Label-Efficient Deep Learning with the Pre-trained Models

Ziting Wen

Msc (SJTU)

A thesis submitted in fulfillment
of the requirements of the degree of
Doctor of Philosophy

Australian Centre for Robotics
School of Aerospace, Mechanical and Mechatronic Engineering
The University of Sydney

Submitted April 2024; revised February 2025

Statement of Originality

I hereby declare that this submission is my own work and that, to the best of my knowledge and belief, it contains no material previously published or written by another person nor material which to a substantial extent has been accepted for the award of any other degree or diploma of the University or other institute of higher learning, except where due acknowledgement has been made in the text.

2 April 2024

Authorship Attribution Statement

Parts of this thesis are based on the following:

1. Chapter 3 of this thesis is currently under submission and is available as a preprint, Ziting Wen, Oscar Pizarro, and Stefan Williams. NTKCPL: Active learning on top of self-supervised model by estimating true coverage. arXiv preprint:2306.04099, 2023 (Wen et al., 2023). I designed the study, conducted the experiments, analysed the results and wrote the drafts.

2. Chapter 4 of this thesis is published, Ziting Wen, Oscar Pizarro, and Stefan Williams. Feature alignment: Rethinking efficient active learning via proxy in the context of pre-trained models. Transactions on Machine Learning Research, 2024. (Wen et al., 2024). I designed the study, conducted the experiments, analysed the results and wrote the drafts.

3. Chapter 5 of this thesis is published, Ziting Wen, Oscar Pizarro, and Stefan Williams. Active self-semi-supervised learning for few labeled samples. Neurocomputing, 614:128772, 2025 (Wen et al., 2025). I designed the study, conducted the experiments, analyzed the results, and wrote the drafts.

In addition to the statements above, in cases where I am not the corresponding author of a published item, permission to include the published material has been granted by the corresponding author.

2 April 2024

As supervisor for the candidature upon which this thesis is based, I can confirm that the authorship attribution statements above are correct.

2 April 2024

Abstract

The test error of deep learning models often follows a power-law decrease with the increase of training data and model size. A powerful model requires collecting a significant amount of training data. Data collection, especially the labeled data, is expensive and labor-intensive. It hinders the application of deep learning models in many fields, especially those requiring specialized knowledge such as medical and other research domains. The recent development of pre-training techniques has paved the way to alleviate this issue. Self-supervised pre-training produces a valuable feature extractor without any annotations. Subsequently, the pre-trained model is applied to specific tasks of interest by fine-tuning with labeled data. The pre-training fine-tuning framework significantly reduces the demand for human annotations and accelerates model convergence, gradually replacing training from scratch as the main approach in deep learning training. However, as pre-training cannot leverage unlabeled data to enhance the task head (such as the classifier head) and the pre-trained features may differ from those suitable for the target task of interest, simple fine-tuning still necessitates a considerable amount of labeled data.

To address this issue, we propose to integrate active learning and semi-supervised learning on top of pre-training. Combining pre-trained models with active learning introduces challenges such as intensified phase transitions and increased training costs. The intensified phase transition implies that the types of samples to be selected change rapidly with the alteration of annotation quantity, so existing methods are effective only within a limited budget range of labeled data. To tackle this, we introduce a novel active learning strategy, Neural Tangent Kernel Clustering-Pseudo-Labels (NTKCPL), which allows active learning to directly estimate the model's empirical risk in the active learning pool. Additionally, based on the analysis of estimation errors, we propose a pseudo-label generation method to reduce the estimation error. Experimental results demonstrate that our approach outperforms existing methods and has a wider effective labeling budget range. Furthermore, since pre-trained models often large scale and require significant time to train, combining them with active learning significantly increases computational time. Therefore, we propose a new efficient active learning framework that improves active learning accuracy by aligning training methods and active learning features. Experiments

show that our method significantly reduces computation time without increasing the overall costs (sum of annotation and computational costs) compared to the standard active learning. Moreover, to further diminish the need for manual annotations, we introduce Active Self-Semi-Supervised Learning (AS3L). By comprehensively utilizing pre-training weight initialization, active sample selection, and semi-supervised learning guided by prior pseudo-labels, we greatly enhance the model performance when dealing with limited labeled data (an average of 1-4 samples per class).

Acknowledgements

I would like to express my sincere gratitude to my advisors, Prof. Stefan Williams and Prof. Oscar Pizarro. Without your support and guidance, my PhD journey would have been impossible. I am truly grateful to Stefan for always reminding me to step back and consider the big picture when I found myself immersed in the quagmire of intricate details. Learning to prioritize real-world needs over fixating on the marginal improvements in pre-defined tasks is a crucial lesson I learned from your guidance. Special thanks to Oscar for your assistance in various aspects of my doctoral research, continuous meetings (especially the early morning meetings after you moved to Norway), meticulous paper revisions, and inspiring me to apply the algorithms to various real-world scenarios. Your encouragement to "Don't be shy" will stay with me for a lifetime.

I would like to express my gratitude to the progress evaluation panel during my doctoral studies, Dr. Andrew Hill, Dr. Viorela Ila, and Dr. Donald Dansereau. Your valuable suggestions not only facilitated the smooth progress of my doctoral research but also enhanced the overall quality of my study. Additionally, thank you to Assoc. Prof. Guodong Shi for encouraging me to participate in tutoring for AMME3500, which provided me with valuable insights into teaching activities.

Gratitude to all the members of the Marine Robotics Group: Jackson Shields, Heather Doig, Wilhelm Marais, Gideon Billings, Suraj Bijjahalli, Jay Zhang. Coffee talks and research meetings have immersed me in many unique and intriguing aspects of marine robotics research. Thank you to Ariell Friedman for providing me with the opportunity to apply the developed algorithms to real-world marine image annotation.

Thank you to my friends, Bowen Yi, Ruigang Wang, encountering alumni at ACFR always feels incredibly warm. Feiyu Wang, sharing similar research interests has brought us many insightful conversations, both technical and regarding our careers. I am grateful to Xi Wang, Johnny Cheng, Yurui Zhang, Yijun Chen, Dechuan Liu, Yingjie Mi—lunchtimes wouldn't have been as joyful without all of you. Thanks to my friends Yunyi Gong and Xuyang Chen, your joyful conversations have been a relief from the unique pressures of PhD life.

Grateful to my parents, without your unwavering support throughout, none of these stories would have unfolded. Love you.

If a model works well with few annotations,
it is by standing on the shoulders of pre-training.

Contents

Statement of Originality i

Authorship Attribution Statement ii

Abstract iii

Acknowledgements v

Contents vii

List of Figures xi

List of Tables xvi

List of Algorithms xix

List of Acronyms xx

Glossary xxi

1 Introduction 1

1.1 Motivation 1

1.2 Problem Statement 3

1.3 Contributions 5

1.4 Outline 7

2 Background and Related Works **9**

2.1 Background . . . 9

2.2 Pre-training: Self-supervised Learning . . . 10

2.2.1 Problem Setting . . . 10

2.2.2 Pretext task . . . 11

2.2.3 Invariant feature learning . . . 13

2.2.4 Transfer to downstream tasks . . . 16

2.3 Active Learning . . . 17

2.3.1 Problem Setting . . . 17

2.3.2 Active Learning Strategies . . . 17

2.3.3 Low-Budget Strategy and Phase Transition . . . 24

2.3.4 Sampling Efficiency . . . 25

2.4 Semi-supervised Learning . . . 26

2.4.1 Problem Setting . . . 26

2.4.2 Semi-Supervised Learning from Scratch . . . 27

2.4.3 Semi-Supervised Learning with Self-Supervision . . . 30

2.4.4 Active Semi-Supervised Learning . . . 31

2.5 Summary . . . 33

3 NTKCPL: Active Learning on Top of the Self-Supervised Model **35**

3.1 Introduction . . . 35

3.2 Insight . . . 38

3.3 Preliminaries . . . 41

3.4 Method . . . 42

3.4.1 Framework . . . 42

3.4.2 Approximation Error . . . 44

3.4.3 Clustering Pseudo-Label . . . 48

3.4.4 Computational Complexity . . . 49

3.5 Results . . . 50

3.5.1 Datasets . . . 50
3.5.2 Baseline . . . 51
3.5.3 Implementations . . . 51
3.5.4 Main Experiment Results . . . 52
3.5.5 Detailed Study . . . 56
3.5.6 Running Time . . . 60
3.5.7 Empirical Evidence for Approximation Analysis . . . 61
3.6 Limitations and Discussions . . . 64
3.7 Summary . . . 64

4 Efficient Active Learning with a Large Pre-trained Model 65
4.1 Introduction . . . 65
4.2 Preliminaries . . . 69
4.2.1 Selection via Proxy . . . 69
4.2.2 LogME-PED . . . 70
4.3 Observations and Analysis . . . 71
4.3.1 Impact of Sample Selection Differences on SVPp Performance 71
4.3.2 Role of Region A2 and B2 in Improving SVPp Performance . 74
4.3.3 Discussions . . . 77
4.4 Method . . . 78
4.4.1 Aligned Selection via Proxy . . . 78
4.4.2 Computational Complexity . . . 80
4.5 Results . . . 81
4.5.1 Experiment Setup . . . 82
4.5.2 Metric . . . 84
4.5.3 Experiment Results . . . 85
4.5.4 Running Time . . . 89
4.5.5 Detailed Study . . . 90
4.5.6 Improvement from Small Batchsize . . . 97
4.6 Limitations and Discussions . . . 98
4.7 Summary . . . 99

5 Training from a Better Start Point: Active Self-Semi-Supevivsed Learning **100**

5.1 Introduction . . . 100

5.2 Observations . . . 103

5.3 Method . . . 106

5.3.1 Preliminaries on Semi-Supervised Training . . . 106

5.3.2 Active Self-Semi-Supervised Learning Framework . . . 107

5.3.3 Prior Pseudo-label Generation . . . 109

5.3.4 Single-shot Active Labeled Sample Selection . . . 110

5.4 Results . . . 113

5.4.1 Results on CIFAR-10, CIFAR-100 and STL-10 . . . 113

5.4.2 Results on ImageNet . . . 116

5.4.3 Model Detailed Analysis . . . 118

5.5 Limitations . . . 125

5.6 Summary . . . 126

6 Conclusion **127**

6.1 Summary . . . 127

6.2 Future Work . . . 130

6.2.1 Cost-aware Learning . . . 130

6.2.2 Trustworthy Label-Efficient Learning for Scientific Research . . . 130

6.2.3 Multi-Modal Pre-training for Label-Efficient Learning . . . 131

6.2.4 Life-long Learning . . . 132

List of References **133**

List of Figures

1.1 The concept maps of the main label-efficient learning methods that have appeared in the literature. 4

1.2 The overview of label-efficient learning on top of self-supervised pre-trained models. 5

2.1 Conceptual diagram of the context prediction pretext task: random sampling of two patches from input images and training the model to predict the spatial relationship between these patches. The image is from Context Prediction (Doersch et al., 2015). 11

2.2 The framework of the masked autoencoder involves dividing input images into small patches and randomly masking out 75% of them. Subsequently, the model is trained to reconstruct the entire image. The image is from MAE (He et al., 2022). 12

2.3 The framework of self-supervised learning, where both the original image and its augmented version serve as inputs to the feature encoder. 13

2.4 The framework of active learning, where the model re-training and the sample selection are repeatedly performed until the annotation budget is exhausted or the model performance reaches satisfactory levels. . . 18

3.1 Active learning gain of TypiClust (Hacohen et al., 2022) (designed for the low-budget regime) and BADGE (Ash et al., 2020) (designed for the high-budget regime) on CIFAR-100, both trained with and without self-supervised models. 37

3.2 Visualizing Coverage for different active learning strategies. 40

3.3 The framework of our method NTKCPL. 42

3.4 Performance of different active learning strategies. The shaded area represents standard deviation. 54

3.5 Active learning gain of different active learning strategies. The shaded area represents the standard deviation. 55

3.6 Estimation of empirical risk on the whole active learning pool for CIFAR-100, where the NN using true labels represents the ground truth. 56

3.7 The ablation study on ImageNet-100. 57

3.8 Effect of the maximum number of clusters C_{max} on active learning performance on CIFAR-10. 59

3.9 Performance of different active learning strategies. The shaded area represents standard deviation. 60

3.10 Comparison of cumulative sample selection time on single RTX 3090 GPU on (a) SVHN and (b) CIFAR-100. 62

3.11 The validity of the claim $(argmax\hat{f}_y(x) = g(onehot(argmax\hat{f}_{cpl}(x))))$ at different numbers of annotations on (a) CIFAR-10 and (b) CIFAR-100. The shaded area represents standard deviation. 62

3.12 Proportion of approximation error caused by impure samples within CPL to the P_{nff} term on (a) CIFAR-10 and (b) CIFAR-100. The shaded area represents standard deviation. 63

3.13 Proportion of approximation error caused by over-clustering of CPL to the P_{fnf} term on (a) CIFAR-10 and (b) CIFAR-100. The shaded area represents standard deviation. 63

4.1 Comparing sample selection time and accuracy in active learning (margin sampling) with different scale models. 66

4.2 Motivation of efficient active learning. 66

4.3 The Selection via Proxy based on pre-trained features (SVPp) Framework. 69

4.4 Sample Selection Discrepancy: Proxy vs. Fine-tuned Models. 72

4.5 The impact of different regions on SVPp active learning performance. Replacing samples selected by the proxy model with those from regions A1, A2, B1, B2. The active learning gain refers to the difference in accuracy between active learning and the random baseline. 73

4.6 Difficulty distribution of samples from different regions. 74

4.7 The influence of samples from regions A2 and B2 on the similarity between fine-tuned and pre-trained features. Exchange-backbone accuracy loss refers to the accuracy difference between the fine-tuned model's linear classifier placed on the pre-trained backbone and the fine-tuned model. 75

4.8 Comparison of active learning performance based on the fine-tuned model (standard active learning) and proxy model when employing different training methods. The active learning gain refers to the difference in accuracy between active learning and the random baseline (using fine-tuning). 76

4.9 Observation of redundant region samples. 77

4.10 The Data Selection Pipeline in our approach, Aligned Selection via Proxy (ASVP). 79

4.11 The accuracy of the standard method, SVPp, and our method ASVP on the ImageNet using (a) Margin, (b) BADGE, and (c) Confidence sampling. 86

4.12 The equivalent number of non-active learning label amounts comparison of the standard method, SVPp, and our method ASVP on the ImageNet. 87

4.13 The accuracy of the standard method, SVPp, and our method ASVP on the CIFAR-10 using (a) Margin, (b) BADGE, (c) Confidence and (d) ActiveFT(al) sampling. 87

4.14 The equivalent number of non-active learning label amounts comparison of the standard method, SVPp, and our method ASVP on the CIFAR-10. 88

4.15 The accuracy of the standard method, SVPp, and our method ASVP on the CIFAR-100 using (a) Margin, (b) BADGE, (c) Confidence and (d) ActiveFT(al) sampling. 89

4.16 The equivalent number of non-active learning label amounts comparison of the standard method, SVPp, and our method ASVP on the CIFAR-100. 90

4.17 The accuracy of the standard method, SVPp, and our method ASVP on the Pets using (a) Margin, (b) BADGE, (c) Confidence and (d) ActiveFT(al) sampling. 91

4.18 The equivalent number of non-active learning label amounts comparison of the standard method, SVPp, and our method ASVP on the Pets using (a) Margin, (b) BADGE, (c) Confidence and (d) ActiveFT(al) sampling. The equivalent non-active learning label amount refers to the number of samples selected by a random baseline to achieve the same accuracy as active learning. 92

4.19 Computation efficiency and overall cost comparison of the standard active learning method, SVPp, and our method ASVP on ImageNet using (a) Margin, (b) BADGE, and (c) Confidence sampling. 92

4.20 Computation efficiency and overall cost comparison of the standard active learning method, SVPp, and our method ASVP on CIFAR-10 using (a) Margin, (b) BADGE and (c) Confidence sampling. 93

4.21 Computation efficiency and overall cost comparison of the standard active learning method, SVPp, and our method ASVP on CIFAR-100 using (a) Margin, (b) BADGE and (c) Confidence sampling. 93

4.22 Computation efficiency and overall cost comparison of the standard active learning method, SVPp, and our method ASVP on Pets using (a) Margin, (b) BADGE and (c) Confidence sampling. 93

4.23 The impact of the position of updating pre-computed features on the savings of label amounts in CIFAR-10 across each active learning iteration. 95

4.24 The absolute differences in LogME-PED scores between consecutive active learning iterations on CIFAR-10. 96

4.25 The absolute differences in LogME-PED scores between consecutive active learning iterations on CIFAR-100. 96

4.26 The absolute differences in LogME-PED scores between consecutive active learning iterations on Pets. 97

5.1 Steep performance drops when reducing annotations. Accuracy trained with state-of-the-art semi-supervised learning algorithm, Flexmatch (Zhang et al., 2021), with different numbers of annotations. 101

5.2 In the CIFAR-100 dataset, consistency of sample labels with neighboring sample labels in self-supervised and semi-supervised feature spaces. 104

5.3 In the CIFAR-10 dataset, consistency of sample labels with neighboring sample labels in self-supervised and semi-supervised feature spaces. . 104

5.4 Test Accuracy of the semi-supervised models initialized with self-supervised pre-training and random initialization. Specifically, semi-supervised training utilizing 10 labeled samples on CIFAR-10. Semi-supervised training method employed: FlexMatch. 105

5.5 The framework of our Active Self-Semi-Supervised learning(AS3L). . 108

5.6 T-SNE visualization of semi-supervised and self-supervised features, where semi-supervised features are trained with 40 labels on CIFAR-10 followed our method. 110

5.7 Ablation study. Test accuracy of semi-supervised training in CIFAR-10 with 10 labels. 119

5.8 Ablation study. Test accuracy of semi-supervised training in CIFAR-10 with 40 labels. 120

5.9 The impact of the number of clusters C in an active learning strategy on the accuracy of the prior pseudo-label. 122

5.10 The impact of the number of clusters C in an active learning strategy on the class coverage. 122

5.11 The relationship between distance and dominant labels in self-supervised feature space on CIFAR-10, where the model is self-supervised training by Simsiam (Chen and He, 2021). 125

5.12 The relationship between distance and dominant labels in self-supervised feature space on CIFAR-100, where the model is self-supervised training by Simsiam (Chen and He, 2021). 125

List of Tables

3.1 Hyperparameters 52

3.2 The data, training method, and model architecture used for pre-training the model. 53

3.3 Comparison of accuracy of different active learning strategies on CIFAR-10. All results are averages over 5 runs. The best results are shown in red and the second-best results are shown in blue. 53

3.4 Accuracy of NTK with true label on CIFAR-100 58

3.5 Accuracy of NTK with true label on ImageNet-100 58

3.6 The details of pre-training methods. 59

3.7 Accuracy comparisons between the MLP-head probing (freezing the self-supervised backbone and training the MLP classifier head) and the fine-tuning on CIFAR-10. All results are averages over 5 runs. The best results are shown in red. 61

3.8 Accuracy comparisons between the MLP-head probing (freezing the self-supervised backbone and training the MLP classifier head) and the fine-tuning on CIFAR-100. All results are averages over 5 runs. The best results are shown in red. 61

3.9 Accuracy comparisons between the MLP-head probing (freezing the self-supervised backbone and training the MLP classifier head) and the fine-tuning on ImageNet-100. All results are averages over 5 runs. The best results are shown in red. 62

4.1 The hyper-parameters of the fine-tuning. 84

4.2 The hyper-parameters of the linear-probing then fine-tuning. 84

4.3 Average sample savings ratios for standard active learning and selection via proxy active learning. 86

4.4 Computation time comparison of various active learning sample selection methods on ImageNet using single V100 GPU. 89

4.5 Sample selection time of standard active learning methods, SVPp, and our proposed method ASVP on CIFAR-10, CIFAR-100, and Pets. Experiments conducted on single V100 GPU. 91

4.6 Ablation Study of ASVP. 94

4.7 Ablation Study: Comparing the average sample saving ratio using the active learning strategy BADGE. 94

4.8 Ablation Study: Comparing the average sample saving ratio using the active learning strategy confidence. 94

4.9 Ablation Study: Comparing the average sample saving ratio using the active learning strategy ActiveFT(al). 95

4.10 The influence of the position of updating pre-computed feature on average sample savings ratios on CIFAR-10. The best results are shown in red and the second-best results are in blue. 95

4.11 Positions of updating pre-computed features. The number of the PED convergence iterations is used to determine if updating pre-computed features is required. The positions are estimated by the absolute difference in LogME-PED scores between consecutive active learning iterations, where a difference smaller than 1 indicates an update. . . . 96

4.12 The influence of the number of active learning iterations on average sample savings ratios and sampling time on ImageNet. 98

5.1 Impact of self-supervised pre-training weight initialization on semi-supervised performance. The experiments adopted the semi-supervised training method, FlexMatch. 105

5.2 Comparison of accuracy on CIFAR-10, CIFAR-100 and STL-10. All results are averaged over 3 runs. The best results are shown in red and the second best results are shown in blue. 116

5.3 Comparison of practical running time on a single RTX 3090 GPU. SelfMatch time refers to the total runtime of complete semi-supervised training, while our method's time indicates the duration required to reach SelfMatch's final accuracy. 117

5.4 Comparison of accuracy on ImageNet. The best results are shown in red and the second best results are shown in blue. 118

5.5 Ablation Study on CIFAR-10. Accuracy is the average over 3 runs. . 119

5.6 Ablation study of proposed active learning strategy on CIFAR-10. Accuracy reported is the average over 3 runs. 121

5.7 Comparison of active sampling strategies. The accuracy reported is the average over 3 runs. The best results are shown in red and the second best results are shown in blue. 123

5.8 Accuracy of prior pseudo-label comparison of different active strategies and label propagation methods. The accuracy reported is the average over 3 runs. The best results are shown in red and the second best results are shown in blue. 124

5.9 Ablation study of K on CIFAR-10, accuracy of y_{prior} and Expected calibration error (ECE) reported is the average over 3 runs. 124

List of Algorithms

1 NTKCPL 44
2 CPL generation 49
3 ASVP 81

List of Acronyms

AL	Active Learning
AS3L	Active Self-Semi-Supervised Learning
CPL	Clustering Pseudo-Labels
FT	Fine-Tuning
LogME	Logarithm of Maximum Evidence
LP-FT	Linear-Probing Fine-Tuning
NTK	Neural Tangent Kernel
PED	Potential Energy Decline
PPL	Prior Pseudo-Labels
SVP	Selection via Proxy
SVPp	Selection via Proxy based on Pre-trained features

Glossary

Label-efficient learning refers to any training methods aiming at minimizing the need for human annotations while training a powerful machine learning model.

