi-supervised learning, and active learning – can be skillfully intertwined and leveraged together.

References

Ash, J. T., Zhang, C., Krishnamurthy, A., Langford, J., and Agarwal, A. Deep batch active learning by diverse, uncertain gradient lower bounds. *ArXiv*, abs/1906.03671, 2019.

Ash, J. T., Goel, S., Krishnamurthy, A., and Kakade, S. M. Gone fishing: Neural active learning with fisher embeddings. In *Neural Information Processing Systems*, 2021.

Bartlett, M., Romiti, S., Sharmanska, V., and Quadrianto, N. Okapi: Generalising better by making statistical matches match. *arXiv preprint arXiv:2211.05236*, 2022.

Beck, N., Sivasubramanian, D., Dani, A., Ramakrishnan, G., and Iyer, R. K. Effective evaluation of deep active learning on image classification tasks. *ArXiv*, abs/2106.15324, 2021.

Beery, S., Agarwal, A., Cole, E., and Birodkar, V. The iwildcam 2021 competition dataset. *arXiv preprint arXiv:2105.03494*, 2021.

Berthelot, D., Carlini, N., Cubuk, E. D., Kurakin, A., Sohn, K., Zhang, H., and Raffel, C. Remixmatch: Semi-supervised learning with distribution matching and augmentation anchoring. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=HklkeR4KPB.

Borsos, Z., Tagliasacchi, M., and Krause, A. Semi-supervised batch active learning via bilevel optimization. In *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 3495–3499, 2021. doi: 10.1109/ICASSP39728.2021.9414206.

Cai, Z., Ravichandran, A., Favaro, P., Wang, M., Modolo, D., Bhotika, R., Tu, Z., and Soatto, S. Semi-supervised vision transformers at scale. In Oh, A. H., Agarwal, A., Belgrave, D., and Cho, K. (eds.), *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=7a2IgJ7V4W.

Chan, Y.-C., Li, M., and Oymak, S. 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)*, pp. 3455–3459, 2021. doi: 10.1109/ICASSP39728.2021.9414665.

Chen, T., Kornblith, S., Norouzi, M., and Hinton, G. E. A simple framework for contrastive learning of visual representations. *ArXiv*, abs/2002.05709, 2020.

Christie, G., Fendley, N., Wilson, J., and Mukherjee, R. Functional map of the world. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 6172–6180, 2018.

Citovsky, G., DeSalvo, G., Gentile, C., Karydas, L., Rajagopalan, A., Rostamizadeh, A., and Kumar, S. Batch active learning at scale. *Advances in Neural Information Processing Systems*, 34:11933–11944, 2021.

Coleman, C., Chou, E., Katz-Samuels, J., Culatana, S., Bailis, P., Berg, A. C., Nowak, R., Sumbaly, R., Zaharia, M., and Yalniz, I. Z. Similarity search for efficient active learning and search of rare concepts. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pp. 6402–6410, 2022.

Coleman, C. A., Yeh, C., Mussmann, S., Mirzasoleiman, B., Bailis, P. D., Liang, P., Leskovec, J., and Zaharia, M. A. Selection via proxy: Efficient data selection for deep learning. *ArXiv*, abs/1906.11829, 2019.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., and Houlsby, N. An image is worth 16x16 words: Transformers for image recognition at scale. *ArXiv*, abs/2010.11929, 2020.

Emam, Z. A. S., Chu, H.-M., Chiang, P.-Y., Czaja, W., Leapman, R., Goldblum, M., and Goldstein, T. Active learning at the imagenet scale. *arXiv preprint arXiv:2111.12880*, 2021.

Gao, M., Zhang, Z., Yu, G., Arık, S. Ö., Davis, L. S., and Pfister, T. Consistency-based semi-supervised active learning: Towards minimizing labeling cost. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part X 16*, pp. 510–526. Springer, 2020.

Hacohen, G., Dekel, A., and Weinshall, D. Active learning on a budget: Opposite strategies suit high and low budgets. In Chaudhuri, K., Jegelka, S., Song, L., Szepesvari, C., Niu, G., and Sabato, S. (eds.), *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 8175–8195. PMLR, 17–23 Jul 2022. URL https://proceedings.mlr.press/v162/hacohen22a.html.

Huang, S., Wang, T., Xiong, H., Huan, J., and Dou, D. Semi-supervised active learning with temporal output discrepancy. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 3447–3456, October 2021.

Jia, C., Yang, Y., Xia, Y., Chen, Y.-T., Parekh, Z., Pham, H., Le, Q. V., Sung, Y.-H., Li, Z., and Duerig, T. Scaling up visual and vision-language representation learning with noisy text supervision. In *International Conference on Machine Learning*, 2021.

Kirillov, A., Mintun, E., Ravi, N., Mao, H., Rolland, C., Gustafson, L., Xiao, T., Whitehead, S., Berg, A. C., Lo, W.-Y., et al. Segment anything. *arXiv preprint arXiv:2304.02643*, 2023.

Koh, P. W., Sagawa, S., Marklund, H., Xie, S. M., Zhang, M., Balsubramani, A., Hu, W., Yasunaga, M., Phillips, R. L., Gao, I., et al. Wilds: A benchmark of in-the-wild distribution shifts. In *International Conference on Machine Learning*, pp. 5637–5664. PMLR, 2021.

Kolesnikov, A., Beyer, L., Zhai, X., Puigcerver, J., Yung, J., Gelly, S., and Houlsby, N. Big transfer (bit): General visual representation learning. In *European Conference on Computer Vision*, 2019.

Lagunas, M., Impata, B., Martinez, V., Fernandez, V., Georgakis, C., Braun, S., and Bertrand, F. Transfer learning for fine-grained classification using semi-supervised learning and visual transformers. *arXiv preprint arXiv:2305.10018*, 2023.

Lewis, D. D. A sequential algorithm for training text classifiers: Corrigendum and additional data. In *Acm Sigir Forum*, volume 29, pp. 13–19. ACM New York, NY, USA, 1995.

Lüth, C. T., Bungert, T. J., Klein, L., and Jaeger, P. F. Toward realistic evaluation of deep active learning algorithms in image classification. *ArXiv*, abs/2301.10625, 2023.

Mittal, S., Tatarchenko, M., Çiçek, Ö., and Brox, T. Parting with illusions about deep active learning. *ArXiv*, abs/1912.05361, 2019.

Mussmann, S., Reisler, J., Tsai, D., Mousavi, E., O'Brien, S., and Goldszmidt, M. Active learning with expected error reduction. *arXiv preprint arXiv:2211.09283*, 2022.

Ouali, Y., Hudelot, C., and Tami, M. An overview of deep semi-supervised learning, 2020.

Pukelsheim, F. *Optimal design of experiments*. SIAM, 2006.

Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pp. 8748–8763. PMLR, 2021.

Ren, P., Xiao, Y., Chang, X., Huang, P.-Y., Li, Z., Gupta, B. B., Chen, X., and Wang, X. A survey of deep active learning. *ACM Comput. Surv.*, 54(9), oct 2021. ISSN 0360-0300. doi: 10.1145/3472291. URL https://doi.org/10.1145/3472291.

Scheffer, T., Decomain, C., and Wrobel, S. Active hidden markov models for information extraction. In *Advances in Intelligent Data Analysis: 4th International Conference, IDA 2001 Cascais, Portugal, September 13–15, 2001 Proceedings 4*, pp. 309–318. Springer, 2001.

Sener, O. and Savarese, S. Active learning for convolutional neural networks: A core-set approach. *arXiv: Machine Learning*, 2017.

Settles, B. Active learning literature survey. 2009.

Siméoni, O., Budnik, M., Avrithis, Y., and Gravier, G. Rethinking deep active learning: Using unlabeled data at model training. In *2020 25th International Conference on Pattern Recognition (ICPR)*, pp. 1220–1227, 2021. doi: 10.1109/ICPR48806.2021.9412716.

Singh, A., Hu, R., Goswami, V., Couairon, G., Galuba, W., Rohrbach, M., and Kiela, D. Flava: A foundational language and vision alignment model. *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 15617–15629, 2021.

Sohn, K., Berthelot, D., Carlini, N., Zhang, Z., Zhang, H., Raffel, C. A., Cubuk, E. D., Kurakin, A., and Li, C.-L. Fixmatch: Simplifying semi-supervised learning with consistency and confidence. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., and Lin, H. (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 596–608. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/06964dce9addb1c5cb5d6e3d9838f733-Paper.pdf.

Tamkin, A., Nguyen, D., Deshpande, S., Mu, J., and Goodman, N. Active learning helps pretrained models learn the intended task. *arXiv preprint arXiv:2204.08491*, 2022.

van Engelen, J. E. and Hoos, H. H. A survey on semi-supervised learning. *Machine Learning*, 109(2):373–440, Feb 2020. ISSN 1573-0565. doi: 10.1007/s10994-019-05855-6. URL https://doi.org/10.1007/s10994-019-05855-6.