Phase transition This phenomenon occurs in active learning, where most active learning strategies that outperform a random baseline with a small number of labels may become inferior to a random baseline when the total number of labels is large, and vice versa.

Effective budget range The range of labeled samples over which active learning strategies outperform a random baseline.

High-budget refers to situations with lots of labeled samples.

Low-budget refers to situations with a limited total number of labeled samples.

Look-ahead strategy refers to active learning strategies that select new samples based on the outputs of models trained with candidate-labeled samples.

Chapter 1

Introduction

1.1 Motivation

Deep learning models have significantly enhanced machine understanding of various types of inputs, including vision (Dosovitskiy et al., 2020; He et al., 2016), language (Wolf et al., 2020), speech (Mehrish et al., 2023). However, alongside the acknowledgment of the capabilities of deep learning models, practitioners have noted that the performance of these models is closely tied to the scale of training data and models (Alabdulmohsin et al., 2022; Tay et al., 2021). Specifically, the test error of the model follows a power-law decrease with respect to the training data and model size (Hestness et al., 2017). In contrast to human learners, deep learning models often require substantial training data to exhibit generalization. The acquisition of large amounts of data, especially annotated data, is expensive and time-consuming. In fields such as medicine (Zhang et al., 2022c), wildlife (Tuia et al., 2022), or marine research (Alonso et al., 2019), where annotations often require specialized knowledge, the cost of labeling is prohibitively high. The scarcity of labeled data restricts the application of deep learning models in these domains.

To reduce the dependence of deep learning models on human annotations, the research community has explored various methods, such as pre-training and fine-tuning, active learning, and semi-supervised learning. These approaches have demonstrated

success in reducing the demand for labeled data. However, as the total amount of annotations decreases, these methods may lead to unstable training or fail to significantly improve the model's generalization performance (Kim and Lee, 2022; Mittal et al., 2019). For example, active learning typically requires a large initial pool of labeled samples to bootstrap the model, enabling it to generate high-quality feature representations or uncertainty estimations (Hacohen et al., 2022; Yuan et al., 2020). Semi-supervised learning also relies on labeled data to bootstrap, and limited labels result in noisy pseudo-labels. Training on these inaccurate labels further degrades model performance. Moreover, pre-training and fine-tuning methods can generate a valuable feature extractor without the need for labeled data. However, when fine-tuned for a specific task with limited annotations, the initially well-trained feature extractor may become distorted, leading to a degradation in the model's generalization performance as the number of annotations decreases.

Individually applying active learning, semi-supervised learning, or pre-training techniques struggles to further reduce the reliance on human annotations for training. However, these methods exhibit complementary characteristics, suggesting that their combination holds promise for improving label efficiency. For example, the difficulty in generating high-quality feature representations and reliable uncertainty estimates with limited labels is a key limitation of both active learning and semi-supervised learning. Pre-training, by contrast, can provide high-quality feature representations without any human annotations, thereby addressing this issue.

Nevertheless, integrating these techniques is far from straightforward (Mittal et al., 2019). Simple combinations of active learning with pre-trained models or semi-supervised learning often result in performance degradation (Bengar et al., 2021; Chan et al., 2021; Hacohen et al., 2022). Combining pre-training and semi-supervised learning through weight initialization improves final performance when there are an adequate number of annotations (Kim et al., 2021a). However, as the number of annotations decreases, it becomes challenging for semi-supervised learning to benefit from pre-training weight initialization (Wen et al., 2025). In other words, while these methods intuitively complement each other and have the potential to further reduce

the need for annotations through integration, this combination introduces unique challenges that are worthy of further exploration.

Motivated by this, this thesis aims to address the unique challenges that arise during the integration of these methods, ensuring that the combined approach benefits from pre-trained models, active learning, and semi-supervised learning. Consequently, we show that leveraging both annotated and unlabeled data improves the model's performance under the scenario of limited labeled data.

Self-supervised pre-training can generate valuable feature representations without relying on labeled data. These representations provide a rich feature space that can complement active learning and semi-supervised learning, particularly when labeled data is scarce. As a result, self-supervised pre-trained models serve as a strong foundation for integration. In this thesis, we propose active learning strategies and an efficient active learning framework built on pre-trained models. Furthermore, we explore the integration of semi-supervised learning to enhance model performance under conditions of extremely limited labeled data.

1.2 Problem Statement

The thriving development of self-supervised pre-training technology is steering label-efficient learning away from the traditional training-from-scratch approach, moving towards training on pre-trained models (Cai et al., 2021; Lüth et al., 2023; Munjal et al., 2022). This paradigm shift presents several advantages:

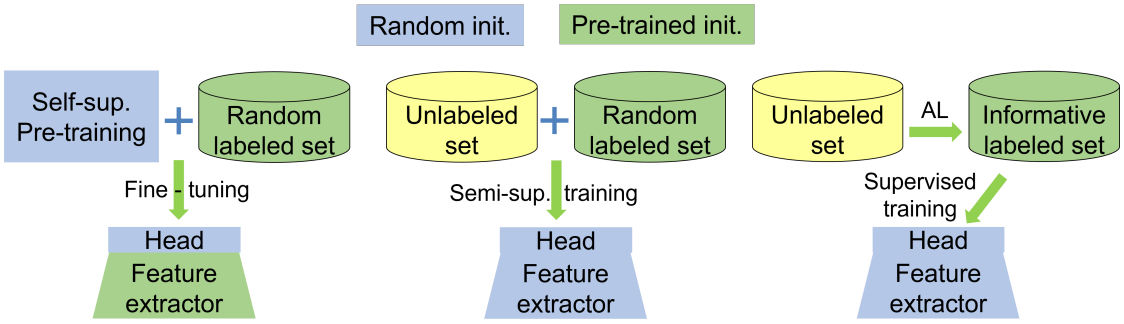
1. Effectiveness: Recent research has revealed that self-supervised pre-training models, followed by fine-tuning using labeled data, can yield faster training and higher accuracy compared to training from scratch with labeled data (Feichtenhofer et al., 2022).

2. Convenience: The combination of other label-efficient methods (active learning and semi-supervised learning) and self-supervised pre-trained models is technically straightforward. Through weight initialization, other methods can directly benefit

from the pre-training. Moreover, various pre-trained feature extractors are readily accessible (You et al., 2022) for practitioners to download and employ.

In this context, this thesis explores the unique challenges and suitable methods of conducting label-efficient learning on top of pre-trained models. The key insight is that different label-efficient learning methods help reduce the demand for annotation from different and complementary perspectives. As illustrated in fig. 1.1, active learning explores the construction of informative labeled datasets but overlooks leveraging unlabeled data for training. On the other hand, semi-supervised training and pre-training fine-tuning focus on utilizing unlabeled data to enhance models but neglect the selection of valuable labeled data. Integrating the strengths of these methods has the potential to further reduce the demand for human annotations.

Furthermore, self-supervised pre-training can produce a valuable feature extractor without any human labels. It serves as the cornerstone for other label-efficient methods such as active learning and semi-supervised learning. Thus, this thesis mainly explores label-efficient learning methods on top of self-supervised pre-trained models.



(a) Pre-train fine-tuning **(b)** Semi-supervised learning **(c)** Active learning

Figure 1.1 – The concept maps of the main label-efficient learning methods that have appeared in the literature. (a) Self-supervised pre-training fine-tuning, where the feature extractor is initially trained on all data, followed by fine-tuning the entire model with labeled data. (b) Semi-supervised learning, involving training the model from scratch using both unlabeled and randomly selected labeled data. (c) Active learning, where the model is trained from scratch using actively selected labeled data.

1.3 Contributions

Given the rapid advancement of pre-training techniques and the abundance of available pre-trained models (which serve as feature extractors), utilizing pre-trained models has become an indispensable cornerstone for reducing the demands of human annotations and enabling the application of deep learning models in domains where labeled data is expensive. To this end, this thesis focuses on conducting active learning and semi-supervised learning on top of self-supervised pre-trained models. The overview of the thesis is shown in fig. 1.2. This thesis extensively investigates the new challenges that arise in the integration of pre-trained models with active learning and semi-supervised learning. These challenges include the narrowing effective budget range in active learning, increased computational costs in active learning, and the difficulty of effectively utilizing pre-training information in semi-supervised learning. In response, the thesis proposes novel solutions tailored to address these challenges.

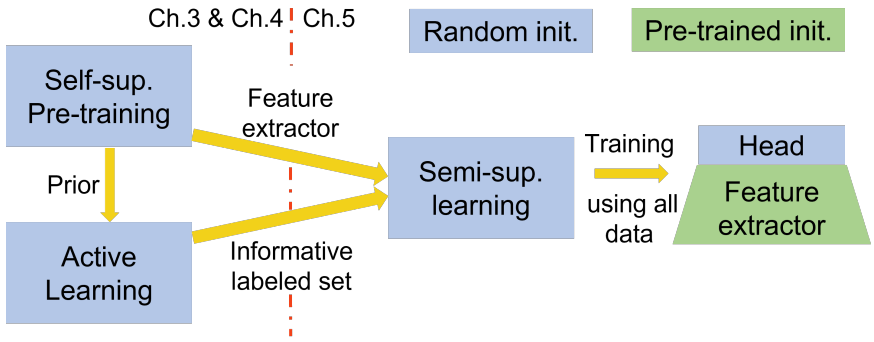


Figure 1.2 – The overview of label-efficient learning on top of self-supervised pre-trained models. Chapter 3 delves into the effectiveness of active learning when employed with pre-trained models. Chapter 4 investigates the efficiency of active learning in this context. Chapter 5 aims to enhance model performance by leveraging the advantages of pre-trained models, active learning, and semi-supervised learning.

The main contributions are summarized as follows:

Effective active learning on top of self-supervised models (Chapter 3)

- We observe that the effective budget range of existing active learning strategies significantly shortens when combined with self-supervised pre-training models.

In other words, practitioners are unable to determine, for an unseen dataset, which active learning strategy to choose to gain benefits (outperforming a random baseline) for their annotation budget. We argue that this is due to the inability of current active learning strategies to estimate the model's empirical risk in the active learning pool.

- We propose a novel active learning strategy with a wider effective budget range, termed Neural Tangent Kernel Clustering-Pseudo-Labels (NTKCPL). This approach estimates empirical risk on the whole active learning pool using pseudo-labels and NTK. To further reduce the estimation error of empirical risk in NTKCPL, we conduct a thorough analysis of the approximation error and introduce a pseudo-label generation method to mitigate it. Extensive experiments demonstrate that our approach outperforms state-of-the-art active learning methods and exhibits a wider effective budget range.

Cost-aware efficient active learning (Chapter 4)

- To more accurately reflect the cost considerations in the practical use of active learning, we propose a comprehensive evaluation metric. Specifically, we introduce the sample saving ratio to indicate how many annotated samples an active learning method can save compared to a random baseline. Building upon this, we calculate the total cost, encompassing both the annotation costs and the computational costs.

- We investigate existing efficient active learning frameworks that may lead to performance degradation, leaving practitioners facing a dilemma between computational efficiency and total cost. Through empirical analysis, we argue that the issue might stem from the disparity between pre-training features and active learning features. Consequently, we propose a novel efficient active learning approach, aligned selection via proxy (ASVP). Experiments show that our proposed ASVP saves similar or greater total costs in most cases compared to standard active learning while having similar running time to efficient active learning methods.

Active self-semi-supervised learning (Chapter 5)

- We observe that when only limited labels are available, weight initialization cannot effectively transfer self-supervised pre-training information to the semi-supervised model, resulting in a sharp decline in the performance of semi-supervised learning as the annotations decrease. Therefore, we propose a simple yet effective semi-supervised training method that generates prior pseudo-labels, PPL, through label propagation on pre-trained features. The PPL serves as an intermediary to assist the semi-supervised model in inheriting valuable information from self-supervised pre-training.
- To further improve the performance of the model in the context of extremely limited annotations, we propose a new active learning strategy to improve the accuracy of PPL. As a result, through the combination of active learning strategy, PPL-guided semi-supervised learning, and weight initialization, we significantly improve the model's performance in scenarios with extremely few labeled samples.

1.4 Outline

Chapter 2 introduces the research background and related works of this thesis, including the self-supervised pre-training fine-tuning framework, active learning strategies, semi-supervised training methods, and various combined approaches, such as active semi-supervised learning, self-supervised semi-supervised learning. In addition, we have summarized the shortcomings of existing methods.

Chapter 3 explores the challenge of a shortened effective budget range in active learning on top of pre-training models. We propose explicitly estimating the model's empirical risk in the active learning pool through NTK and clustering pseudo-labels, significantly expanding the effective budget range of active learning.

Chapter 4 focuses on the challenge of efficiently conducting active learning on top of pre-trained models. We propose a practical evaluation metric for efficient active

learning: total cost. Furthermore, we analyze the reasons behind the inefficacy of existing efficient active learning methods and propose a novel efficient active learning method based on this analysis.

Chapter 5 focuses on the effective utilization of pre-trained models in semi-supervised learning under conditions of extremely limited labeled data. We propose a new semi-supervised learning framework that integrates the advantages of active learning, significantly improving performance under conditions of extremely limited labeled data.

Chapter 6 concludes this thesis and delves into potential future research directions aimed at bridging the gap between label-efficient research and practical applications.

Chapter 2

Background and Related Works

2.1 Background

The remarkable progress in deep learning owes much to the abundance of data. In its early stages, deep learning primarily relied on supervised learning, necessitating the use of human annotations for model training. However, acquiring large amounts of annotated data is often expensive and labor-intensive. To address this issue, the research community has extensively explored approaches aiming to achieve label-efficient learning.

In this chapter, we provide an overview of key methods that reduce the demands on manual annotation. In practice, acquiring unlabeled data is significantly more convenient and cost-effective than obtaining annotated data. Consequently, the scale of unlabeled data surpasses that of annotated counterparts by a considerable margin. Addressing the utilization of unlabeled data emerges as an intriguing and significant challenge. In section 2.2, we introduce the pre-training and fine-tuning framework. This approach explores the utilization of vast amounts of unlabeled data to generate a valuable feature extractor. Subsequently, fine-tuning with a small amount of annotated data could produce high-performance models.

Moving on to section 2.3, we introduce the second category of methods, active learn-

ing. Active learning suggests that not all annotated samples have an equal impact on the model. Rather than randomly selecting samples, it proposes iteratively training the model and selecting samples to build a superior labeled dataset. This helps a system build a better model using less annotation effort.

Next, in Section 2.4, we delve into semi-supervised learning. Similar to the pre-training and fine-tuning framework, semi-supervised learning also focuses on leveraging large amounts of unlabeled data. The distinction lies in the fact that the pre-training and fine-tuning framework focuses solely on improving the feature extractor with unlabeled data, whereas semi-supervised learning directly enhances the model's performance on specific tasks of interest (such as image classification, detection, and segmentation) using a small amount of labeled data and lots of unlabeled data.

Finally, in section 2.5, we provide a summary of the limitations of existing methods and identify research gaps in the field.

2.2 Pre-training: Self-supervised Learning

2.2.1 Problem Setting

In the real world, the high cost of data labeling implies that the scale of labeled data available is significantly smaller than the pool of unlabeled data. Consequently, effectively harnessing a vast amount of unlabeled data U for model training has garnered considerable attention from researchers and practitioners. Self-supervised training stands out as a crucial approach for leveraging unlabeled data. It trains a feature extractor $f(\cdot;\theta)$ by utilizing inherent knowledge within the data to construct a self-supervised signal. The objective of this model is to generate valuable feature representations. Building upon this model, practitioners can readily transfer it to various downstream tasks, such as classification, detection, and segmentation, with minimal annotated data. In this thesis, we focus on visual self-supervised representation learning methods. In sec. 2.2.2 and sec. 2.2.3, we provide an overview of existing

self-supervised training methods, and in sec. 2.2.4, we elaborate on the techniques for fine-tuning pre-trained models.

2.2.2 Pretext task

In the realm of self-supervised representation learning, a substantial body of research has been dedicated to exploring suitable pretext tasks. Pretext tasks refer to training objectives that are unrelated to the ultimate task of interest, such as image classification or detection. Importantly, these tasks do not need additional human annotations and can be employed for representation learning.

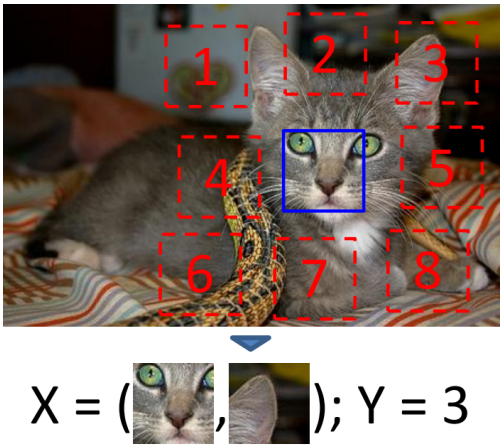


Figure 2.1 – Conceptual diagram of the context prediction pretext task: random sampling of two patches from input images and training the model to predict the spatial relationship between these patches. The image is from Context Prediction (Doersch et al., 2015).

Some studies constructed pretext tasks based on the characteristics of images. For example, some tasks involve predicting rotations (Gidaris et al., 2018) or colorization (Larsson et al., 2016). Predicting rotations entails randomly rotating input images and tasking the model with predicting the rotation status. Moreover, colorization tasks involve inputting grayscale images and predicting their colors. Other approaches leverage the spatial relationships among patches within an image to formulate pretext tasks. The Jigsaw Puzzle task extracts multiple patches from an image and shuffles their positions (Noroozi and Favaro, 2016). Then the model is used to

rearrange these patches. In the Context Prediction task (Doersch et al., 2015), two patches are randomly sampled, and the model is tasked with predicting their spatial relationship as shown in fig. 2.1.

Image reconstruction stands out as a classic category of pretext tasks, extensively explored within the research community. By incorporating methods such as network architecture design (e.g., autoencoders (Goodfellow et al., 2016)), task design (inpainting (Pathak et al., 2016), denoising autoencoders (Vincent et al., 2008)), and loss design (e.g., variational autoencoders (Kingma and Welling, 2013), beta-VAE (Kingma and Welling, 2013)), these tasks aim to enable the model to learn valuable representations while reconstructing input images. Despite a period during which reconstruction-based self-supervised learning exhibited weaker performance compared to methods based on invariant feature learning (introduced in sec. 2.2.3), recent studies have significantly improved the effectiveness of self-supervised training by introducing more challenging reconstruction tasks (Assran et al., 2022; Chen et al., 2024; He et al., 2022; Xie et al., 2022).

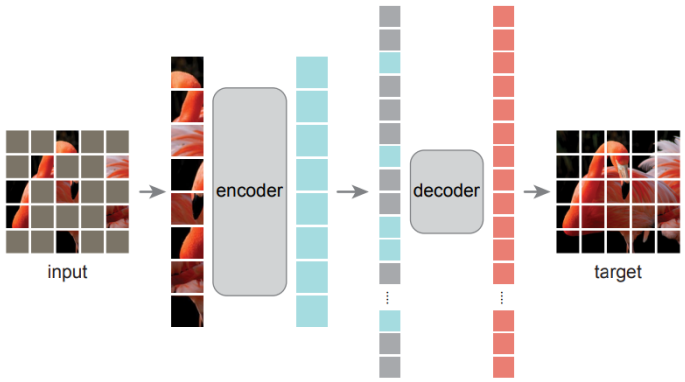


Figure 2.2 – The framework of the masked autoencoder involves dividing input images into small patches and randomly masking out 75% of them. Subsequently, the model is trained to reconstruct the entire image. The image is from MAE (He et al., 2022).

The Masked Autoencoder (MAE) suggests that visual inputs contain abundant redundant information (He et al., 2022). Consequently, to acquire valuable representations, it is essential to intensify the difficulty of the reconstruction task. Specifically, MAE

randomly masks out a substantial number of patches from input images (75% masking ratio) as depicted in fig. 2.2. Building on this argument, Evolved MAE actively selects patches to mask out during the training process (Feng and Zhang, 2023). As training progresses, it progressively masks out regions with richer information content, creating more challenging reconstruction tasks.

2.2.3 Invariant feature learning

Another category of self-supervised representation learning methods is achieved through invariant feature learning. The primary assumption is that when applying data transformations to input images, the model's output features should remain similar to those of the original images. These data transformations typically include random flips, color distortions, and so on (Chen et al., 2020a). The usual framework for such methods involves pairing the original image with its transformed counterpart as inputs to the model. Subsequently, a loss function is constructed to compel the model to produce similar features for paired inputs.

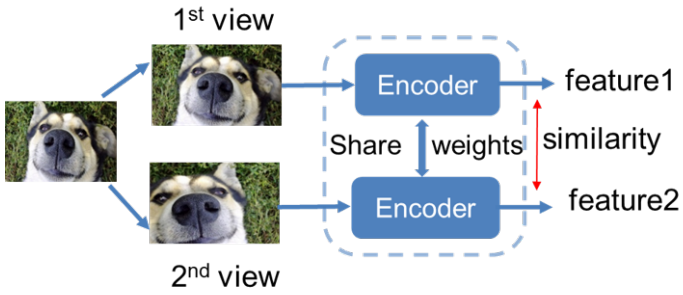


Figure 2.3 – The framework of self-supervised learning, where both the original image and its augmented version serve as inputs to the feature encoder. The loss function aims to encourage the similarity of features extracted from these two views.

However, research has uncovered that simple execution of this training process may lead to the model converging to a trivial solution, mapping all inputs to the same feature. To address this issue, methods based on contrastive learning, clustering, distillation, and redundancy reduction have been proposed.

Constrastive learning compels positive pairs of features to become more similar

and, simultaneously, encourages the features of positive and negative pairs to differ. In this context, positive pairs consist of the original input image and its transformed counterpart, while negative pairs involve randomly selected other images. SimCLR (simple framework for contrastive learning) employs other images from the same batch as negative samples (Chen et al., 2020a), necessitating a relatively large batch size for effective self-supervised training. CPC (contrastive predictive coding) samples different patches from an image to construct positive and negative pairs (Oord et al., 2018). In this context, patches that are close in distance within the original image are more likely to contain similar content and are considered positive pairs, while distant patches are considered negative pairs. Some methods tackle this challenge by implementing a memory bank to store representations of the entire dataset (Misra and Maaten, 2020; Wu et al., 2018). In each training iteration, parts of this memory bank are sampled as negative examples. However, maintaining a large memory bank requires substantial memory. To address this concern, the momentum-updated encoder is introduced (He et al., 2020) and used in other methods (Chen et al., 2020b,c). This encoder is used to generate and maintain a set of negative samples. This approach enables the use of a large number of negative samples in an online manner, addressing the memory constraints associated with maintaining a massive memory bank. However, it requires the maintenance of an additional momentum encoder and needs additional forward computations.

Clustering serves as another effective method to avoid trivial solutions. Unlike contrastive learning, which functions at the image level, clustering methods initially organize images into distinct clusters. Subsequently, these clusters are employed to assign pseudo-labels, facilitating the joint learning of features and clustering. Deep Clustering involves iterative processes of feature extraction, label updates, and model training (Caron et al., 2018), preventing it from being executed online. This may potentially contribute to training instability.

To address this issue, Online Deep Clustering has been proposed (Zhan et al., 2020), akin to the wisdom in contrastive learning methods (Misra and Maaten, 2020; Wu et al., 2018). This method avoids disjointed labels and model updates by construct-

ing a sample memory bank and a cluster center memory bank. Similarly, SwAV (Swapping Assignments between multiple Views of the same image) adopts an online clustering approach, enhancing self-supervised training effectiveness by incorporating a dual-perspective swapped prediction (Caron et al., 2020).

Self-distillation This category of methods, compared to the aforementioned positive-negative contrastive learning approaches, is more straightforward. These approaches do not necessitate negative sample pairs and instead employ self-distillation techniques to avoid trivial solutions. Both BYOL (Bootstrap Your Own Latent) (Grill et al., 2020) and DINO (Self-distillation with no labels) (Caron et al., 2021; Oquab et al., 2024) maintain a moving-average model as a teacher and an online model as a student. The original image and its transformed counterpart are fed into both the student and teacher models. Subsequently, the output representations are encouraged to be similar. SimSiam (Simple Siamese Representation Learning) further simplifies this by identifying key operations that effectively prevent trivial solutions (Chen and He, 2021). It suggests removing the moving-average model, just stopping the gradient backward-propagation in the branch serving as the teacher model.

Redundancy Reduction Some researchers, inspired by the principle of reducing redundancy in neuroscience (Barlow et al., 1961), propose the design of loss functions to encourage models to learn features with low redundancy (where the dimensions of the features are mutually independent) (Bardes et al., 2022a,b; Zbontar et al., 2021). This approach naturally avoids the collapsed features during self-supervised training while eliminating the need for complex training techniques and architectural designs employed by previous methods.

The aforementioned self-supervised learning methods effectively avoid the issue of trivial solutions (i.e., outputting the same features for any input). However, most methods require positive pairs that are semantically similar but visually different, and the generation of such samples is quite limited, often achieved through data augmentation. This makes it challenging for pre-trained models to acquire rich positive sample information beyond the level of a single image.

2.2.4 Transfer to downstream tasks

Self-supervised pre-training yields a feature extractor capable of generating valuable feature representations. The subsequent step involves transferring this feature extractor to tasks of practical interest to practitioners. Typically, this is achieved through training with a small amount of annotated data. The most common training approaches are linear probing and fine-tuning (Chen et al., 2020b; He et al., 2022). Linear probing involves freezing the weights of the pre-trained feature extractor during training and updating only the task-specific head, such as the classifier head for image classification. This method is a parameter-efficient approach, where different downstream tasks only require training and saving simple task heads, without the need to separately store the feature extractor.

However, as the tasks in self-supervised training may differ from the downstream tasks of interest, the feature extractor obtained from self-supervised pre-training may not perform well for downstream tasks, especially when there is a shift in the data domain. In such cases, fine-tuning becomes a more appropriate method. The strengths and weaknesses of fine-tuning and linear probing are complementary. While fine-tuning incurs higher training costs and lacks parameter efficiency, it exhibits better adaptability to downstream tasks. The specific procedure for fine-tuning involves initializing the network with pre-trained weights and subsequently fine-tuning all model weights using annotated data from the downstream task.

Moreover, recent research indicates that fine-tuning often sacrifices valuable pre-training information while fitting annotated data (Kumar et al., 2022; Ren et al., 2022). In other words, its performance surpasses that of linear probing on in-distribution data but deteriorates on out-of-distribution data. To address this, a two-stage training approach has been proposed, involving linear probing followed by fine-tuning. For this method, the number of training iterations for linear probing significantly influences the final performance, but configuration methods remain under-explored.

2.3 Active Learning

2.3.1 Problem Setting

Active learning is a methodology designed to improve label efficiency by leveraging the assumption that different samples have varying contributions to the model's performance. Under this premise, active learning, in contrast to random selection, attempts to identify and annotate the most valuable samples. As a result, it achieves comparable accuracy with fewer annotations compared to random selection (Ren et al., 2021).

The goal of active learning can be formulated as eq. 2.1. Here, active learning tries to find a labeled subset L from the active learning pool $D = \{(x_i, y_i)\}_{i=1}^{N}$, such that the model trained on L minimizes the empirical risk over the entire active learning pool D, where θ^* refers to the model parameters trained with L. In this thesis, we focus on a neural network model, $f(x;\theta)$, where θ represents its parameters. This model maps input x to the label space of K_y classes.

$$\begin{aligned} & min_{L \in D} \frac{1}{N} \sum_{(x,y) \in D} Loss(f(x;\theta^*), y) \\ s.t. \quad & \theta^* \in argmin_{\theta} \frac{1}{|L|} \sum_{(x,y) \in L} Loss(f(x;\theta), y) \end{aligned} \tag{2.1}$$

2.3.2 Active Learning Strategies

Most active learning strategies operate within a framework of multiple iterative rounds as shown in fig. 2.4. In each iteration of active learning, the model is trained using the annotated samples. Subsequently, the next batch of samples is selected for annotation by the oracle based on the model trained in this iteration. After multiple rounds, active learning ceases when the model performance reaches a satisfactory level or when the annotation budget is exhausted.

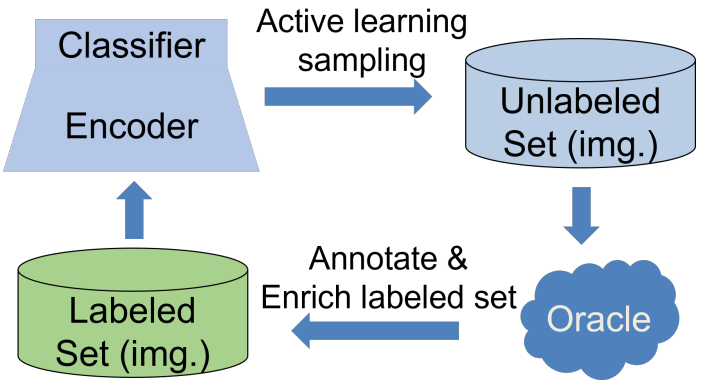


Figure 2.4 – The framework of active learning, where the model re-training and the sample selection are repeatedly performed until the annotation budget is exhausted or the model performance reaches satisfactory levels.

Since active learning is employed to select suitable samples for annotation, during the active learning process, the labels of the active learning pool are unknown. Therefore, the active learning objective, as represented in eq. 2.1, cannot be directly solved. Existing methods mostly rely on various heuristic principles or approximate upper bounds to accomplish sample selection, as elaborated below.

Uncertainty Sampling

Most uncertainty-based active learning methods are heuristic strategies, selecting new samples for annotation based on the assumption that samples with uncertain predictions from the current model have higher values. Specifically, based on the model's output $f(x;\theta^*)$, assuming that the output has already been normalized into probability form through softmax, then the samples with the smallest margin, least confidence, and largest entropy are selected and annotated, where these metrics are defined as follows, where the $sec_{max}(\cdot)$ returns the second-largest value of its inputs and the $f^i(x;\theta^*)$ denotes the i^{th} output of the $f(x;\theta^*)$.

$$margin := max(f(x;\theta^*)) - sec_{max}(f(x;\theta^*)) \tag{2.2}$$

$$confidence := max(f(x;\theta^*)) \tag{2.3}$$

$$entropy := -\sum_{i}(f^i(x;\theta^*)\ log(f^i(x;\theta^*))) \tag{2.4}$$

However, Yarin Gal, et al., argued that these metrics based on model predictions would inadvertently include uncertainty originating from the data itself (aleatoric uncertainty) rather than solely reflecting the uncertainty of model parameters (epistemic uncertainty) (Gal et al., 2017). In this context, samples with high model uncertainty imply that selecting and annotating such samples holds promise for enhancing the model. On the other hand, samples with high data uncertainty indicate inherent difficulty, making it challenging for model adjustments to significantly reduce the predictive uncertainty for these samples. Therefore, in the context of active learning, the more meaningful consideration is the model uncertainty. This objective is defined as eq. 2.5, where H denotes the entropy and $f(x)$ is the output of the model. The computational cost for solving this objective is prohibitively high, rendering it impractical for real-world applications. Fortunately, it is the mutual information between the model output $f(x)$ and the model parameters θ. Thus, this objective can be rearranged as eq. 2.6.

$$argmax_x(H(\theta|D) - E_{f(x)\sim p(f(x)|x,D)}H(\theta|x, f(x), D)) \tag{2.5}$$

$$argmax_x(H(f(x)|x, D) - E_{\theta\sim p(\theta|D)}H(f(x)|x, \theta)) \tag{2.6}$$

To address this objective, the key lies in sampling from the model parameter space and computing the term on the right of eq. 2.6. Recent research has achieved this sampling by incorporating dropout layers within neural networks. During this process, there is no need to train multiple neural networks; rather, it performs multiple forward passes on the trained network. This enables the computation of model uncertainty with a relatively modest computational cost.

As mentioned before, however, active learning involves multiple rounds of iterations. In practical applications, considering the computational cost of training neural net-

works, it is common to select a batch of samples in each iteration of active learning. However, for active learning strategies based on uncertainty sampling, batch selection often leads to redundant samples. This is because, in each iteration, the model is likely to exhibit high uncertainty for the same types of samples. Improving the model's predictions for this type doesn't necessarily require selecting all instances of it; often, selecting a small subset is sufficient for improvement. Consequently, the primary drawback of such active learning strategies is the lack of diversity.

Some advanced uncertainty metrics are proposed to improve this issue. For example, batchBALD suggests addressing the issue through the union of mutual information (Kirsch et al., 2019). Nevertheless, these approaches introduce high computational complexity. Consequently, to tackle the problem of sample redundancy, alternative avenues are explored, namely diversity-based active learning methods.

Diversity Sampling

Diversity-based strategies aim to thoroughly explore the entire active learning pool by selecting samples different from the current annotated dataset. Following this principle, K-medoids clusters samples in the feature space and selects points close to the cluster centers (Kang et al., 2004). Moreover, VAAL (Variational Adversarial Active Learning) leverages the discriminator in adversarial training to aid in selecting samples that are more dissimilar to the annotated ones (Sinha et al., 2019).

$$|\frac{1}{N}\sum_{(x,y)\in D} Loss(f(x;\theta^*),y)| \leq \mathcal{O}(\delta_s) + \mathcal{O}(\sqrt{\frac{1}{N}}) \tag{2.7}$$

Another branch of methods provides theoretical analysis demonstrating that the covering radius δ_s (i.e., the farthest distance between labeled samples and the remaining unlabeled samples) is related to the upper bound of the active learning empirical risk. as eq. 2.7, where D denotes active learning pool, N is the number of samples within the D, $f(x;\theta^*)$ is the neural networks trained on labeled set, and δ_s is coverage radius (Sener and Savarese, 2018). Based on this formulation, selecting diverse samples

can reduce the coverage radius δ_s, thereby reducing the upper bound of the model's empirical risk on the active learning pool.

However, these methods are susceptible to selecting outlier points. To address this issue, CoreGCN employs graph neural networks, taking advantage of their inherent characteristics, which include smoothness in nodes' neighborhoods (Caramalau et al., 2021). This inherent smoothness property makes graph neural networks less susceptible to the influence of outliers. Additionally, leveraging the Wasserstein distance, which measures the distance of transport between two distributions, naturally avoids the influence of outliers (Shui et al., 2020). As a result, within the framework of Coreset, Mahmood et al. propose an upper bound based on the Wasserstein distance (Mahmood et al., 2022).

In summary, diversity sampling avoids the issue of selecting redundant samples, which is beneficial for batch sample selection. However, the selected samples may be less informative, meaning that the impact of selected samples on model performance may be inconsequential.

Combination of Diversity and Uncertainty

To overcome the limitations of relying solely on uncertainty or diversity, a good approach is to design hybrid strategies that combine both uncertainty and diversity for sample selection. Some studies achieve this target by designing sample selection methods or metrics manually. Lin Yang et al. introduced the suggestive annotation algorithm, initially identifying the most uncertain samples as candidates and subsequently selecting diverse samples from this subset (Yang et al., 2017). Here, the selection of diverse samples is achieved by choosing samples with the maximum distance from the features of the existing annotated samples. BADGE (Batch Active Learning by Diverse, Uncertain Gradient) employs diversity sampling using k-means++ on gradient embeddings (Ash et al., 2020). The gradient embeddings facilitate the selection of uncertain samples.

Additionally, some learning-based methods are used to combine uncertainty and diversity. SRAAL (State-Relabeling Adversarial Active Learning) adopts an adversar-

ial training approach to formulate a hybrid method by incorporating an uncertainty indicator module (Zhang et al., 2020). This integration ensures that the selected samples benefit from both diversity and uncertainty considerations. WAAL (Wasserstein Adversarial Active Learning) calculates the Wasserstein-1 distance between annotated samples and the entire active learning pool through adversarial training (Shui et al., 2020). Subsequently, sample selection is accomplished by minimizing both the Wasserstein-1 distance and an agnostic-label upper bound loss (indicating uncertainty). Although these methods can select diverse and uncertain samples, they cannot determine whether the model predictions will change after adding these samples. In other words, they still cannot directly estimate the model's empirical risk on the active learning pool.

Expected Model Change

The underlying principle of such active learning strategies is to select samples that induce the maximum change in the model's output (Roy and McCallum, 2001). Directly training the model to determine how the model's output changes when new samples are added is impractical due to the high cost of training neural networks. Therefore, most of these methods choose samples that result in the maximum loss or gradient norm change in the model. For instance, the learning loss approach constructs an auxiliary network module to predict the model's loss for each sample and subsequently utilizes this module to select samples with the highest loss (Yoo and Kweon, 2019). Meanwhile, Rasha et al. employ traditional machine learning methods to generate pseudo-labels for unlabeled samples and then select samples with the highest gradient norm (Sheikh et al., 2020).

However, similar to uncertainty-based methods, these active learning approaches face the issue of selecting redundant samples during batch selection. LookAhead (Mohamadi et al., 2022) addresses this problem by leveraging Neural Tangents Kernel (NTK). NTK provides a good approximation of neural network outputs (Jacot et al., 2018; Lee et al., 2019), and after completing the NTK kernel computation, the computational cost for predicting the model's output changes after annotating new samples is marginal. Thus, based on NTK, LookAhead efficiently calculates the changes in

model predictions after adding newly labeled samples. However, computing the expected model changes relies on using the model's predictions as pseudo-labels, which are often inaccurate. Consequently, the samples selected may not necessarily induce the maximum change in the model.

Generative Active Learning

Several active learning algorithms have explored the integration of generative models such as Variational Autoencoders (VAE) (Kingma and Welling, 2013) or Generative Adversarial Networks (GAN) (Goodfellow et al., 2020) to generate supplementary samples, enhancing the overall model performance. A pioneering contribution in this realm was GAAL (Generative Adversarial Active Learning) (Zhu and Bento, 2017). GAAL leverages the distance between synthetic samples and the decision boundary of the classifier to assess the value of generated samples. Subsequently, the algorithm selects the most valuable synthetic samples for labeling. Notably, this approach abandons existing training samples, rendering the model's performance entirely reliant on the authenticity and diversity of samples generated by the generative model.

To merge real training samples with generated ones, Toan et al. proposed a method that identifies the most uncertain samples using the BALD uncertainty criteria (Tran et al., 2019). Subsequently, an encoder is employed to extract features from these samples, and the Auxiliary Classifier GAN (ACGAN) (Odena et al., 2017) is utilized to generate informative samples. This approach ensures the generation of informative samples. However, a drawback lies in the similarity between samples generated through this method and the selected labeled samples, resulting in redundancy.

The primary advantage of incorporating generative models lies in the ability to acquire additional labeled samples without incurring the cost of manual labeling. Nevertheless, the authenticity of generated images remains a concern, and the training cost of the generative model is notably high.

2.3.3 Low-Budget Strategy and Phase Transition

Most of the aforementioned active learning strategies require a large initial labeled dataset to effectively carry out active learning (Chandra et al., 2021; Mittal et al., 2019). For instance, uncertainty sampling calculates uncertainty based on model predictions, while diversity-based active learning relies on the model to generate sample features. A small labeled set may result in a model being unable to capture uncertainty effectively or generate high-quality features, leading to poor performance in active learning. This phenomenon is commonly referred to as the "cold start" problem (Zhu et al., 2019). In practical scenarios, there exist some applications with limited labeled data (termed as low-budget regimes), such as domains with high annotation costs like medical images (Budd et al., 2021; Zhang et al., 2022c) or specialized animal research (Norouzzadeh et al., 2021).

To improve the effectiveness of active learning in low-budget regimes, Guy et al. first proposed that different types of samples are often required in low-budget and high-budget regimes, leading to the application of distinct active learning strategies (Hacohen et al., 2022). When the total labeled quantity is limited, selecting typical samples is advisable, while in situations with abundant annotations, choosing non-typical samples is preferred. This phenomenon is referred to as a "phase transition". Active learning strategies inclined towards selecting typical samples often outperform random baselines when the total labeled quantity is low, but they become inferior as the total labeled quantity increases, which is categorized as a low-budget strategy. Conversely, strategies favoring non-typical samples are labeled as high-budget strategies. Building upon this analysis, TypiClust is introduced to enhance the performance of active learning in low-budget regimes by selecting typical samples (Hacohen et al., 2022). Specifically, TypiClust argues that samples in high-density regions are more likely to be typical and estimates density by calculating the average distance to k-nearest neighbors. To ensure diverse selection, TypiClust initially clusters samples and then chooses samples in high-density regions within each cluster. However, the clustering operation in TypiClust does not always guarantee diverse sample selection. To address the occurrence of small clusters, TypiClust sets

a maximum cluster number. When the required labeled quantity surpasses this limit, inevitable selection of multiple typical samples within the same cluster may lead to redundancy and a decrease in active learning performance. Probcover effectively addresses this issue by representing the distance between samples as a graph, iterative choosing samples with the maximum degree (i.e., having the most neighbors), and removing the selected node and its connected edges until the labeled data selection is complete (Yehuda et al., 2022).

Furthermore, Wasserstein distance, reflecting transport distance between two distributions, inherently tends to select representative samples, making it an effective active learning strategy in low-budget regimes. Exploiting this, Mahmood et al. choose samples that minimize the Wasserstein distance between the labeled and unlabeled sets (Mahmood et al., 2022). However, this method has a very high computational complexity, making it suitable only for selecting a small number of annotated samples. To address this issue, ActiveFT designs a parameterized model to efficiently minimize the Wasserstein distance, facilitating sample selection (Xie et al., 2023).

In summary, while existing low-budget active learning strategies have significantly improved the performance of active learning in low-budget regimes, they gradually become ineffective as the number of annotations increases, performing worse than random baselines (Hacohen and Weinshall, 2024). Moreover, due to the difficulty in determining the number of annotations at which low-budget active learning strategies become ineffective (Wen et al., 2023), practitioners still find it challenging to use them.

2.3.4 Sampling Efficiency

Most active learning methods operate in a multi-round mode, iteratively training models and selecting samples to label. This mode entails significant training time and costs. In the context of active fine-tuning, this phenomenon becomes more pronounced, as larger models are better equipped to effectively leverage data in unsupervised pre-training (Chen et al., 2020b; Oquab et al., 2024).

Increasing the active learning batch size, which refers to the number of samples se-

lected per active learning iteration (Citovsky et al., 2021; Zhang et al., 2022b), is one solution to enhance computational efficiency. However, it weakens the performance of active learning. Moreover, even when increasing the active learning batch size, there remains the considerable computational cost associated with training the entire network multiple times. Therefore, recent research has explored single-shot active learning based on pre-trained features (Xie et al., 2023). However, as the number of labels increases, the effectiveness of this method gradually diminishes in comparison to other standard active learning strategies.

Another solution to boost active learning efficiency is to use less computationally intensive models or training methods as proxy tasks for sample selection, known as Selection via Proxy (SVP) (Coleman et al., 2019). Typical proxy tasks include reducing model complexity, such as using models like ResNet8 or ResNet14, early stopping, or training linear classifiers or MLPs based on pre-trained features (Zhang et al., 2024). However, these proxy tasks often have an impact on active learning performance and may result in savings in computation costs not outweighing the increase in annotation costs. As a result, practitioners are often faced with a trade-off between computational efficiency and the overall cost.

2.4 Semi-supervised Learning

2.4.1 Problem Setting

Similar to self-supervised pre-training, semi-supervised training focuses on leveraging a large amount of unlabeled data to enhance model performance. However, there is a key distinction. In self-supervised pre-training, a feature extractor is trained on a pretext task different from the final task of interest. Subsequently, the model is fine-tuned using target-task labeled data. In other words, the unlabeled data has not been directly applied to improve the performance of models on the target tasks. When there is a significant difference between the feature representations required for pre-training tasks and those needed for the target tasks, the ultimate performance

may fall short of expectations.

Semi-supervised training adopts a pseudo-labeling approach, directly treating unlabeled data as if it were labeled. This enables unlabeled data to simultaneously improve both the feature extractor and the task head. The feature representation obtained through this approach is more closely aligned with the final task of interest.

2.4.2 Semi-Supervised Learning from Scratch

Semi-supervised training commonly leverages unlabeled samples through pseudo-labeling techniques (Lee et al., 2013; Rizve et al., 2021) and consistency regularization methods (Berthelot et al., 2019, 2020; Verma et al., 2022; Xie et al., 2020). These approaches compel the model to provide consistent predictions across various augmented samples.