Wang, W., Bao, H., Dong, L., Bjorck, J., Peng, Z., Liu, Q., Aggarwal, K., Mohammed, O. K., Singhal, S., Som, S., and Wei, F. Image as a foreign language: Beit pretraining for all vision and vision-language tasks. *ArXiv*, abs/2208.10442, 2022a.

Wang, Y., Chen, H., Fan, Y., SUN, W., Tao, R., Hou, W., Wang, R., Yang, L., Zhou, Z., Guo, L.-Z., Qi, H., Wu, Z., Li, Y.-F., Nakamura, S., Ye, W., Savvides, M., Raj, B., Shinozaki, T., Schiele, B., Wang, J., Xie, X., and Zhang, Y. USB: A unified semi-supervised learning benchmark for classification. In *Thirty-sixth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2022b. URL https://openreview.net/forum?id=QeuwINa96C.

Wen, Z., Pizarro, O., and Williams, S. B. Training from a better start point: Active self-semi-supervised learning for few labeled samples. 2022.

Wortsman, M., Ilharco, G., Kim, J. W., Li, M., Kornblith, S., Roelofs, R., Lopes, R. G., Hajishirzi, H., Farhadi, A., Namkoong, H., et al. Robust fine-tuning of zero-shot models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7959–7971, 2022.

Xing, Z., Dai, Q., Hu, H., Chen, J., Wu, Z., and Jiang, Y.-G. Svformer: Semi-supervised video transformer for action recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 18816–18826, 2023.

Yao, L., Huang, R., Hou, L., Lu, G., Niu, M., Xu, H., Liang, X., Li, Z., Jiang, X., and Xu, C. Filip: Fine-grained interactive language-image pre-training. *ArXiv*, abs/2111.07783, 2021.

Yehuda, O., Dekel, A., Hacohen, G., and Weinshall, D. Active learning through a covering lens. In Koyejo, S., Mohamed, S., Agarwal, A., Belgrave, D., Cho, K., and Oh, A. (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 22354–22367. Curran Associates, Inc., 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/8c64bc3f7796d31caa7c3e6b969bf7da-Paper-Conference.pdf.

Yu, J., Wang, Z., Vasudevan, V., Yeung, L., Seyedhosseini, M., and Wu, Y. Coca: Contrastive captioners are image-text foundation models. *Trans. Mach. Learn. Res.*, 2022, 2022.

Yuan, L., Chen, D., Chen, Y.-L., Codella, N. C. F., Dai, X., Gao, J., Hu, H., Huang, X., Li, B., Li, C., Liu, C., Liu, M., Liu, Z., Lu, Y., Shi, Y., Wang, L., Wang, J., Xiao, B., Xiao, Z., Yang, J., Zeng, M., Zhou, L., and Zhang, P. Florence: A new foundation model for computer vision. *ArXiv*, abs/2111.11432, 2021.

Zhai, X., Kolesnikov, A., Houlsby, N., and Beyer, L. Scaling vision transformers. *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1204–1213, 2021.

Zhan, X., Wang, Q., Huang, K.-H., Xiong, H., Dou, D., and Chan, A. B. A comparative survey of deep active learning. *ArXiv*, abs/2203.13450, 2022.

Zhang, B., Wang, Y., Hou, W., WU, H., Wang, J., Okumura, M., and Shinozaki, T. Flexmatch: Boosting semi-supervised learning with curriculum pseudo labeling. In Ranzato, M., Beygelzimer, A., Dauphin, Y., Liang, P., and Vaughan, J. W. (eds.), *Advances in Neural Information Processing Systems*, volume 34, pp. 18408–18419. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper_files/paper/2021/file/995693c15f439e3d189b06e89d145dd5-Paper.pdf.

Zhang, J., Katz-Samuels, J., and Nowak, R. Galaxy: Graph-based active learning at the extreme. *arXiv preprint arXiv:2202.01402*, 2022.

Zhu, X., Lafferty, J. D., and Ghahramani, Z. Combining active learning and semi-supervised learning using gaussian fields and harmonic functions. In *International Conference on Machine Learning*, 2003.

Zhu, X. J. Semi-supervised learning literature survey. 2005.

A. Active Learning Strategies

We describe the active learning setup and introduce some basic active learning strategies in this section.