Pseudo-labels is based on the clustering assumption, which suggests that the decision boundary of model predictions should be located in the low-density regions of the sample space. The most popular approach in practical use is to use the model's predictions on unlabeled samples as pseudo-labels for training (Lee et al., 2013). However, model predictions often contain a significant number of errors, and directly using all model predictions as pseudo-labels for training may lead to over-fitting to these errors. To address this, recent studies have employed methods such as uncertainty filtering to exclude unreliable pseudo-labels (Rizve et al., 2021) and the use of curriculum pseudo-labels (Cascante-Bonilla et al., 2021).

Some methods employ a distinct module for generating pseudo-labels. For example, Dengyong et al. adopted label propagation methods (Zhou et al., 2003) to generate pseudo-labels (Iscen et al., 2019). Specifically, for every T training iterations, the neural network outputs sample features. Based on these features, label propagation is performed to generate pseudo-labels, which are subsequently utilized for model training. To enhance the accuracy of pseudo-labels, researchers propose employing graph neural networks to generate and select reliable pseudo-labels (Yang et al., 2020).

The re-selection of pseudo samples after numerous training iterations results in incorrect pseudo-labels having a more pronounced impact on the model, ultimately leading to sub-optimal performance.

Consistency-regularization is grounded on the insight that model predictions should remain consistent when a small perturbation is added to inputs. The perturbation can arise from various sources, including data augmentation (Xie et al., 2020), dropout layers (Laine and Aila, 2016), or adversarial training (Miyato et al., 2018; Tarvainen and Valpola, 2017). Subsequently, a perturbed version of the output is generated, and metrics such as mean square error (MSE), cross-entropy, or KL divergence are employed as regularization terms in the loss function.

The pi-model achieves consistency in predictions by feeding the same input sample through a network with different data augmentations, incorporating dropout layers (Laine and Aila, 2016). Subsequently, the model is encouraged to provide consistent predictions for perturbed samples using Mean Squared Error (MSE) loss. In this setup, the network requires two forward passes for the same sample to compute the consistency loss. To enhance computational efficiency, temporal ensembling is introduced, employing a moving average to replace one of the perturbed predictions (Laine and Aila, 2016). This compels the model's predictions of the moving average and the perturbed sample to align. Subsequently, UDA investigated the effectiveness of leveraging a substantial variety of strong data augmentations to enhance the performance of semi-supervised learning (Xie et al., 2020).

Combination of pseudo-labels and consistency-regularization Mixmatch unifies pseudo-labeling and consistency regularization techniques by introducing a sharpening operation (Berthelot et al., 2019). The primary purpose of pseudo-labeling is essentially to encourage the model to make high-confidence predictions on unlabeled samples, known as entropy minimization (Grandvalet and Bengio, 2004; Lee et al., 2013). Inspired by this concept, Mixmatch utilizes a sharpening function to reduce the entropy of the target predictions in the consistency regularization term, thereby achieving the effect of pseudo-labeling, where the target prediction is obtained by averaging multiple model predictions. To improve the accuracy of target predictions

for unlabeled samples, ReMixmatch introduces the concept of augmentation anchoring (Berthelot et al., 2020). Considering that the model's accuracy in predicting weak data augmentations (such as image flips) is higher than for strong data augmentations, the former is utilized as the target prediction for unlabeled samples. FixMatch further integrates pseudo-labeling and consistency regularization by transforming the model's target predictions for unlabeled samples into pseudo-labels and then directly calculating the loss using cross-entropy (Sohn et al., 2020). Additionally, to reduce the interference of incorrect pseudo-labels, only samples with high prediction confidence are used to compute the consistency loss. While these algorithms achieve good performance, they often face challenges in terms of convergence. To tackle this issue, Alphamatch proposes an EM-like optimization framework (Gong et al., 2021). Additionally, Flexmatch introduces a curriculum pseudo-labeling strategy (Zhang et al., 2021).

Recent research has explored the impact of erroneous pseudo-labels on semi-supervised training through various approaches, including modeling the classification head as a Gaussian distribution (Tang et al., 2022), using a confidence estimation blocks (Kim et al., 2022), incorporating counterfactual reasoning techniques (Wang et al., 2022b), and optimizing for worst-case scenarios (Chen et al., 2022). Moreover, other research explored the utilization of reliable low- to mid-confidence unlabeled samples (Deng et al., 2024; Tang and Jia, 2022)

In semi-supervised learning, training the model with pseudo-labels generated from model predictions can lead to over-fitting to its errors. Therefore, successful semi-supervised learning often relies on a sufficient number of labeled samples to gradually correct erroneous pseudo-labels. As the number of labeled samples decreases, the performance of semi-supervised learning experiences a sharp decline due to the lack of an adequate supervisory signal. Also, the issue of limited labels often leads to training instability, which often results in the optimal checkpoint being distant from the final training iteration. A new metric is designed to select high-performing checkpoints, to mitigate the challenge posed by the instability in semi-supervised training (Kim and Lee, 2022).

2.4.3 Semi-Supervised Learning with Self-Supervision

Integrating self-supervised training is a promising choice to improve the performance of semi-supervised training with limited annotations. This is because self-supervised training can provide valuable pre-trained features or self-supervised losses without the need for labeled data. There are three primary categories of methods to leverage self-supervised training. The first and most straightforward way is to propagate labels on self-supervised features directly (Bai et al., 2021). However, self-supervised features are not perfect for a specific task so simple label propagation based on this feature does not produce ideal results.

The second category of methods uses both semi-supervised loss and self-supervised loss to train the model. S4L trains a model with self-supervised loss (predict rotation (Gidaris et al., 2018)) and semi-supervised loss, then re-trains the model with the model's prediction (Zhai et al., 2019). CoMatch adds a low-dimensional embedding for self-supervised loss (Li et al., 2021). Other methods like LESS (Lucas et al., 2022) incorporate self-supervised signals into the semi-supervised training process through online clustering (Gui et al., 2022; Nassar et al., 2023). These methods improve the performance of semi-supervised training when dealing with extremely limited labeled samples. However, when the number of labels is limited, the poor pseudo-labels may affect the representation generated by the model, thereby hindering the effective utilization of self-supervision loss. Moreover, these methods still fail to address the challenge of effectively transferring knowledge from existing pre-trained models to semi-supervised models.

The third category of methods transfers information obtained from self-supervised training primarily relying on weight initialization. After initializing the network with self-supervised training weights, as demonstrated in SelfMatch, the model is trained with conventional semi-supervised methods (Kim et al., 2021a; Xu et al., 2023). EMAN proposes the use of exponential moving averages, including model parameters and batch normalization parameters, as a teacher model (Cai et al., 2021). This improves the performance of semi-supervised training by slowing down the rate at which self-supervised pre-training weights are altered. While self-supervised train-

ing provides commendable initial values for the weights of the backbone, it does not give initial values for classifiers due to the different loss and network architecture. In scenarios with limited labeled samples, poor pseudo-labels at the beginning of semi-supervised training can rapidly disrupt the good initial weights established through self-supervised learning. Therefore, this thesis (Chapter 5) introduces a framework designed to facilitate the effective transfer of knowledge acquired during the self-supervised stage.

2.4.4 Active Semi-Supervised Learning

As discussed in sections 2.3.1 and 2.4.1, active learning and semi-supervised learning represent distinct approaches to optimizing the utilization of unlabeled data, aiming to achieve superior model performance within limited labeling costs. Researchers have naturally endeavored to combine these two paradigms to minimize the demands of labels. Unfortunately, extensive experiments (Mittal et al., 2019) reveal that the deteriorated performance is obtained by combining various existing active learning methods with semi-supervised learning. In some cases, the results are comparable to, or even worse than, that of the semi-supervised learning with randomly selected samples. Furthermore, the strong data augmentations typical in semi-supervised learning, such as rand-augmentation, tend to undermine the effectiveness of existing active learning methods (Kim et al., 2021b; Munjal et al., 2022). This implies that achieving effective cooperation between active learning and semi-supervised learning demands additional exploration.

To bridge this gap, new active learning strategies have been devised. ARAL (Adversarial Representation Active Learning) integrates tightly active and semi-supervised learning (Mottaghi and Yeung, 2019). Its architecture, akin to VAAL (Sinha et al., 2019), utilizes a conditional GAN (cGAN) (Mirza and Osindero, 2014) as a generator, enabling direct use of the auxiliary classifier as the final classification model. It is not just an active learning strategy but rather an active semi-supervised learning framework, where it benefits from labeled, unlabeled, and generated samples. How-

ever, challenges arise in integrating it with existing state-of-the-art consistency-based semi-supervised learning frameworks, leading to subpar performance.

To address this issue, new active strategies are designed to integrate with existing semi-supervised learning frameworks. Consistency-based methods advocate selecting samples challenging for semi-supervised models, i.e., those with inconsistent predictions, for data augmentation (Gao et al., 2020). Results confirm that combining active learning and semi-supervised learning further diminishes sample labeling costs. However, from the wisdom of active learning, some samples chosen in this way may be considered redundant. Jiannan et al. proposes combining adversarial training and graph-based label propagation to select samples near cluster boundaries with high uncertainty (Guo et al., 2021). Additionally, Oriane et al. introduce a novel strategy, joint label propagation, to integrate with semi-supervised learning based on label propagation (Siméoni et al., 2021). This strategy targets unlabeled samples less influenced by existing labeled samples under the semi-supervised learning setting. Nevertheless, experimental results indicate that active semi-supervised learning based on this standard performs weaker than random selection. Subsequent efforts introduce a sample selection strategy based on sample summary (Borsos et al., 2020). This active semi-supervised learning is formulated as a bi-level optimization problem (Borsos et al., 2021), optimizing the selection of labeled samples to ensure the model trained with the chosen samples minimizes generalization loss across all training samples. Experiments reveal that this strategy, when combined with the semi-supervised learning algorithm Mixmatch (Berthelot et al., 2019), yields significantly better results than random selection and consistency-based strategies. However, the bi-level optimization formulation doesn't align well with the fundamental nature of semi-supervised learning, which involves using unlabeled samples for training.

Furthermore, these strategies entail multi-round approaches, incurring an unbearable computational burden in semi-supervised learning scenarios. The considerable training cost associated with multi-round active semi-supervised learning contradicts the overarching goal of minimizing costs in training a powerful model. In contrast, single-shot active learning strategies request all labels at once, providing the poten-

tial benefits of active learning with minimal additional computational burden. This field has received limited attention until now. Several innovative active learning approaches have emerged, focusing on selecting labeled samples based on representations obtained from self-supervised training (Hacohen et al., 2022; Wang et al., 2022a). These approaches select all labeled samples in a single shot, making them easily adaptable to semi-supervised learning contexts. For instance, Xudong et al. propose a diversity-based sampling strategy that selects samples in the self-supervised feature space (Wang et al., 2022a). However, these methods often face challenges related to low accuracy in pseudo-labeling when labeled samples are limited.

2.5 Summary

This chapter discusses the existing methods developed for reducing the need for human annotation: active learning, self-supervised learning fine-tuning framework, and semi-supervised learning. These methods explore the problem from different perspectives, inspiring the idea of combining them pairwise or even integrating all three categories to further minimize the demand for annotations. For example, employing a feature extractor obtained through self-supervised pre-training, applying active learning strategies to select and construct labeled sets, and subsequently fine-tuning the pre-trained model using semi-supervised training techniques. This collaborative approach holds a promise to reduce the need for manual annotation further, benefiting the real-world application of deep learning models, especially in domains where manual annotation is costly.

However, the integration of these methods is not as seamless as anticipated. Recent research has demonstrated that simply combining active learning strategies with self-supervised pre-training or semi-supervised training may not yield benefits and could be detrimental. Active learning strategies face the challenge of phase transition, making it difficult to guarantee gains when the total annotation quantity changes. The combination of self-supervised pre-training and semi-supervised training is relatively natural, but challenges emerge when the benefits of self-supervised pre-training fail to

transfer effectively to the semi-supervised model as the annotation quantity decreases. Therefore, a careful analysis of the unique challenges in combining these techniques and the development of appropriate, effective integration frameworks is worth further exploration.

Additionally, although these three methods explore the problem from different angles, their common foundation is utilizing additional computation to reduce the need for manual annotation. As these technologies progress, it is worthwhile to re-examine whether the cost savings from reduced human annotation still outweigh the significantly increased computational cost.

Chapter 3

NTKCPL: Active Learning on Top of the Self-Supervised Model

3.1 Introduction

Active learning is a path to alleviate the labeling cost by selecting informative subsets of samples to annotate. However, the benefits of active learning have been increasingly questioned in recent years (Mittal et al., 2019; Munjal et al., 2022). One of the main concerns is that training a model initialized by self-supervised learning with randomly selected labeled samples often yields results far beyond those obtained by existing active learning with supervised training (randomly initialized or initialized by the last round of the active learning model) (Chan et al., 2021; Chen et al., 2020a,b; Chen and He, 2021; Grill et al., 2020). Because training on top of the self-supervised model utilizes the information from lots of unlabeled data while training from scratch only accesses labeled data. Therefore, a more practical approach involves implementing active learning on top of the self-supervised models.

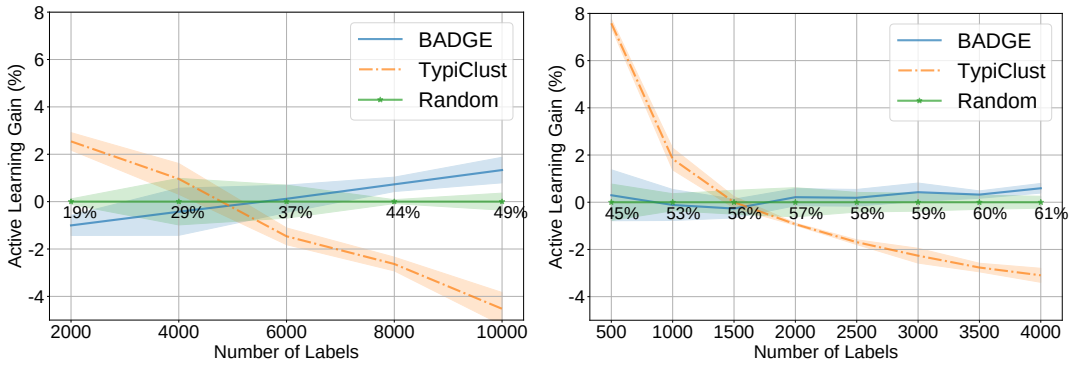
However, combining active learning with self-supervised models is non-trivial. Several studies have highlighted that numerous established active learning methods struggle to outperform the random baseline when combining with self-supervised models (Bengar et al., 2021; Hacohen et al., 2022; Yehuda et al., 2022). Recent research attributes

this issue to budget range (Hacohen et al., 2022). A peculiar phenomenon termed phase transition in active learning has been observed: active learning strategies that beat the random baseline with a high number of samples (high-budget) tend to do worse when labels are limited (low-budget), and vice versa as shown in fig. 3.1a.

In high-budget regimes, effective active learning strategies often prioritize harder samples, such as those with features most dissimilar to the labeled set, as exemplified by Coreset (Sener and Savarese, 2018). Conversely, in low-budget regimes, selecting easier and more representative samples can yield greater active learning gains, as seen in methods like TypiClust (Hacohen et al., 2022). These methods rely on predefined heuristic rules to select samples but fail to account for the influence of labeling budgets on their effectiveness. This oversight limits the transferability of active learning strategies designed for supervised learning settings (high-budget scenarios) to contexts involving self-supervised models (low-budget scenarios).

Most active learning methods are developed and validated in supervised training contexts, where high-budget scenarios dominate. In contrast, self-supervised models achieve strong performance with significantly fewer labeled samples, shifting the focus to low-budget active learning. Motivated by this shift, recent studies have proposed strategies tailored for low-budget scenarios (Hacohen et al., 2022; Yehuda et al., 2022).

However, we discovered that it's not straightforward to categorize scenarios involving self-supervised models as low-budget settings. When training on top of self-supervised models, the phase transition point (where the low-budget active learning strategies cannot outperform random baseline) emerges much earlier than in supervised training scenarios. As illustrated in fig. 3.1, in the training-from-scratch scenario, the phase transition point occurs at approximately 5000 labeled instances, whereas in the self-supervised model scenario, it appears at around 1600 labeled instances. What complicates matters further is that the location of this phase transition point is influenced by various factors, exhibiting significant variations across different datasets (Hacohen and Weinshall, 2024). For new datasets, determining the phase transition point becomes challenging, making it difficult to select effective active learning strategies. Motivated by this observation, we propose an active learning strategy with a wider



(a) without self-supervised Model **(b)** with self-supervised Model

Figure 3.1 – Active learning gain of TypiClust (Hacohen et al., 2022) (designed for the low-budget regime) and BADGE (Ash et al., 2020) (designed for the high-budget regime) on CIFAR-100, both trained with and without self-supervised models. The active learning gain denotes the accuracy difference between the active learning strategy and the random baseline. The shaded area represents standard deviation The training details are shown in sec. 3.5.3.

effective budget range.

We initiated our analysis by exploring the potential reason for the narrow effective budget range within existing low-budget active learning strategies in sec. 3.2, suggesting that this limitation might arise from their inability to precisely estimate the empirical risk of the model of the active learning pool. We then propose improving this problem by explicitly estimating the empirical risk within the active learning pool. We propose a novel active learning strategy, Neural Tangent Kernel Clustering-Pseudo-Labels (NTKCPL), which uses the NTK (Lee et al., 2019; Mohamadi et al., 2023) and CPL to approximate empirical risk on active learning pool in sec. 3.4.1. And we analyze which factor affects approximation error in sec. 3.4.2. Based on this analysis, we design a CPL generation method in sec. 3.4.3. Extensive experimental results in sec. 3.5 demonstrate that our method outperforms state-of-the-art approaches in most cases and has a wider effective budget range. As part of the results (sec. 3.5) we also show our method is effective for self-supervised features of different quality.

Our contribution is summarized as follows: (1) We propose a novel active learning strategy, NTKCPL, by estimating empirical risk on the whole active learning pool based on pseudo-labels. (2) We analyze the approximation error of the empirical risk

in the active learning pool when NTK and CPL are used to approximate networks and true labels. (3) Our method outperforms both low- and high-budget active learning strategies within a range of annotation quantities one order of magnitude larger than traditional low-budget active learning experiments. This means that our approach can be used more confidently for active learning on top of self-supervised models than existing low-budget strategies.

3.2 Insight

The goal of active learning is to identify an unlabeled subset from the active learning pool $D = \{(x_i)\}_{i=1}^{N}$ that leads to the minimized empirical risk over the entire pool when trained with the labels provided by the oracle, as illustrated in eq. 3.1, where θ^* refers to the model parameters trained with labeled set L. This chapter focuses on a neural network model, $f(x;\theta)$, where θ represents its parameters. This model maps input x to the label space of K_y classes.

$$\begin{aligned} & min_{L \subset D} \frac{1}{N} \sum_{(x,y) \in D} Loss(f(x;\theta^*), y) \\ s.t. \quad & \theta^* = argmin_\theta \frac{1}{L} \sum_{(x,y) \in L} Loss(f(x;\theta), y) \end{aligned} \tag{3.1}$$

Unfortunately, calculating this loss is infeasible since labels of the active learning pool are unavailable during active learning. To address this problem, Coreset transforms empirical risk minimization into feature space coverage based on Lipschitz continuity (Sener and Savarese, 2018). A recent active learning method, Probcover, under the assumption of a 1-nearest neighbor classifier and 0-1 loss, further decomposes the approximation error into uncovered error $C(\delta_s)$ and impurity error $\pi(\delta_s)$ as shown in eq. 3.2 (Yehuda et al., 2022). Here, $C(\delta_s)$ refers to the error incurred due to unlabeled data being at a distance greater than δ_s from any labeled sample. $\pi(\delta_s)$ accounts for the error arising from the proximity of unlabeled samples to labeled samples (within a

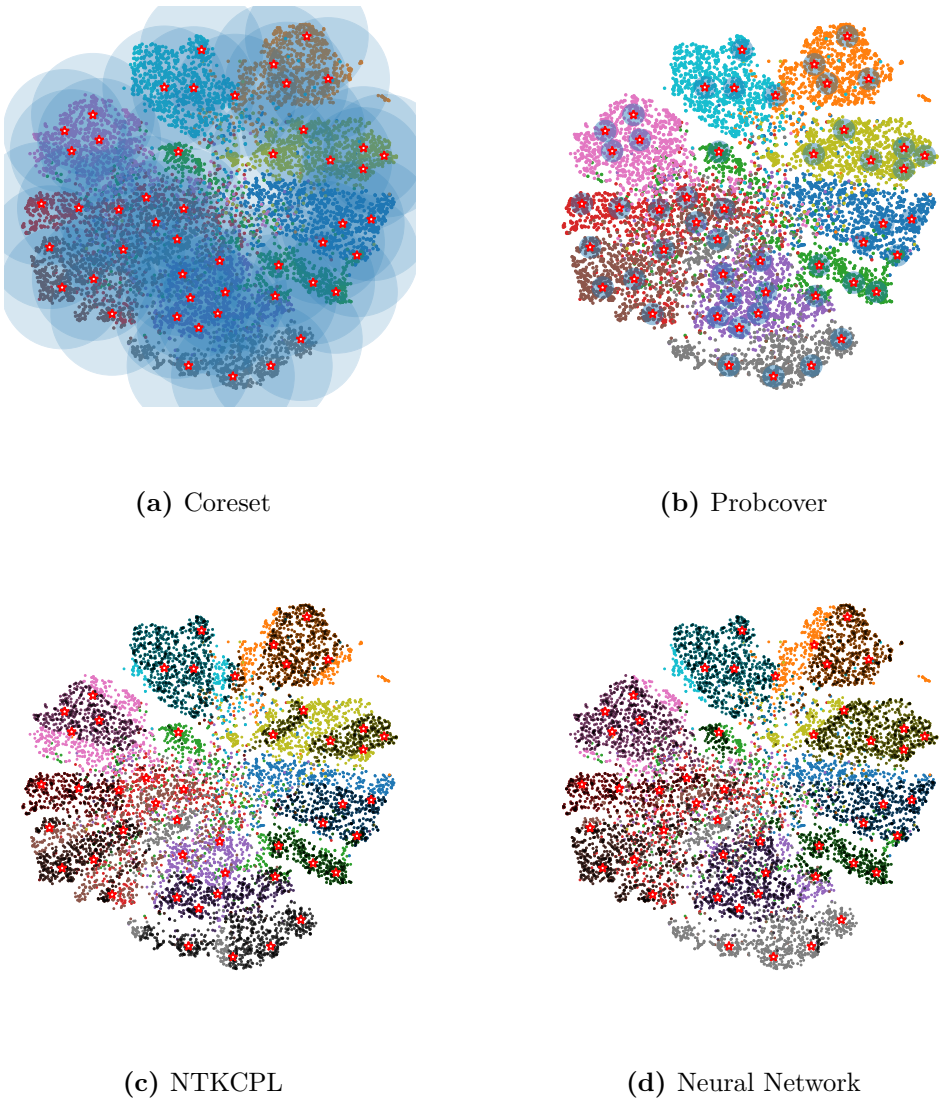
distance of δ_s) but belonging to different categories. As δ_s increases, $C(\delta_s)$ decreases, while $\pi(\delta_s)$ increases.

$$|\frac{1}{N}\sum_{(x,y)\in D} Loss(f(x;\theta^*),y)| \leq C(\delta_s) + \pi(\delta_s) \tag{3.2}$$

Then, these active learning methods minimize the upper bound in eq. 3.2 to minimize empirical risk, thereby selecting samples. To avoid trivial upper bounds, it becomes essential to strike a balance between $C(\delta_s)$ and $\pi(\delta_s)$. Recent research suggests adopting different sample selection strategies based on annotation quantities (Yehuda et al., 2022). In the high-budget regime, assuming full coverage, i.e., $C(\delta_s) = 0$, then samples are selected by minimizing δ_s (as depicted in fig. 3.2a). Applying a high-budget active learning strategy with few labeled samples means that ensuring full coverage requires a very large coverage radius δ_s. This means that a large number of samples belonging to different classes will be covered by the same labeled sample, resulting in significant impurity error $\pi(\delta_s)$. Conversely, in scenarios with limited labeled samples, where uncovered error $C(\delta_s)$ cannot be ignored, a small δ_s is set to ensure a small impurity error $\pi(\delta_s)$. Subsequently, samples are selected by minimizing the uncovered error (as shown in fig. 3.2b).

Remark In this framework, ensuring the effectiveness of active learning requires adjusting δ_s as the number of samples changes. However, the approach to finding a proper δ_s remains uncertain. Furthermore, distance-based active learning methods struggle to precisely estimate empirical risk. For instance, Coreset (Sener and Savarese, 2018) directly minimizes δ_s without determining the accurate variations in empirical risk. Probcover (Yehuda et al., 2022) assumes minimal loss for unlabeled samples within a distance of δ_s from labeled samples, while the loss for the remaining samples is unknown.

To address this issue, we propose using NTK (Neural Tangents Kernel) and pseudo-labels to directly estimate empirical risk, rather than relying on distance-based bounds. Consequently, there is no need to adjust δ_s to strike a balance between uncovered er-



(a) Coreset **(b)** Probcover

(c) NTKCPL **(d)** Neural Network

Figure 3.2 – Visualizing Coverage: The data points are generated through T-SNE of CIFAR-10 self-supervised features (simsiam (Chen and He, 2021)). Each color represents a different category, and labeled samples are marked with a red star. In distance-based active learning strategies like Coreset (Sener and Savarese, 2018) and Probcover (Yehuda et al., 2022), the coverage of unlabeled samples implies that their distance to labeled samples is less than δ_s shown within blue circles. For our approach, NTKCPL, and a real neural network, the coverage indicates that the model's loss for the unlabeled samples is 0. These covered samples are depicted in black. Remarkably, compared to distance-based active learning strategies, NTKCPL's coverage estimation aligns more closely with the true coverage of the neural network.

ror and impurity error. Moreover, our approach allows for a more accurate estimation of empirical risk. As illustrated in fig. 3.2, our method exhibits a higher similarity between regions where the estimated loss is zero and the actual regions with zero loss. The specific details of our method are introduced in sec. 3.4.

3.3 Preliminaries

Neural Tangent Kernel (NTK) is a powerful tool to analyze the training dynamics of neural networks. The neural network is equivalent to the kernel regression with NTK when the network is infinitely wide and its weights are initialized properly (Arora et al., 2019; Jacot et al., 2018). Subsequently, this equivalence has been extended to finite-width networks, in which case the NTK is referred to as the empirical NTK (Lee et al., 2019). The empirical NTK, $\mathcal{K}$, is shown in eq. 3.3, where the f denotes a neural network with parameters θ and $\mathcal{X}$ denotes training samples. In the following text, unless otherwise specified, the empirical NTK will be abbreviated as NTK. When training with the MSE loss, the neural network's prediction for any test sample x in the limit of infinite iterations can be expressed using the NTK as shown in eq. 3.4, where $\mathcal{Y}$ denotes labels of trainset, f_0 denotes the output of network with initialized weights and $\hat{f}(x)$ denotes the output calculated based on NTK.

Additionally, for active learning scenarios, Mohamad proposes the computation time of using NTK can be further reduced by considering the block structure of the matrix (Mohamadi et al., 2022, 2023), which means that look ahead type active learning strategies can be implemented in a reasonable amount of time. For example, if we want to use the look-ahead active learning strategy, each active learning cycle takes 3 hours to train the entire network of 15 epochs on the MNIST dataset, while it takes only 3 minutes to use NTK with a block structure (Mohamadi et al., 2022).

$$\mathcal{K}(\mathcal{X}, \mathcal{X}) = \nabla_\theta f(\mathcal{X}) \nabla_\theta f(\mathcal{X})^T \tag{3.3}$$

$$\hat{f}(x) = f_0(x) + \mathcal{K}(x, \mathcal{X})\,\mathcal{K}(\mathcal{X}, \mathcal{X})^{-1}(\mathcal{Y} - f_0(\mathcal{X})) \tag{3.4}$$

3.4 Method

3.4.1 Framework

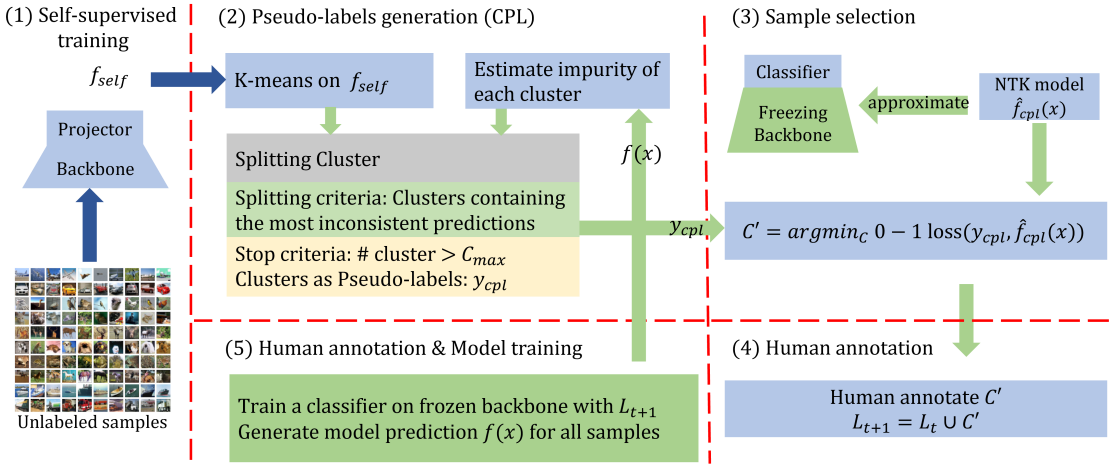


Figure 3.3 – The framework of NTKCPL. Before starting active learning, we perform self-supervised pre-training using all unlabeled data to obtain pre-trained features f_{self}. Subsequently, pseudo-labels are generated through clustering. After the first round of active learning, the pseudo-labels are refined using model predictions (sec. 3.4.3). We then combine the pseudo-labels with NTK to select samples that minimize the empirical risk of the model in the active learning pool (alg. 1). Finally, samples selected by the oracle are annotated and used for model training.

We propose a look ahead strategy, NTKCPL, to approximate the empirical risk on the whole active learning pool directly. The framework is shown in fig. 3.3. There are two challenges: (1) estimate empirical risk without true labels and (2) estimate predictions of models trained with a large amount of possible candidate sets efficiently and accurately.

For the first challenge, clusters on self-supervised features provide good pseudo-labels. Because most samples in the same cluster have the same label (Wen et al., 2025). And

when the number of clusters is increased, it can improve the purity of clusters, where purity refers to the probability that the sample has the same label within the same cluster. We refer to these K_{cpl} clusters as clustering-pseudo-labels (CPL), denoted as y_{cpl}.

For the second challenge, as introduced in sec. 3.3, NTK approximates the network well for random initialization and the computation time is acceptable. However, the study has shown that the quality of NTK approximation for finite-width neural networks depends on factors such as the depth-to-width ratio and the standard deviation of parameter initialization. It further indicates that NTK provides a good approximation primarily for wide and shallow networks or for deep networks in the ordered phase (Seleznova and Kutyniok, 2022). Moreover, recent research demonstrates that using a shallow MLP classifier as a proxy model, with pre-trained features as input, can achieve active learning performance comparable to that of sample selection based on fully fine-tuned models (Zhang et al., 2024). Based on these insights, we utilize NTK to approximate the shallow MLP classifier instead of the entire pre-trained model. Specifically, the inputs of NTK are self-supervised features, $feas_{self}$, i.e., the output of the backbone initialized with the self-supervised weights. This ensures both a reliable NTK approximation and reduced computation time for NTK. Accordingly, we choose a training method following (Mahmood et al., 2022) that freezes the encoder initialized by self-supervised learning and trains only the MLP as a classifier. That training method achieves better or equal performance than fine-tuning the whole network in the low-budget case as shown in sec. 3.5.5 while its prediction is more consistent with the results of NTK. We denote the prediction of sample x by the NTK using CPL, y_{cpl}, as $\hat{f}_{cpl}(x; L)$. For simplicity, we will omit L in the following text. Now, the active learning goal in eq. 3.1 is approximated as eq. 3.5.

$$min_{L \in D} \frac{1}{N} \sum_{(x,y) \in D} Loss(\hat{f}_{cpl}(x), y_{cpl}) \tag{3.5}$$

The algorithm is shown in Alg. 1. For computational simplicity and without loss of generality, we use 0-1 loss to calculate empirical risk in eq. 3.5. In each round of active

learning, after computation of NTK based on eq. 3.3 and generation of CPL based on the method introduced in sec. 3.4.3, the sample that minimizes the empirical risk on the whole active learning pool after adding labeled set is selected.

Algorithm 1 NTKCPL

Input: self-supervised feature $feas_{self}$, active learning feature $feas_{al}$ labeled set L, unlabeled set U, budget b, initial cluster number C_0, maximum cluster number C_{max}, neural network's prediction $f(x)$ at the last active learning round
Output: new labeled set L
$N_{clu} = min\{b/2, C_{max}\}$
$y_{cpl} \leftarrow$ CPL generation($feas_{al}$, $f(x)$, C_0, N_{clu}, L) based on Alg. 2
Initialize CPL labeled set $L_{cpl} = \{(x, y_{cpl}) | x \in L\}$
for $itr = 1$ **to** b **do**
    $Emp\_risk = []$
    for $(x_i, y_{cpl,i})$ **in** U **do**
        Compute NTK prediction on U, $\hat{f}_{cpl}(U)$, based on eq. 3.4
        Emp_risk += [0-1$Loss(\hat{f}_{cpl}(U), y_{cpl}]$
    end for
    $i^* = argmin Emp\_risk$
    $L_{cpl} = L_{cpl} \cup (x_{i^*}, y_{cpl,i^*})$, $U = U \backslash x_{i^*}$
end for
Query label y_{i^*} of selected batch samples x_{i^*} from oracle
return $L = L \cup (x_{i^*}, y_{i^*})$

3.4.2 Approximation Error

In this section, we analyze what affects the accuracy of NTKCPL estimates of empirical risk on the whole active learning pool. The approximation error is the difference between the true empirical risk and the estimated empirical risk using NTK and CPL. The approximation error is given by:

$$\frac{1}{N} \sum_{(x,y) \in D} \left| Loss(f(x), y) - Loss(\hat{f}_{cpl}(x), y_{cpl}) \right|$$

$$\leq \frac{1}{N}\sum_{(x,y)\in D}(\Big|Loss(f(x),y) - Loss(\hat{f}_y(x),y)\Big| + \Big|Loss(\hat{f}_y(x),y) - Loss(\hat{f}_{cpl}(x),y_{cpl})\Big|) \tag{3.6}$$

As shown in eq. 3.6, it is bounded by the sum of NTK and CPL error, where the NTK error represents the first term on the right-hand side of eq. 3.6, corresponding to the difference between the predictions of the NTK model using true labels, $\hat{f}_y(x)$, and the neural network, $f(x)$. The CPL error refers to the second term on the right-hand side of eq. 3.6, which is the difference caused by CPL during NTK estimation, where $\hat{f}_{cpl}(x)$ denotes the predictions of NTK with the CPL. For the NTK error, as we mentioned in sec. 3.4.1, NTK is used to approximate the classifier only to obtain better consistency.

When the NTKCPL method adopts the 0-1 loss, the approximation error from CPL, previously defined in eq. 3.6, can be further expressed as eq. 3.7 by substituting the indicator function $\mathbb{I}$ into the prior equation. It only takes non-zero values in two scenarios. First, when $argmax\ \hat{f}_y(x) = y$ and $argmax\ \hat{f}_{cpl}(x) \neq y_{cpl}$, denoting the probability of this occurrence as P_{fnf}. Second, when $argmax\ \hat{f}_y(x) \neq y$ and $argmax\ \hat{f}_{cpl}(x) = y_{cpl}$, denoting the probability of this event as P_{nff}. Thus, CPL error expands to eq. 3.8. To shed light on the circumstances giving rise to P_{nff} or P_{fnf}, we elucidate the relationship between NTK predictions when using CPL and those when using true labels in the following proposition.

$$error_{CPL} = \frac{1}{N}\sum_{(x,y)\in D}\Big|\mathbb{I}(argmax\ \hat{f}_y(x) = y) - \mathbb{I}(argmax\ \hat{f}_{cpl}(x) = y_{cpl})\Big| \tag{3.7}$$

$$error_{CPL} = P_{nff} + P_{fnf} \tag{3.8}$$

Proposition If the true labels of labeled samples are the dominant labels in their corresponding CPL clusters, $\hat{f}_y(x) = g(\hat{f}_{cpl}(x))$, where g is the label mapping function, defined as follows.

Definition Label mapping function, g, converts NTK's predictions about CPL classes, $\hat{f}_{cpl}(x)$, into predictions about true classes based on dominant labels within the corresponding CPL clusters as shown in eq. 3.9, where $\hat{f}^j_{cpl}$ and $g(\hat{f}_{cpl}(x))^j$ denote the j^{th} outputs of $\hat{f}_{cpl}$ and g, respectively, and K_{cpl} is the total number of categories in the CPL. The dominant class of cluster k refers to the category of the true label that has the most count among the samples within cluster k.

$$g(\hat{f}_{cpl}(x))^j = \sum_{k=1}^{K_{cpl}} \mathbb{I}_{j,k} \hat{f}^k_{cpl}(x) \tag{3.9}$$

$$\mathbb{I}_{j,k} = \begin{cases} 1, & j \text{ is dominant class of } k \\ 0, & otherwise \end{cases} \tag{3.10}$$

Proof of Proposition When the training samples is $\mathcal{X}$, the prediction of NTK for the test sample x, $\hat{f}(x)$, is as in eq. 3.11. Here, $\mathcal{Y}$ represents the true labels of training samples in one-hot encoding form, and the i^{th} row of $\mathcal{Y}$ corresponds to the one-hot form true label of the i^{th} training sample. We denote $\mathcal{K}(x,\mathcal{X})\mathcal{K}(\mathcal{X},\mathcal{X})^{-1}$ as weight vector W.

$$\begin{aligned} \hat{f}(x) &= f_0(x) + \mathcal{K}(x,\mathcal{X})\,\mathcal{K}(\mathcal{X},\mathcal{X})^{-1}(\mathcal{Y} - f_0(\mathcal{X})) \\ &= f_0(x) - W f_0(\mathcal{X}) + W\mathcal{Y} \end{aligned} \tag{3.11}$$

The difference between NTK predictions with true labels, $\hat{f}_y(x)$, and predictions with CPL transformed to true label classes through a label mapping function g, $g(\hat{f}_{cpl}(x))$, is shown in eq. 3.12, where $\mathcal{Y}_{cpl}$ denotes CPL of training samples with one-hot form. We write the product of the row vector W and the labels as the element-wise multiplication, where w_i represents the i^{th} element of W, y_i and $y_{cpl,i}$ are the i^{th} rows of the label matrix $\mathcal{Y}$ and the $\mathcal{Y}_{cpl}$, respectively. By the definition of label mapping function, g, when true labels of labeled samples are the dominant

classes in their corresponding CPL clusters, $y_i = g(y_{cpl,i})$, the result of eq. 3.12 is zero, i.e., $\hat{f}_y(x) = g(\hat{f}_{cpl}(x))$.

$$\begin{aligned} \hat{f}_y(x) - g(\hat{f}_{cpl}(x)) &= W\mathcal{Y} - g(W\mathcal{Y}_{cpl}) \\ &= \sum_{i \in L}(w_i y_i) - \sum_{i \in L}(w_i g(y_{cpl,i})) \\ &= \sum_{i \in L} w_i (y_i - g(y_{cpl,i})) \end{aligned} \tag{3.12}$$

Explanation of Impurity Error and Over-clustering Error According to the proposition, we argue $argmax\hat{f}_y(x)$ is most likely to be equal to $g(onehot(argmax\hat{f}_{cpl}(x)))$, i.e., the prediction of NTK model with the true labels is most likely equal to the corresponding dominant class of the prediction of the NTK model with CPL. The empirical evidence is shown in sec. 3.5.5.

For the case of P_{fnf}, as derived in eq. 3.13, the true label y is the dominant class of the CPL cluster $argmax\hat{f}_{cpl}(x)$, where the first equality is obtained based on the definition of P_{fnf} and the second equality is derived from the argument based on the proposition. Also, according to the definition of P_{fnf}, $y_{cpl} \neq argmax\hat{f}_{cpl}(x)$, indicating that this true label class consists of at least two different CPL clusters, namely y_{cpl} and $argmax\hat{f}_{cpl}(x)$. In other words, this true class has been over-clustered.

$$\begin{aligned} y &= argmax\hat{f}_y(x) \\ &= g(onehot(argmax(\hat{f}_{cpl}(x)))) \end{aligned} \tag{3.13}$$

For the case of P_{nff}, as shown in eq. 3.14, the true label y is not the dominant class of the y_{cpl}, where the first equality is obtained based on the definition of P_{nff}, the second equality is derived from the argument based on the proposition, and the third equality is obtained based on the definition of P_{nff}. Thus, the true label y is not the dominant class of y_{cpl}. This implies that the samples within this CPL cluster include

at least two different true labels, namely y and the dominant class of y_{cpl}. In other words, this CPL cluster is impure.

$$\begin{aligned} y &\neq argmax\hat{f}_y(x) \\ &= g(onehot(argmax(\hat{f}_{cpl}(x)))) \\ &= g(y_{cpl}) \end{aligned} \tag{3.14}$$

3.4.3 Clustering Pseudo-Label

As deduced from the previous analysis, the effect of CPL on the approximation error comes from the purity of the clusters and over-clustering. To improve clustering purity, we take two approaches: (1) clustering on the active learning feature, $feas_{al}$, i.e., the output of the penultimate layer of the classifier, and (2) increasing the number of clusters. However, increasing the number of clusters may cause the labeled samples not to cover all classes of the CPL (under-coverage) and also increase the risk of over-clustering error.

To improve the under-coverage, we set the number of clusters to half of the total number of labels, i.e., each cluster includes two labeled samples on average. To improve the over-clustering, we design a clustering-splitting approach instead of directly increasing the number of clusters. It splits the low-purity clusters and keeps the high-purity ones to reduce the extra over-clustering errors within samples located in the high-purity clusters. Specifically, we use the prediction of the neural network in each round of active learning to estimate the number of confusing samples within each cluster, i.e., the number of samples from classes that are different from the dominant class. The clusters that contain the largest number of confusing samples are split sequentially until a predefined number of clusters is reached. The cluster splitting algorithm is shown in Alg. 2, where we adopt the constrained K-Means (Wagstaff et al., 2001) to improve the clusters from labeled sample constraints.

Algorithm 2 CPL generation

Input: active learning feature $feas_{al}$, neural network predictions $f(x)$, initial cluster number C_0, maximum cluster number C_{max}, labeled set L
Output: CPL y_{cpl}
Initialize $Clus$ with C_0 clusters on $feas_{al}$ using Constrained K-means
for $i = 1$ to $C_{max} - C_0$ **do**
    Find the cluster, $Clus_i$, that contains the most confusing samples
    Split $Clus_i$ into two using K-means, resulting in new clusters $Clus_{n1}$, $Clus_{n2}$
    Replace $Clus_i$ with $Clus_{n1}$, $Clus_{n2}$
end for
return $y_{cpl} \leftarrow Clus$

3.4.4 Computational Complexity

The computational complexity of our algorithm can be broken down into two main components. The first component is the generation of CPL as described in Algorithm 2 of this chapter and the second component involves the utilization of NTK approximations. Let's denote the size of the labeled dataset as L, the size of the unlabeled dataset as U, the budget for selecting labeled samples per active learning round as b, and the number of parameters in the model as P (in our case, the parameters of the MLP classifier used in NTK computation).

Referring to (Mohamadi et al., 2022), we can write that the complexity of computing the NTK kernel is $O(LUP + L^2P + L^3)$. The time complexity for making predictions using the NTK for a single sample is $O(UL^2)$. Consequently, the time complexity for computing the NTK kernel and selecting b samples is approximately $O(b(LUP + L^2P + L^3 + UL^2))$.