We start by describing the active learning setups. The learner starts with a large pool of unlabeled examples $U = \{x_i\}_{i \in [n]}$ and a small fraction of labeled examples L, where each example x comes from the input space $\mathcal{X}$ with some unknown label y belonging to labeling space $\mathcal{Y}$. At the beginning of every batch, adhering to a certain active learning strategy, the algorithm adaptively selects new examples to label (i.e., moving the labeled examples from U to L) based on the current model h. We use $h_\theta(x)$ to denote the predicated softmax vector; we also use $[h_\theta(x)]_i$ to denote the i-th coordinate of the prediction. The model h is then retrained based on the updated dataset L, U with a certain training strategy. The ultimate goal is to use as small of a labeling budget as possible to achieve some desired performance (e.g., small error).

Below we introduce some active learning strategies that have been used in our experiments.

- Confidence (Lewis, 1995): An uncertainty-based active learning strategy that selects examples with the least confidence score in terms of the top predicated class, i.e., $\max_i [h_\theta(x)]_i$.
- Entropy (Settles, 2009): An uncertainty-based active learning strategy that selects examples with the highest entropy of the predicted distribution $h_\theta(x)$.
- Margin (Scheffer et al., 2001): An uncertainty-based active learning strategy that selects examples with the smallest prediction margin between the top-2 classes, i.e., $[h_\theta(x)]_{i^\star} - \max_{i \neq i^\star} [h_\theta(x)]_i$, where $i^\star = \arg\max [h_\theta(x)]_i$.
- BADGE (Ash et al., 2019): An active learning strategy that incorporates both uncertainty and diversity in sampling using k-means++ in the hallucinated gradient space.
- BAIT (Ash et al., 2021): An active learning strategy that incorporates both uncertainty and diversity by sampling from a Fisher-based selection objective using experimental design. BAIT can be viewed as a more general version of BADGE.
- GALAXY (Zhang et al., 2022): A graph-based active learning strategy that incorporates both uncertainty and diversity by first building a graph and then adaptively sampling examples on the shortest path of the graph.

B. Hyper-parameter tuning

Adhering to the guidelines proposed by (Lüth et al., 2023), we are transparent about our method configuration, which many active learning studies fail to report. For each dataset, we utilize a separate validation set, typically with size around 10% of the training pool. We begin the process by adjusting the hyper-parameters on a subset of the training data, which is randomly queried and constitutes around 10% of the total training pool. The selection of hyper-parameters is mainly based on the criterion of achieving the highest accuracy on the validation set. These hyper-parameters are then consistently applied in all subsequent data collection batches and across varied experimental settings (e.g., experiments with different batch sizes). While it's arguable that this fixed hyper-parameter approach may not always yield optimal results, it is practically suitable in real-world scenarios and allows for fair comparison in this paper.

C. Speeding Up Existing Active Learning Algorithms

Notation. Let $U = \{x_1, ..., x_N\}$ denote the set of N unlabeled examples and K denote the number of classes in a dataset. For each $i \in [N]$, we further use $p_i \in \mathbb{R}^K$ and $\widehat{y}_i \in [K]$ to denote the predictive probability and predictive label respectively on example x_i. Lastly, we use $v_1, ..., v_N \in \mathbb{R}^d$ to denote the penultimate layer output of a neural network where d is the number of dimensions.

Implementation of BADGE. Current implementation of BADGE (`https://github.com/JordanAsh/badge`) explicitly compute gradient embeddings g_i of each unlabeled example x_i. In particular, each g_i is a Kd-dimensional vector and can be computed via vectorizing $q_i v_i^\top$ where $q_i \in \mathbb{R}^K$ is defined as

$$q_{i,j} = \begin{cases} 1 - p_{i,j} & \text{if} \quad j = \widehat{y}_i \\ -p_{i,j} & \text{otherwise} \end{cases}$$

During each iteration of BADGE (B iterations in total for each batched selection of B examples), the dominating computation lies in computing the ℓ-2 distance between N pairs of gradient embeddings. Currently, this is implemented by naively computing $\|g_i - g_j\|_2$ with an $O(Kd)$ complexity each.

We instead use the following decomposition:

$$\begin{aligned}\|g_i - g_j\|_2 &= \|g_i\|_2 + \|g_j\|_2 - 2g_i^\top g_j \\ &= \|q_i\|_2 \cdot \|v_i\|_2 + \|q_j\|_2 \cdot \|v_j\|_2 - 2 \cdot (q_i^\top q_j) \cdot (v_i^\top v_j).\end{aligned}$$

where the last expression can be computed with $O(K + d)$ complexity, effectively reducing the computational time by an order of magnitude. In our ImageNet experiment, this means a 512-fold reduction in computation time.