Regarding the CPL computation, the primary computational complexity arises from the execution of k-means clustering $(C_{max} - C_0)$ times, where we split the most impure cluster into two, resulting in $k = 2$. Here, C_{max} represents the maximum number of clusters, C_0 is the initial number of clusters, the maximum number of iterations for k-means is denoted as I, and the feature dimension used for clustering is represented by d. As shown in Algorithm 1, each active learning round involves

generating CPL once and the selection of b samples, resulting in an overall time complexity of $O(b(LUP+L^2P+L^3+UL^2)+UdI)$. In our scenario, the dominant term is $O(bLUP)$, which is the same as the NTK-based LookAhead framework (Mohamadi et al., 2022).

3.5 Results

Our approach is validated on five datasets with various qualities of self-supervised features. Datasets with good self-supervised features, such as CIFAR-10 (Krizhevsky et al., 2009), CIFAR-100 (Krizhevsky et al., 2009), and ImageNet-100 (a subset of ImageNet (Deng et al., 2009), following splitting in (Van Gansbeke et al., 2020)), are included. SVHN (Netzer et al., 2011) with poor self-supervised features is also included. Additionally, we consider practical scenarios where the total number of samples in the trainset is insufficient to support effective self-supervised training, such as Oxford-IIIT Pet dataset (Parkhi et al., 2012). In this case, we evaluated the effectiveness of our method based on the model pre-trained on ImageNet (Deng et al., 2009). To further demonstrate the effectiveness of our method with different features for generating CPL, we report results based on both pre-trained features and the penultimate layer outputs of a trained classifier, denoted as NTKCPL(self) and NTKCPL(al), respectively. Detailed analyses can be found in Sec. 3.5.5.

3.5.1 Datasets

CIFAR-10 and CIFAR-100 datasets (Krizhevsky et al., 2009) both consist of 50,000 training samples and 10,000 testing samples, with a resolution of 32x32. CIFAR-10 is composed of 10 classes, while CIFAR-100 consists of 100 classes. Similarly, SVHN dataset (Netzer et al., 2011) also contains 10 classes, with 73,257 training samples and 26,032 testing samples. Its resolution is 32x32. ImageNet-100 is a subset of the ImageNet dataset, following splitting in (Van Gansbeke et al., 2020), comprising 100 classes, with 128,545 training samples and 5,000 testing samples. Oxford-IIIT

Pet (Parkhi et al., 2012) dataset includes 37 classes, with 3,680 training samples and 3,669 testing samples. For both ImageNet-100 and Oxford-IIIT Pet dataset, all samples were resized to a resolution of 224x224 following the method described in (Grill et al., 2020).

3.5.2 Baseline

We compare our proposed method with representative active learning strategies: (1) Random, (2) Entropy (uncertainty sampling, maximum entropy of output) (Lewis and Catlett, 1994), (3) Coreset (diversity active learning strategy, greedy solution of minimum coverage radius) (Sener and Savarese, 2018), (4) BADGE (combination of uncertainty and diversity, kmeans++ sampling on grad embedding) (Ash et al., 2020), where the scalable version (Citovsky et al., 2021; Emam et al., 2021), badge partition, is used in ImageNet-100, CIFAR-100 and Oxford-IIIT Pet because the huge dimension of grad embedding, (5) Typiclust (Hacohen et al., 2022) and Probcover (Yehuda et al., 2022) (designed for low-budget case), (6) Lookahead (maximum output change based on NTK) (Mohamadi et al., 2022).

3.5.3 Implementations

As our method focuses on the low-budget regime, we followed the training method in (Mahmood et al., 2022), freezing weights of the backbone initialized with self-supervised learning and then training a MLP as the classifier. The hyper-parameters for training are set following (Hacohen et al., 2022) and are shown in table 3.1. For the self-supervised model, we adopt simsiam (Chen and He, 2021) for CIFAR-10, CIFAR-100 and SVHN and BYOL (Grill et al., 2020) for ImageNet-100 and Oxford-IIIT Pet. Resnet-18 (He et al., 2016) is used in CIFAR-10 and SVHN, WRN28-8 (Zagoruyko and Komodakis, 2016) is used in CIFAR-100 and Resnet-50 (He et al., 2016) is used in ImageNet-100 and Oxford-IIIT Pet.

We roughly set the maximum number of clusters C_{max} to be around 3-5 times the number of classes of the dataset. For datasets with a larger number of samples per

Table 3.1 – Hyperparameters

Dataset	Backbone	Classifier	Learning Rate	Momentum	Weight Decay	Batch Size
CIFAR-10	Resnt18	MLP64	0.3	0.9	0.0003	100
CIFAR-100	WRN-28-8	MLP512	0.3	0.9	0.0003	100
SVHN	Resnt18	MLP512	0.3	0.9	0.0003	100
ImageNet-100	Resnet50	MLP4096	0.1	0.9	0	512
Oxford-IIIT Pet	Resnet50	MLP512	0.1	0.9	0	128

class (CIFAR-10 and SVHN have thousands of samples per class), we increase C_{max} to around 10 times the number of classes in the dataset. Specifically, for CIFAR-10, CIFAR-100, ImageNet-100, SVHN, and Oxford-IIIT Pet, the maximum number of clusters is 100, 500, 300, 100, and 150, respectively. We followed (Mohamadi et al., 2022) to sample a subset of the unlabeled set as the candidate set to select samples and estimate coverage. The candidate set includes 10,000 samples.

For the query step, we set 500 for CIFAR-100, 20 for SVHN, and 40 for Oxford-IIIT Pet. The varying query step is set for CIFAR-10 and ImageNet-100. Specifically, for CIFAR-10, 20 samples are queried before 100 labels are available, 100 samples are queried before 500 labels and 500 labels are queried before 2000 labels. For ImageNet-100, 200 samples are selected before 1000 labels are available and 500 samples are queried before 2000 labels.

Classifier is a 2-layer MLP with the architecture: Linear + BatchNorm + ReLU + Linear. The dimension of the output of the first linear layer is shown in table 3.1. Total number of training epochs is 100. Data augmentation includes random crops and horizontal flips. The remaining hyper-parameters are shown in table 3.1. The data, training method, and model architecture used for pre-training the model are detailed in table 3.2.

3.5.4 Main Experiment Results

All experiments were run 5 times and the average value and standard deviation are reported. Considering that the experiments are conducted for the scenario with a low

Table 3.2 – The data, training method, and model architecture used for pre-training the model.

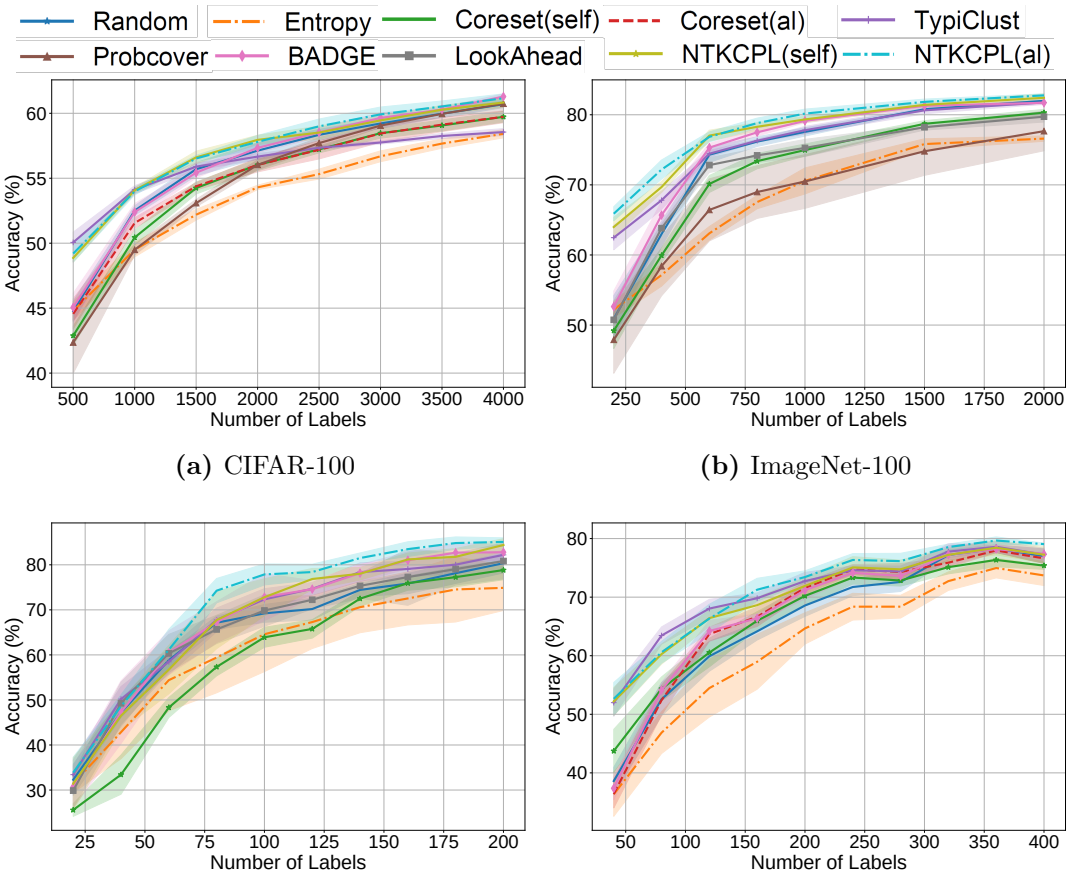
Dataset	Backbone	Pre-train dataset	Pre-train method
CIFAR-10	Resnt18	CIFAR-10	Simsiam
CIFAR-100	WRN-28-8	CIFAR-100	Simsiam
SVHN	Resnt18	SVHN	Simsiam
ImageNet-100	Resnet50	ImageNet	BYOL
Oxford-IIIT Pet	Resnet50	ImageNet	BYOL

annotation budget, it is not practical to construct a validation set to select the best checkpoints (the benefits of constructing a validation set are much less than using these labeled samples as training samples). Therefore, we report the final checkpoint accuracy, not the accuracy of the checkpoints determined by the validation set. The results are shown in fig. 3.4 and table 3.3.

# Labels	*Random*	*Entropy*	*Coreset(self)*	*BADGE*	*TypiClust*	*LookAhead*	*NTKCPL(self)*	*NTKCPL(al)*
20	41.80±3.82	38.58±2.86	20.08±2.75	39.85±3.91	46.38±1.61	40.93±4.04	54.31±3.74	52.67±3.70
40	57.52±3.34	51.10±4.21	36.67±6.29	54.99±3.43	66.18±2.45	58.55±2.71	68.60±2.50	63.55±2.89
60	65.88±3.07	64.46±3.42	46.39±7.41	65.23±1.40	72.93±1.77	66.96±2.90	75.09±1.69	72.22±2.11
80	69.35±3.31	70.49±3.05	58.96±6.15	70.76±1.86	76.98±1.04	72.71±1.94	78.51±1.61	75.32±0.92
100	74.11±1.16	74.34±1.92	62.64±5.07	75.40±0.99	78.24±1.28	75.97±2.04	80.30±1.17	78.45±1.19
200	80.90±0.90	79.86±1.77	76.93±3.56	82.20±1.14	83.16±0.61	81.89±1.31	83.77±1.04	81.87±1.02
300	82.80±0.93	81.43±2.23	82.64±1.42	84.53±0.46	84.16±0.25	83.29±0.89	85.00±0.54	83.78±1.05
400	84.04±0.49	83.37±1.31	84.56±1.15	84.75±0.40	85.13±0.27	84.59±0.59	85.64±0.38	84.73±0.85
500	84.97±0.78	84.24±0.89	85.23±0.59	85.57±0.51	85.37±0.15	85.31±0.12	85.72±0.22	85.48±0.65
1000	86.26±0.38	84.94±0.48	86.75±0.36	86.06±0.31	86.07±0.14	85.69±0.47	86.83±0.33	87.15±0.57
1500	86.95±0.27	85.85±0.39	87.03±0.13	87.05±0.36	86.37±0.11	86.82±0.23	87.18±0.41	87.58±0.29
2000	87.30±0.37	86.92±0.15	87.34±0.27	87.31±0.47	86.55±0.21	87.16±0.19	87.34±0.41	87.87±0.39

Table 3.3 – Comparison of accuracy of different active learning strategies on CIFAR-10. All results are averages over 5 runs. The best results are shown in red and the second-best results are shown in blue.

NTKCPL outperforms SOTA. As shown in table 3.3, fig. 3.4. In most cases, our proposed method outperforms the baseline methods. For the few cases with only a small number of labels, our method shows comparable performance with the low-budget strategy, TypiClust, such as in CIFAR-100 with 500 and 1000 labeled samples, and Oxford-IIIT Pet with 80 and 100 labeled samples.



(a) CIFAR-100 **(b)** ImageNet-100

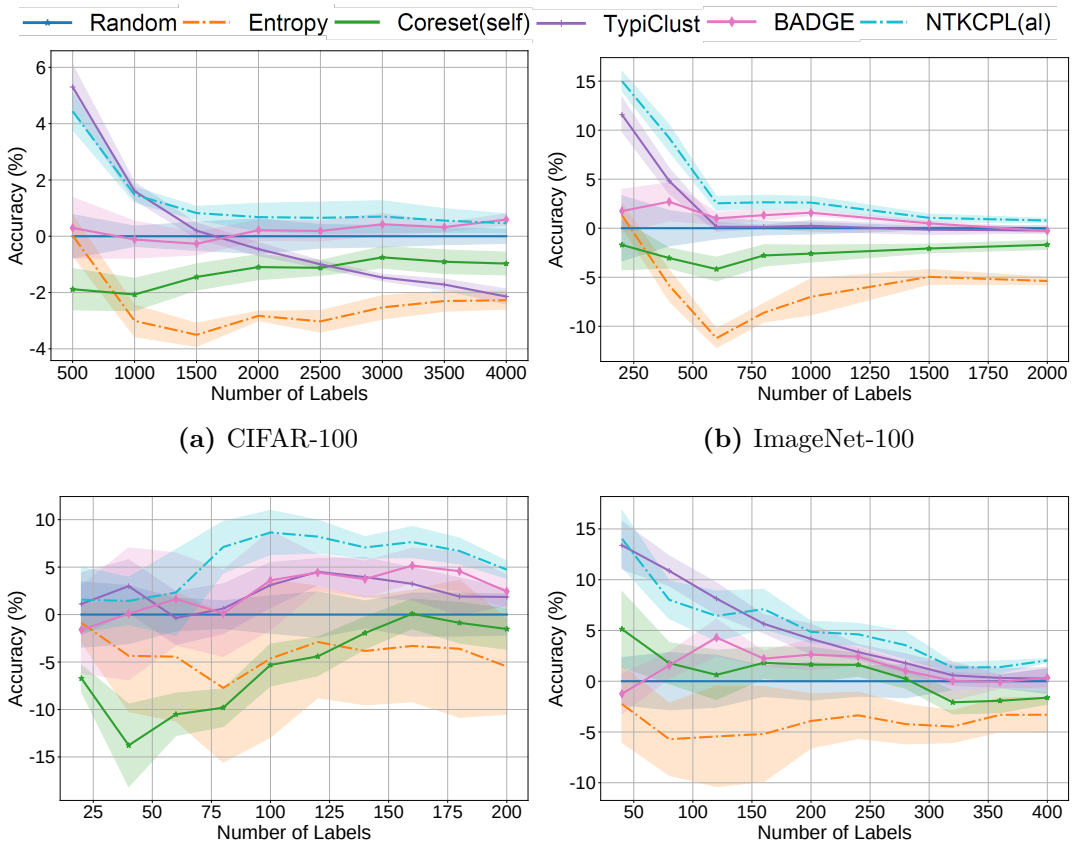
(c) SVHN **(d)** Oxford-IIIT Pet

Figure 3.4 – Performance of different active learning strategies. The shaded area represents standard deviation.

NTKCPL still shows good performance when the self-supervised features do not correspond well to the label classes. Since the loss of self-supervised training is different from that of classification, self-supervised features do not always align well with label classes. In the SVHN dataset, self-supervised features of different classes are mixed because the images include some irrelevant digits on both sides of the digit of interest (Netzer et al., 2011). In this case, our method is similar to other baseline strategies at the beginning of active learning, but it shows better results than baselines after several active learning rounds as shown in fig. 3.4c.

Another common scenario is the lack of sufficient samples to support effective self-supervised training. To evaluate in this context, we choose the Oxford-IIIT Pet dataset with the self-supervised model trained on ImageNet. The result is shown in

fig. 3.4d. Our method has similar accuracy in the first three rounds as the TypiClust and outperforms all baseline methods afterward.



(a) CIFAR-100 **(b)** ImageNet-100

(c) SVHN **(d)** Oxford-IIIT Pet

Figure 3.5 – Active learning gain of different active learning strategies. The shaded area represents the standard deviation.

NTKCPL has a wider effective budget range than SOTA. Active learning based on self-supervised models exhibits an intensified phase transition phenomenon, where low-budget strategies underperform the random baseline when annotation quantities are limited. We plot the active learning gain of our method and baselines on different datasets in fig. 3.5. The average accuracy of our method, NTKCPL(al), outperforms the random baseline at all quantities of labels. In contrast, both the typical high-budget strategy, BADGE, and low-budget strategy, TypiClust, appear to be worse than the random baseline over a range of annotation quantities.

3.5.5 Detailed Study

In this section, we conduct a comprehensive investigation of the NTKCPL method, encompassing the accuracy of NTKCPL in estimating true coverage, the impact of CPL's quality, and the impact of maximum cluster quantity C_{max} on performance. Also, we compare the effect of generating CPL on self-supervised features as well as on the active learning feature on the performance of NTKCPL and the practicality of training methods.

Empirical Risk Estimation To evaluate the effectiveness of our proposed method, NTKCPL, in estimating the empirical risk on the entire active learning pool, experiments were conducted on CIFAR-100. The empirical risk was computed using 0-1 loss. As shown in fig. 3.6, when there are few labeled samples, the inaccurate CPL may lead to a large estimation error for empirical risk error. However, after reaching about 1500 labels, our approach accurately estimates the empirical risk over the entire active learning pool.

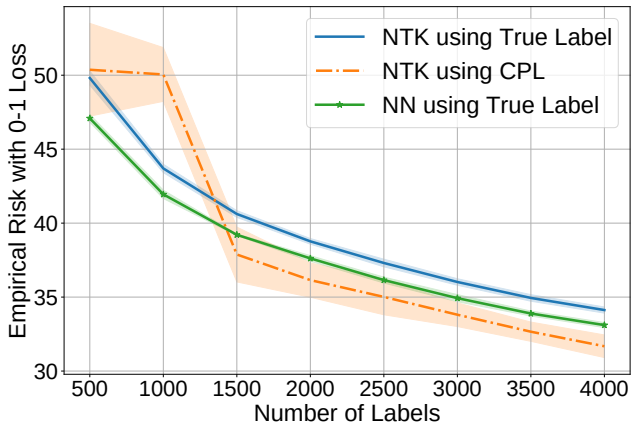


Figure 3.6 – Estimation of empirical risk on the whole active learning pool for CIFAR-100, where the NN using true labels represents the ground truth.

Ablation Study To investigate the contributions of each component in the NTKCPL method, we conducted ablation experiments on the ImageNet-100 dataset. Specifically, we compared the performance when using linear regression versus NTK to

approximate outputs of multi-layer neural networks, and when employing simple K-Means clustering versus CPL as pseudo-labels to guide active learning sample selection. The results are presented in fig. 3.7.

Whether CPL or K-Means is used as pseudo-labels, NTK consistently leads to significant performance improvements over linear regression. This suggests that NTK's ability better to approximate the outputs of a multi-layer neural network is crucial for look-ahead active learning. Moreover, CPL exhibits notable performance gains over K-Means only when paired with the NTK model. This indicates that the benefits of pseudo-labels rely on the NTK's capability to approximate neural network outputs effectively. Additionally, as the number of labeled samples increases, the advantage of NTKCPL over the NTK using K-Means for pseudo-labeling becomes increasingly evident. This observation aligns with our analysis in sec. 3.4.2, which shows that using K-Means clusters as pseudo-labels struggles to balance impure errors and over-clustering errors, thereby reducing its effectiveness in approximating the true empirical risk. In contrast, our proposed CPL method effectively mitigates these issues, resulting in superior performance.

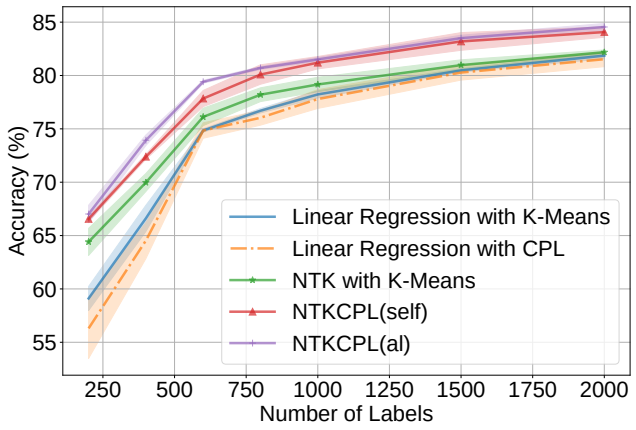


Figure 3.7 – The ablation study on ImageNet-100.

The Impact of the CPL To gauge the influence of CPL quality, we conducted experiments replacing CPL with the true labels. The outcomes are detailed in table 3.4 and table 3.5. There's still room for enhancing CPL quality, especially when working

with a limited number of labels. As the label count grows, NTKCPL progressively converges towards that of NTK using true labels.

Table 3.4 – Accuracy of NTK with true label on CIFAR-100

#Label	NTK with true label	NTKCPL(self)	NTKCPL(al)
500	55.04±0.42	48.87±0.33	49.20±0.66
1000	57.99±0.27	54.08±0.23	53.99±0.27
1500	59.48±0.51	56.61±0.48	56.52±0.24
2000	60.42±0.26	57.95±0.35	57.80±0.49
2500	60.66±0.30	58.52±0.50	59.00±0.56
3000	61.16±0.22	59.46±0.32	59.92±0.57
3500	61.70±0.11	60.27±0.33	60.53±0.42
4000	62.01±0.24	60.83±0.35	61.16±0.34

Table 3.5 – Accuracy of NTK with true label on ImageNet-100

#Label	NTK with true label	NTKCPL(self)	NTKCPL(al)
200	74.59±1.12	63.98±1.39	65.88±1.00
400	78.13±0.68	69.69±0.78	72.18±1.36
600	80.70±0.27	77.03±0.77	76.84±0.72
800	81.38±0.56	78.28±0.58	78.81±0.71
1000	82.09±0.32	79.34±0.44	80.17±0.64
1500	83.19±0.23	81.41±0.31	81.84±0.40
2000	83.67±0.30	82.42±0.56	82.77±0.21

Effect of the Maximum Number of CPL The ablation experiments are conducted on CIFAR-10. We plot the accuracy when the number of annotations selected by active learning is greater than 400 as shown in fig. 3.8. In this range, the number of classes of CPL is fixed at 10, 50, 100, and 200, respectively. The results support our analysis in sec. 3.4.2 that too many or too few clusters will increase the approximation error, which affects the performance of active learning.

Effect of self-supervised feature-based and active learning feature-based clustering-pseudo-labels on NTKCPL. We denote NTKCPL based on active learning features as NTKCPL(al) and NTKCPL based on self-supervised learning features as NTKCPL(self). The results are shown in table 3.3 and fig. 3.4. From

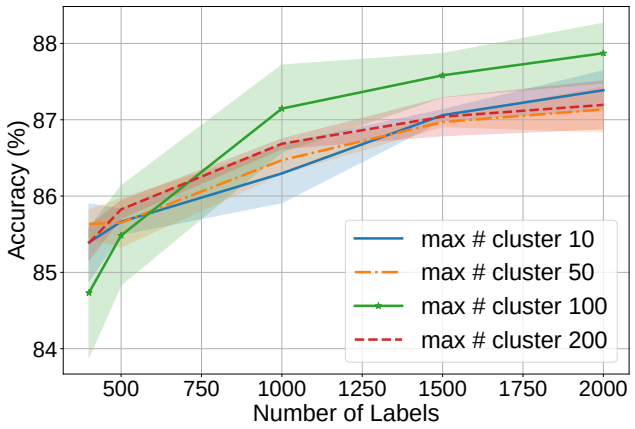


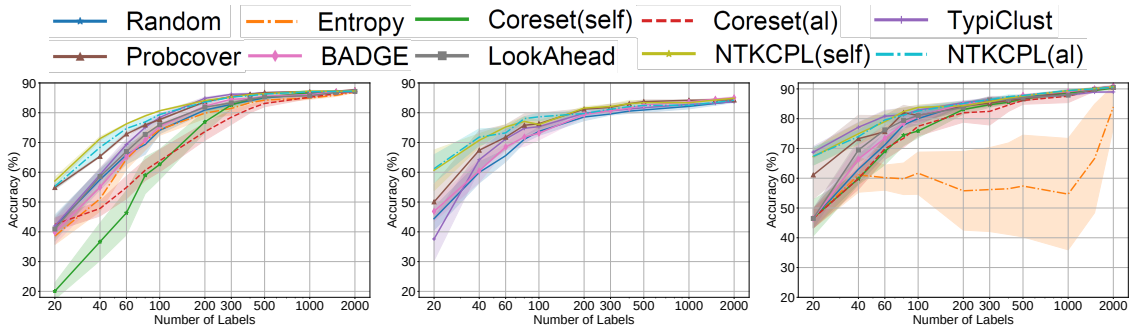
Figure 3.8 – Effect of the maximum number of clusters C_{max} on active learning performance on CIFAR-10.

these experiments, we found that clustering on active learning features yields better results except for the case where the number of annotations is very small. Also, NTKCPL(self) is better than NTKCPL(al) in a wide range of annotation quantities (no more than 500), when self-supervised features are good such as experiment in the CIFAR-10. The possible reason is the dimension of active learning features is smaller than that of the self-supervised features.

Effect of Pre-training Methods on NTKCPL To demonstrate the impact of different pre-training methods on NTKCPL, we conducted experiments using SimCLR, SimSiam, and CLIP pre-trained models on CIFAR-10. The details of the pre-training are provided in table 3.6. The active learning accuracy results are shown in fig. 3.9. NTKCPL exhibits strong performance using various pre-trained models. In the low-budget regime, it outperforms typical low-budget strategies such as TypiClust and ProbCover. Furthermore, compared to baseline methods, NTKCPL consistently outperforms the random baseline across a wider range of labeling budgets.

Table 3.6 – The details of pre-training methods.

Pre-training Method	Pre-training Data	Model
Simsiam (Chen and He, 2021)	CIFAR-10	ResNet-18
SimCLR (Chen et al., 2020a)	CIFAR-10	ResNet-18
CLIP (Radford et al., 2021)	WebImageText	ViT-B/32



(a) Simsiam **(b)** SimCLR **(c)** CLIP

Figure 3.9 – Performance of different active learning strategies. The shaded area represents standard deviation.

Choice of the Training Method In existing low-budget active learning literature, a common setting involves freezing the self-supervised backbone weights and only training the classification head (also the method employed in this chapter's experiments). Fine-tuning stands as another widely adopted training technique utilizing self-supervised models. We assessed the effectiveness of these two training methods within the low-budget scope, with randomly selected labeled samples. As depicted in table 3.7, table 3.8, and table 3.9, within the realm of low-budget, freezing the self-supervised weights and training only the classification head yields superior performance.

3.5.6 Running Time

We compare the computation time of our method with that of LookAhead (Mohamadi et al., 2022) (an active learning method based on NTK) as shown in fig. 3.10. The computational platform used for the experiments in this chapter consisted of an RTX 3090 (24GB) GPU paired with 15 Intel(R) Xeon(R) Platinum 8358P CPUs @ 2.60GHz. The code for this chapter was implemented using the PyTorch framework, with NTK calculations conducted using the Jax framework. Our method demonstrates a comparable sample selection time to LookAhead. When the dataset has few classes, our method takes slightly longer than LookAhead. Conversely, when the dataset has many classes, our method takes slightly less time than LookAhead.

Table 3.7 – Accuracy comparisons between the MLP-head probing (freezing the self-supervised backbone and training the MLP classifier head) and the fine-tuning on CIFAR-10. All results are averages over 5 runs. The best results are shown in red.

#Label	only MLP classifier	Fine-tuning
20	41.80±3.82	34.59±2.86
40	57.52±3.34	47.21±3.68
60	65.88±3.07	56.19±5.17
80	69.35±3.31	60.84±4.39
100	74.11±1.16	66.08±2.86
200	80.90±0.90	78.00±1.00
300	82.80±0.93	81.76±0.54
400	84.04±0.49	83.26±0.97
500	84.97±0.78	84.62±0.91
1000	86.26±0.38	86.70±0.27
1500	86.95±0.27	87.93±0.23
2000	87.30±0.37	88.47±0.25

Table 3.8 – Accuracy comparisons between the MLP-head probing (freezing the self-supervised backbone and training the MLP classifier head) and the fine-tuning on CIFAR-100. All results are averages over 5 runs. The best results are shown in red.

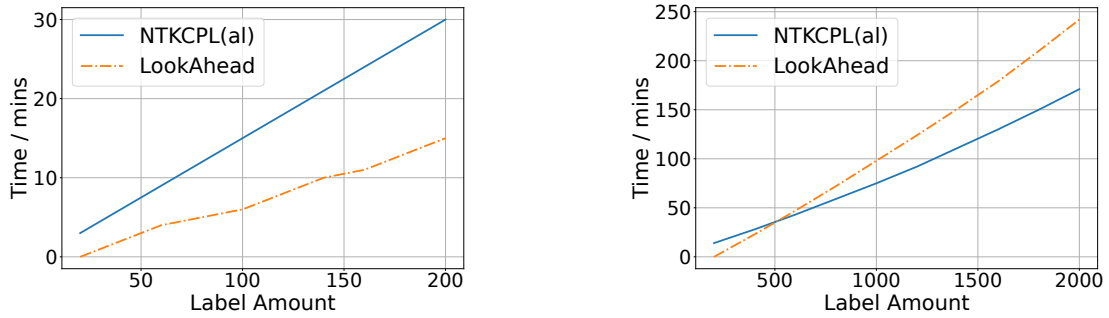
#Label	only MLP classifier	Fine-tuning
500	44.76±0.77	36.52±0.78
1000	52.51±0.36	48.85±0.83
1500	55.70±0.51	53.58±0.39
2000	57.12±0.62	56.78±0.28
2500	58.34±0.42	59.06±0.37
3000	59.22±0.38	61.06±0.70
3500	59.97±0.30	62.14±0.49
4000	60.70±0.25	62.86±0.53

3.5.7 Empirical Evidence for Approximation Analysis

Empirical Evidence of Argument Based on proposition, $\hat{f}_y(x) = g(\hat{f}_{cpl}(x))$, we argue $argmax\hat{f}_y(x)$ is most likely equal to $g(onehot(argmax(\hat{f}_{cpl}(x)))$. We validate this claim on the CIFAR-10 and CIFAR-100 datasets. The validity of this claim at different numbers of annotations is shown in fig. 3.11. This claim is valid in most cases, especially when the dataset has few classes.

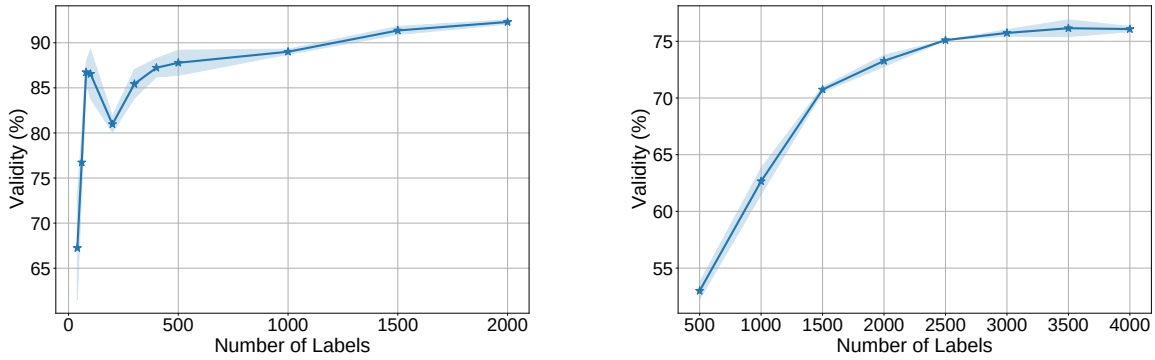
Table 3.9 – Accuracy comparisons between the MLP-head probing (freezing the self-supervised backbone and training the MLP classifier head) and the fine-tuning on ImageNet-100. All results are averages over 5 runs. The best results are shown in red.

#Label	only MLP classifier	Fine-tuning
200	50.89±3.33	46.47±1.54
400	62.98±1.81	60.34±0.89
600	74.30±1.09	68.06±0.50
800	76.17±0.68	71.47±0.46
1000	77.56±0.59	73.16±0.28
1500	80.78±0.16	76.29±0.36
2000	81.98±0.28	78.02±0.44



(a) SVHN **(b)** CIFAR-100

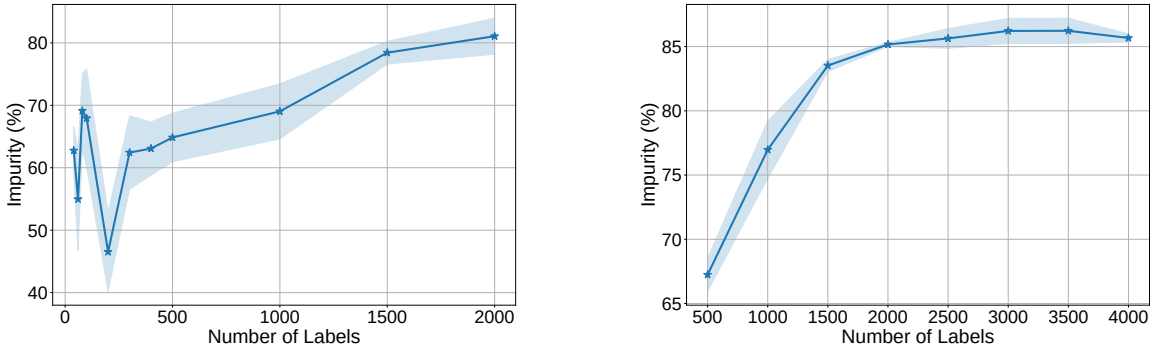
Figure 3.10 – Comparison of cumulative sample selection time on single RTX 3090 GPU on (a) SVHN and (b) CIFAR-100.



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

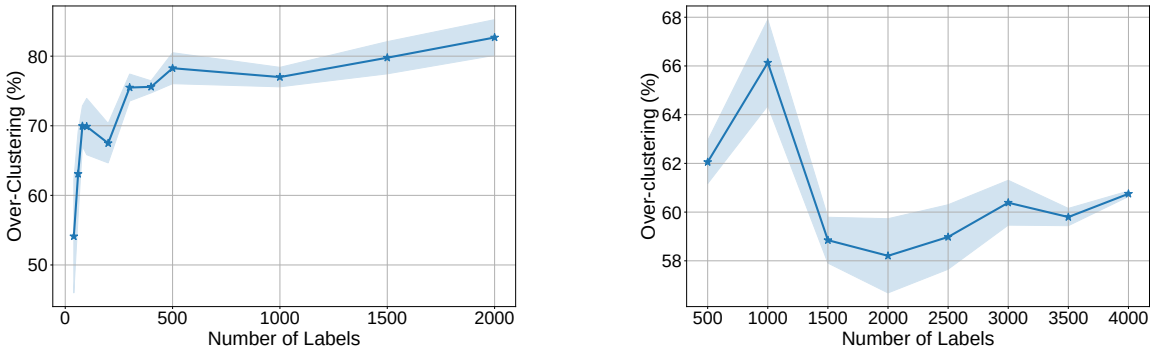
Figure 3.11 – The validity of the claim $(argmax\hat{f}_y(x) = g(onehot(argmax\hat{f}_{cpl}(x))))$ at different numbers of annotations on (a) CIFAR-10 and (b) CIFAR-100. The shaded area represents standard deviation.

Empirical Evidence The analysis of over-clustering error and impurity error is validated on the CIFAR-10 and CIFAR-100 datasets. Specifically, we examined the proportion of impure samples in the P_{nff} samples, as shown in fig. 3.12. Additionally, we also examined the proportion of over-clustering samples in the P_{fnf} samples, as shown in fig. 3.13. The experimental results show that impurity and over-clustering explain most of the approximation errors.



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

Figure 3.12 – Proportion of approximation error caused by impure samples within CPL to the P_{nff} term on (a) CIFAR-10 and (b) CIFAR-100. The shaded area represents standard deviation.



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

Figure 3.13 – Proportion of approximation error caused by over-clustering of CPL to the P_{fnf} term on (a) CIFAR-10 and (b) CIFAR-100. The shaded area represents standard deviation.

3.6 Limitations and Discussions

Due to the limited approximate effectiveness of NTK in fine-tuning pre-trained models, our approach is limited to the fixed training approach, i.e., training the classifier on top of a frozen self-supervised training encoder, which is suitable for the low-budget scenario because the fine-tuning training approach provides higher accuracy in the high-budget case. Therefore, (1) how to accurately approximate the fine-tuning model initialized with self-supervised weights using NTK and (2) whether the samples selected by our current method have good transferability for the fine-tuning would be interesting future directions.

3.7 Summary

In this chapter, we study active learning on top of self-supervised models. Our observations reveal significantly earlier emergence of phase transition points and these points vary notably across different datasets. Consequently, selecting existing low-budget active learning strategies becomes an unreliable choice. To improve this issue, we proposed a novel active learning strategy, NTKCPL, expanding the effective budget range by directly approximating empirical risk across the entire active learning pool. Moreover, we conduct an analysis of the approximation error, devising a CPL generation method based on this analysis to mitigate the approximation error. Our method outperforms SOTA in most cases and has a wider effective budget range. The comprehensive experiments show that our method can work well on self-supervised features with different qualities.

Chapter 4

Efficient Active Learning with a Large Pre-trained Model

4.1 Introduction

Training effective deep neural networks typically requires large-scale data (Deng et al., 2009; He et al., 2016; Liu et al., 2021). However, the high cost of acquiring data, especially annotated data, poses a significant challenge for practitioners (Budd et al., 2021; Norouzzadeh et al., 2021). Active learning and the pre-training fine-tuning paradigm are widely adopted strategies to address this challenge. Active learning reduces the demand for extensive annotation by iteratively selecting and labeling the most informative samples (Ren et al., 2021). Additionally, the pre-training fine-tuning method leverages large-scale unsupervised pre-training to create a powerful foundational model, enabling exceptional performance on downstream tasks with only a limited amount of labeled data (Caron et al., 2021; Chen et al., 2020a; Grill et al., 2020; He et al., 2022). To further minimize labels' demands, researchers have turned their attention to active fine-tuning, which fine-tunes the pre-training model with labeled samples selected by active learning strategies (Bengar et al., 2021; Chan et al., 2021; Xie et al., 2023). However, as the scale of pre-training models continues to grow, the sample selection time of active learning, which often is overlooked in traditional

active learning, becomes a challenge (Coleman et al., 2019). As illustrated in fig. 4.1, larger pre-training models lead to higher accuracy, but also significantly increase the time required for active learning sample selection.

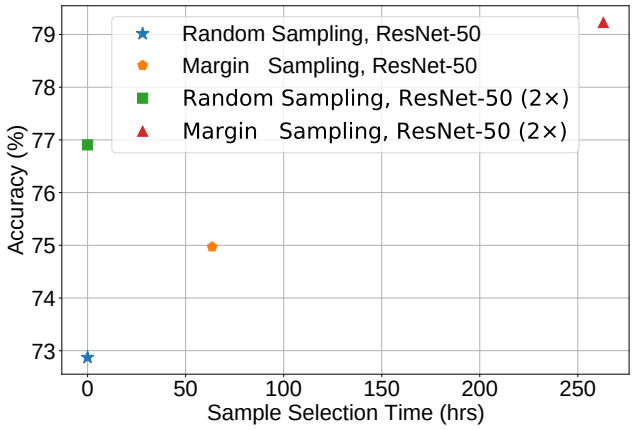


Figure 4.1 – Comparing sample selection time and accuracy in active learning (margin sampling) with different scale models. The active learning included a total of 16 iterations, selecting a total of 400k samples on the ImageNet dataset. All models are pre-trained using BYOL-EMAN (Cai et al., 2021).

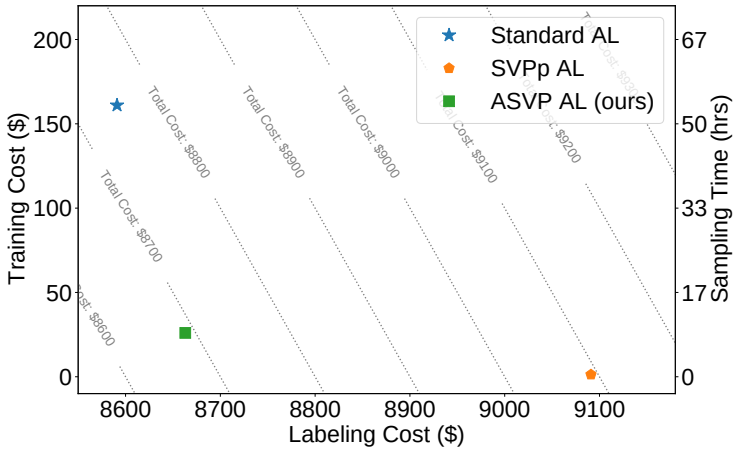


Figure 4.2 – Comparison of the labeling and training costs across random sampling, the standard active learning pipeline, the efficient active learning method (SVPp), and our method (ASVP), using margin sampling to achieve the same accuracy as randomly selecting 400k samples on the ImageNet dataset. We used a ResNet-50 model, and the training cost was priced based on AWS EC2 P3 instances (following existing paper (Zhang et al., 2024)), while annotation costs were estimated using AWS Mechanical Turk[1] with triple reviews.

[1]https://aws.amazon.com/sagemaker/groundtruth/pricing/

To address this issue, recent research has introduced an efficient active learning framework known as Selection via Proxy based on pre-trained features **(SVPp)** (Zhang et al., 2024). To avoid conceptual confusion, in this chapter, we use "effectiveness" to refer to the performance gain brought by active learning compared to the random baseline, and "efficiency" to refer to the time required for active learning to select samples. SVPp begins by forwarding the entire dataset through a pre-trained model once, recording pre-trained features for all samples. Subsequently, in multiple iterations of active learning, a simple MLP classifier is trained using the pre-computed features for sample selection. After the sample selection, the entire model is fine-tuned once.

However, since the proxy model uses pre-trained features as input, it is equivalent to freezing the pre-trained backbone and training only a simple classifier, it has limited feature learning capacity. In contrast, fine-tuning the entire model enables stronger feature learning. As the number of labeled samples increases, the fine-tuned model progressively learns features more aligned with the labeled data. This growing discrepancy between the predictions of the fine-tuned model and those of the proxy model based on pre-trained features can ultimately degrade the performance of SVPp.

We observe that, in comparison to the **standard active learning** method (fine-tuning the whole pre-trained model in each active learning iteration), SVPp may compromise the effectiveness of active learning, resulting in additional annotation costs. As depicted in fig. 4.2, we noticed that while SVPp significantly reduces active learning time (training costs), its decline in active learning performance leads to a notable increase in annotation costs. Practitioners therefore have to face the dilemma of weighing computational efficiency against overall cost (the sum of labeling cost and the training cost).

In light of this challenge, in this chapter, we analyze the factors contributing to the decline in active learning performance when using SVPp in sec. 4.3. Beyond the intuition-aligned reason for SVPp active learning performance decline: fine-tuned features being more capable of distinguishing sample categories than pre-trained features, leading to the selection of redundant samples (i.e., samples the fine-tuned model can already predict correctly), our analysis reveals an intriguing discovery:

when pre-trained features are of comparable quality to fine-tuned features, not all sample selection differences result in a decline in SVPp performance. Moreover, some performance drops are due to samples selected by SVPp causing substantial modifications to the pre-trained model during fine-tuning.

Based on these insights, we propose an aligned selection via proxy strategy, **ASVP**, to enhance the performance of efficient active learning in sec. 4.4. First, we align the pre-computed features used by the proxy model with the one used in standard active learning. Specifically, we update the pre-computed features based on the fine-tuned model when the pre-trained features' ability to distinguish sample categories significantly lags behind the fine-tuned features. In addition, we switch the training method between linear-probing then fine-tuning (LP-FT) (Kumar et al., 2022) and fine-tuning (FT) to mitigate the impact of samples selected by the proxy model on the model's performance.

Extensive experimental validation shows that our method clearly improves the effectiveness of efficient active learning, achieving similar or better total cost savings compared to those of standard active learning, while still maintaining computational efficiency.

The contributions of this chapter can be summarized as follows: (1) We empirically analyze the factors contributing to the performance drop when employing SVPp. The proxy model picks redundant samples where fine-tuned features outperform pre-trained ones. Also, it overlooks samples that are crucial to retain valuable pre-trained model information. (2) We propose a novel efficient active learning approach, aligned selection via proxy (ASVP), based on the above analysis. It improves the performance of efficient active learning while incurring a marginal amount of computation time. (3) We introduce a novel and practical evaluation metric for efficient active learning: the sample saving ratio. This metric directly quantifies the savings in annotation achieved by employing active learning strategies compared with the random baseline. It also facilitates the estimation of overall savings, including savings in both computational and labeling costs, thereby assisting practitioners in making a decision on whether to use efficient active learning strategies.

4.2 Preliminaries

4.2.1 Selection via Proxy

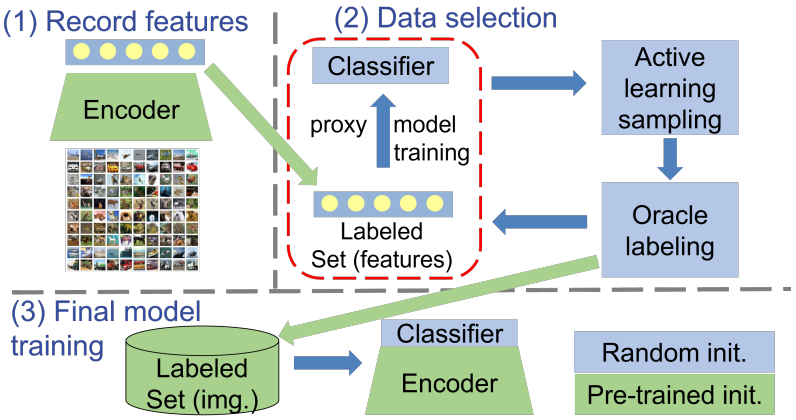


Figure 4.3 – The Selection via Proxy based on pre-trained features (SVPp) Framework. Stage 1: pre-computing features. Stage 2: sample selection based on the proxy model (a simple classifier) with pre-computed features as input. Stage 3: Fine-tuning the pre-trained model using labeled samples.

We briefly review the Selection via Proxy framework based on pre-computed features, SVPp, as shown in fig. 4.3. We denote the labeled set as L, the unlabeled set as U. The pre-trained neural network backbone is defined as $f(\cdot;\theta_p) : \mathcal{X} \to \mathbb{R}^d$, where θ_p represents weights of the pre-trained model, $\mathcal{X}$ is the data space and $\mathbb{R}^d$ is normalized feature space. And we define the predictor as $h(\cdot;\theta_h) : \mathbb{R}^d \to \mathbb{R}^c$, where θ_h is weights of the predictor and $\mathbb{R}^c$ is the output space. SVPp follows a three-step process. First, all samples are forward-passed through $f(\cdot;\theta_p)$, and the pre-trained features are saved. Subsequently, the proxy model is trained, and samples are selected iteratively based on the active learning strategy. In this process, the predictor $h(\cdot;\theta_h)$ serves as the proxy model, utilizing pre-computed features as its input. The final step is fine-tuning the pre-trained model $h(f(\cdot;\theta_p);\theta_h)$ based on the obtained labels.

4.2.2 LogME-PED

In practice, there are numerous models generated through various pre-trained methods. Conducting fine-tuning and testing for each pre-trained model to select the best one would incur substantial computational costs. To address this issue, LogME-PED is proposed for efficient pre-trained model selection (Li et al., 2023b; You et al., 2021).

The performance of linear-probing is often correlated with fine-tuning performance. Therefore, training a linear classifier on top of the pre-trained backbone serves as an indicator of the quality of the pre-trained model. However, research has shown that likelihood is prone to over-fitting and is not a reliable indicator (You et al., 2021). Therefore, **LogME** proposes using evidence as a metric, as shown in equation 4.1, where F denotes the pre-trained features of the labeled set and w denotes the parameters of the linear classifier. LogME assumes that these two probability terms follow Gaussian distributions, allowing for the efficient analytical calculation of equation 4.1. For specific calculation methods, refer to (You et al., 2021).

$$p(y|F) = \int p(w)p(y|w,F)dw \tag{4.1}$$

LogME can effectively assess the quality of pre-trained features. However, due to its inability to evaluate the changes in sample features caused by fine-tuning, LogME does not provide a comprehensive evaluation of the performance of the fine-tuned model.

PED is a physically inspired model, that simulates feature learning dynamics through multiple iterations (Li et al., 2023b). The approach draws an analogy between the optimization objective during fine-tuning and the concept of potential energy in physics. In fine-tuning, adjusting the model parameters forces the features of different classes to be distinguished, akin to the movement of objects influenced by forces in physics. Following this intuition, PED models the features of different classes as separate small balls, where the mean and standard deviation of each class's features correspond to the position and radius of the ball, respectively. The force between balls is proportional to their overlapping length. In this physical model, the positions and forces of

the objects are updated iteratively until convergence. Finally, the updated features are regarded as the features after fine-tuning the pre-trained model. Then, LogME is computed based on these updated features to assess the performance of the fine-tuned model.

4.3 Observations and Analysis

Since practitioners utilizing proxy models for efficient active learning aim to reduce sample selection time while minimizing the loss of active learning gains (between standard active learning based on fine-tuned models and SVPp), the exact overlap between samples selected by the proxy model and those selected by the fine-tuned model is not their primary concern. Therefore, in this section, we first investigate **which differences in sample selection lead to the degradation of SVPp active learning performance**.

4.3.1 Impact of Sample Selection Differences on SVPp Performance

We categorize the samples selected by both the proxy model and the fine-tuned model into different regions based on whether the models can correctly predict the samples and the active learning metrics (such as margin (Scheffer et al., 2001), entropy (Lewis and Catlett, 1994), etc.). As shown in fig. 4.4, the x and y axes represent whether the proxy model and the fine-tuned model can correctly predict the samples, respectively. The closer a sample is to the origin, the more likely it is to be selected in active learning. For uncertainty-based active learning strategies (e.g., margin, entropy, confidence (Wang and Shang, 2014)) and strategies combining uncertainty and diversity (e.g., BADGE (Ash et al., 2020)), a smaller distance from the origin indicates higher uncertainty, such as a smaller margin predicted by the model. For distance-based active learning strategies (e.g., coreset (Sener and Savarese, 2018)), a smaller distance from the origin indicates a greater distance between the sample and the labeled

set. Most effective active learning strategies for fine-tuning pre-trained models are uncertainty-based or combine uncertainty with diversity (where the latter extends the former by selecting diverse samples from the uncertainty candidates) (Emam et al., 2021; Zhang et al., 2024), this paper will use uncertainty-based active learning as an example for analysis.

Standard active learning based on the fine-tuned model selects samples from regions O, A, and B, while efficient active learning based on the proxy model selects samples from regions O, C, and D. In other words, compared to the fine-tuned model, the proxy model misses samples from regions A1, A2, B1, and B2 while additionally selecting samples from regions C and D.

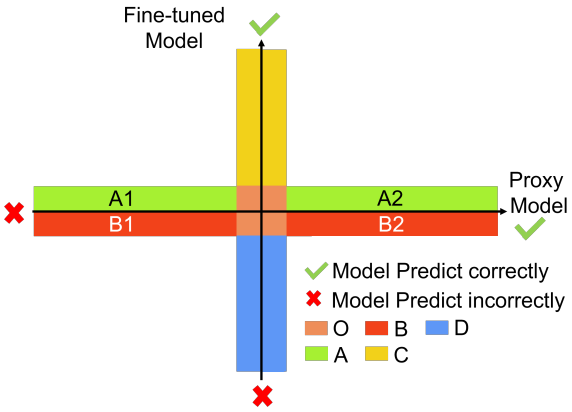


Figure 4.4 – Sample Selection Discrepancy: Proxy vs. Fine-tuned Models. The two axes represent the confidence of predictions for the proxy model and the fine-tuned model, with the positive half-axis indicating correct predictions and the negative half-axis indicating incorrect predictions. Uncertainty-based active learning selects samples with the lowest prediction confidence, meaning the proxy model chooses samples from regions O, C, and D, while the fine-tuned model selects samples from regions O, A, and B. The regions O and A, B, C, D are non-overlapping. All regions depicted here are conceptual illustrations, indicating whether they are selected and correctly predicted by the proxy model or fine-tuned model. Their width and shape do not represent the specific number of selected samples. For convenience in further analysis, we denote the incorrect predictions made by the proxy model in region B as B1, and the correct predictions as B2. Both the proxy model and the fine-tuned model confidently predict the remaining white regions, hence they do not select samples from them.

Empirical Evidence We conducted replacement experiments to demonstrate the impact of missing samples from regions A1, A2, B1, and B2 on SVPp active learning performance. In this section, we used one of the state-of-the-art strategies, margin sampling, with fine-tuned pre-trained models. A total of 400k samples were selected over 16 rounds, with the first 10k samples selected randomly. The experiments were conducted on ImageNet, with detailed settings provided in sec. 4.5.1.

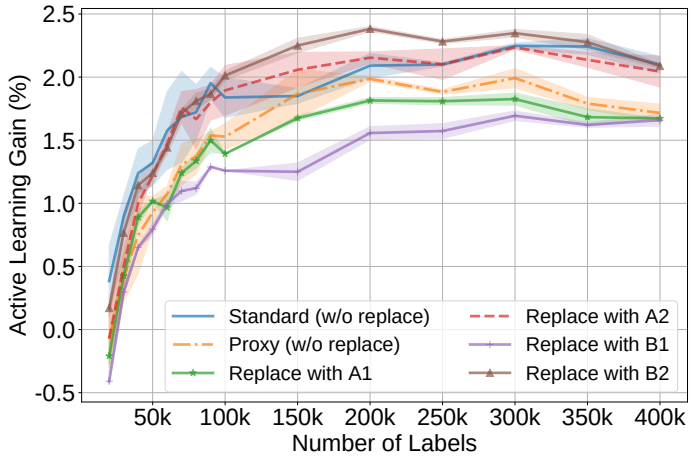
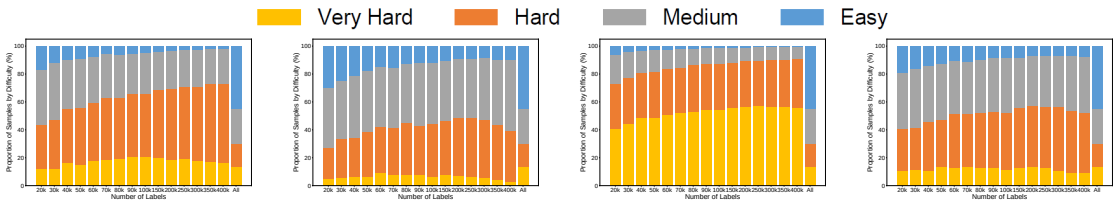


Figure 4.5 – The impact of different regions on SVPp active learning performance. Replacing samples selected by the proxy model with those from regions A1, A2, B1, B2. The active learning gain refers to the difference in accuracy between active learning and the random baseline.

Specifically, during each round of sample selection, we replaced the tail samples selected by the proxy model (i.e., those with the largest margins) with samples from regions A1, A2, B1, and B2. As shown in fig. 4.5, we observed that not all differences in sample selection caused performance degradation in SVPp active learning. Replacing samples with those from regions A2 and B2 improved SVPp active learning performance, replacing samples from region A1 had no effect, while replacing samples from region B1 decreased performance. Based on these observations, to improve the performance of SVPp active learning, we should focus on the sample selection differences from regions A2 and B2.

To investigate why samples from regions A1 and B1 fail to improve SVPp performance, we analyzed the difficulty of samples across different regions. Recent studies suggest that hard-to-learn samples are likely to undermine active learning perfor-

mance (Karamcheti et al., 2021). A1 and B1 regions, compared to A2 and B2, likely contain a higher proportion of difficult samples. This is because the fine-tuned model's predictions on these samples show high uncertainty, while the proxy model's predictions on them are wrong. To validate this hypothesis, we utilized Dataset Maps (Swayamdipta et al., 2020) to assess sample difficulty. Dataset Maps depict sample learnability by analyzing training dynamics from models trained on the entire dataset, using metrics such as average confidence for the correct class and prediction accuracy across training epochs to quantify difficulty. Following Karamcheti et al. (2021), we categorized sample difficulty based on mean confidence levels: very hard (<0.25), hard (>=0.25 and <0.5), medium (>=0.5 and <0.75), and easy (>=0.75). As illustrated in fig. 4.6, regions A1 and B1 indeed contain a higher proportion of very hard samples.



(a) A1 **(b)** A2 **(c)** B1 **(d)** B2

Figure 4.6 – Difficulty distribution of samples from different regions.

4.3.2 Role of Region A2 and B2 in Improving SVPp Performance

Those missed samples (Region A2, B2) that can improve SVPp performance share a common characteristic: the proxy model can confidently predict them correctly, while the fine-tuned model shows high uncertainty in its predictions. This indicates that these samples are well-distinguished using pre-trained features, whereas they are harder to differentiate in the fine-tuned features. Including these samples in the next round of active learning will help the fine-tuned model distinguish them. A question arises: **are the pre-trained feature manifold and the fine-tuned feature manifold (training with samples from regions A2 and B2) similar?**

Empirical Evidence To answer this question, we conducted a backbone exchange experiment to compare the similarity between the fine-tuned and pre-trained features. We put the linear classifier of the fine-tuned model on top of the pre-trained backbone and evaluated its accuracy loss on the test set. The greater the accuracy drop after exchanging the backbone, the more dissimilar the fine-tuned features are from the pre-trained features.

We created five sets of training samples by replacing 0% (w/o replacing), 5%, 10%, 15%, and 20% of the samples selected by the proxy model with the samples from regions A2 and B2 that had the smallest margins. After training on these sets, the accuracy loss is evaluated by putting the fine-tuned model's linear classifier on top of the pre-trained backbone. As shown in fig. 4.7, the accuracy drop decreased as more samples from A2 and B2 were included in the train set, indicating that incorporating these samples helps maintain the feature manifold similar to the pre-trained model.

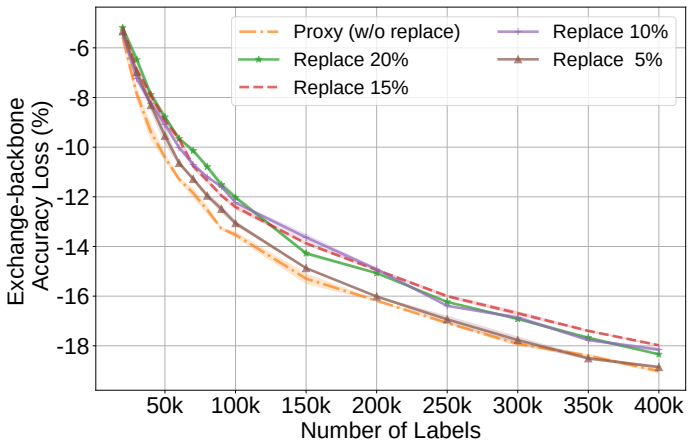


Figure 4.7 – The influence of samples from regions A2 and B2 on the similarity between fine-tuned and pre-trained features. Exchange-backbone accuracy loss refers to the accuracy difference between the fine-tuned model's linear classifier placed on the pre-trained backbone and the fine-tuned model.

Based on this observation, another intriguing question arises: **can directly maintaining the similarity between fine-tuned and pre-trained features mitigate the performance drop caused by the proxy model missing samples from regions A2 and B2?**

Empirical Evidence To explore this, we employed two training methods: fine-tuning and LP-FT (Linear Probing and then Fine-Tuning), to compare the performance differences between SVPp (sample selection via proxy model and evaluation via fine-tuning or LP-FT) and standard active learning (both sample selection and evaluation via fine-tuning or LP-FT). In LP-FT, linear probing is followed by fine-tuning to preserve the similarity between the fine-tuned features and the pre-trained features.

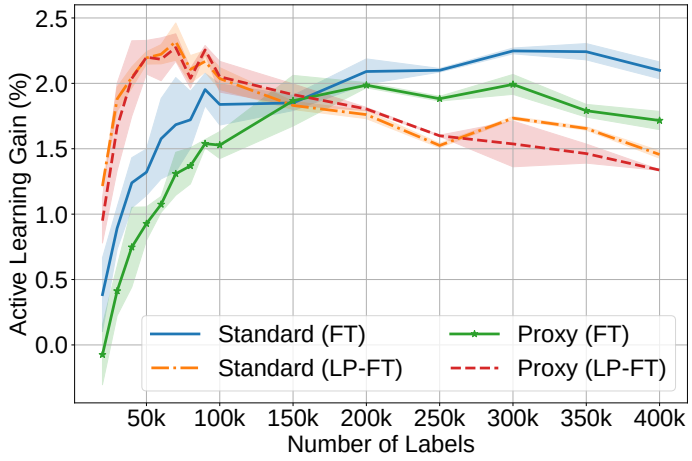


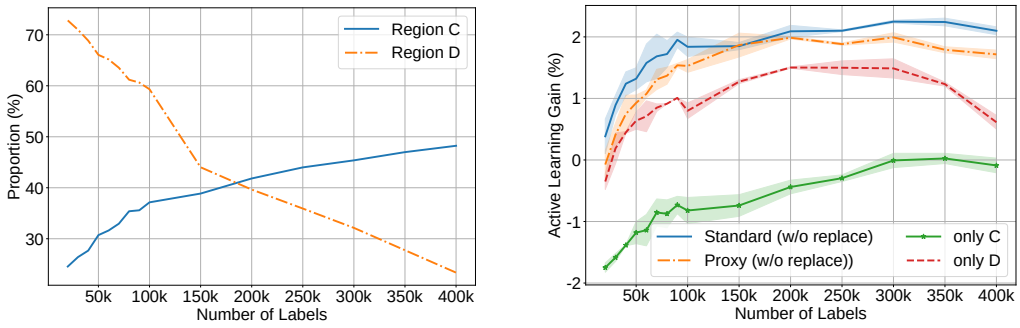
Figure 4.8 – Comparison of active learning performance based on the fine-tuned model (standard active learning) and proxy model when employing different training methods. The active learning gain refers to the difference in accuracy between active learning and the random baseline (using fine-tuning).

As shown in fig. 4.8, after applying LP-FT, the active learning performance based on the proxy model was almost identical to that based on the LP-FT model. This suggests that samples from regions A2 and B2 enhance active learning performance by preserving the similarity between the fine-tuned and pre-trained feature manifold. Therefore, directly preserving this similarity compensates for the decrease in active learning performance caused by the proxy model's omission of samples from regions A2 and B2.

4.3.3 Discussions

The previous section discussed that not all differences in sample selection between SVPp and standard active learning result in performance differences (only those from regions A2 and B2 do). Moreover, when using LP-FT to train the model, the sample selection differences from regions A2 and B2 do not cause a decline in SVPp active learning performance. Next, we discuss the limitations of the above analysis.

Firstly, given the training samples, we always aim to choose the training method that achieves the highest accuracy. Therefore, it is only meaningful to use LP-FT to compensate for the proxy model missing samples from regions A2 and B2 if LP-FT can achieve accuracy that is at least as good as fine-tuning. Secondly, as the number of selected samples increases, the performance gap between the fine-tuned model and the proxy model may widen (Cai et al., 2021). This could reduce the active learning gain of SVPp, thus affecting the previous conclusion about the impact of missed regions (A1, A2, B2, B2) by the proxy model on active learning performance. Therefore, we further analyzed the proportion of samples from regions C and D, as well as their impact on active learning gain.



(a) Proportion of Region C and D

(b) Active Learning Gain

Figure 4.9 – Observations on region C and D: (a) the proportion of region C and D, (b) comparison of active learning gain when selecting only samples from region C or D. The active learning gain refers to the difference in accuracy between active learning and the random baseline.

As shown in fig. 4.9a, with more samples selected during active learning, the proportion of samples from region C selected by the proxy model increases, while the

proportion from region D decreases. This aligns with intuition: as the number of annotations increases, the features of the fine-tuned model become better at distinguishing between sample categories than those of the proxy model, increasing correctly predicted samples (samples from region C). Additionally, as shown in fig. 4.9b, selecting samples from region C results in an active learning gain of less than 0, meaning it performs worse than random selection. These two observations imply that as more samples are selected during active learning, the proxy model selects more samples from region C, causing its active learning performance to decline. As the proxy model's active learning performance declines, it may lead to all missed regions (A1, A2, B1, B2) contributing to performance differences between SVPp and standard active learning.

In summary, when LP-FT training achieves performance comparable to fine-tuning and the proportion of samples from region C is small, using LP-FT can prevent the decline in active learning gains caused by SVPp. The next section discusses how to adjust proxy model-based active learning to build an efficient active learning framework when these conditions are not met.

4.4 Method

4.4.1 Aligned Selection via Proxy

This section first addresses the issue of the increasing proportion of region C samples, which makes LP-FT insufficient to prevent the decline in SVPp active learning performance. Region C refers to samples that the fine-tuned model can predict correctly with high confidence, while the proxy model struggles to differentiate them (high uncertainty). This indicates that the fine-tuned model's features are more suitable for distinguishing these samples. To address this, we recommend fine-tuning the pre-trained model as the number of region C samples increases. Updating the pre-computed features enhances the proxy model's discriminative capacity, thereby reducing the proportion of region C samples. Additionally, when updating pre-computed

features, we switch the model training method from LP-FT to fine-tuning. Continuing with LP-FT when fine-tuned features outperform pre-trained features can lead to suboptimal results by hindering necessary modifications to the pre-trained model.

The ASVP framework is divided into three stages: pre-computation of features, data selection, and final model training. The data selection stage of our method is illustrated in fig. 4.10. During the data selection stage, the proxy model uses pre-computed features (pre-computed features refer to pre-trained features when feature updating is not triggered) as input and performs multiple rounds of active learning (alternating between sample selection and proxy model training). At each iteration, we check if an update to the pre-computed features is needed. If the annotation budget is exhausted without triggering a feature update, LP-FT is used to train the final model. If an update is detected, fine-tuning is performed to generate new pre-computed features for the following rounds of sample selection. After the sample selection, the final model is trained using fine-tuning.

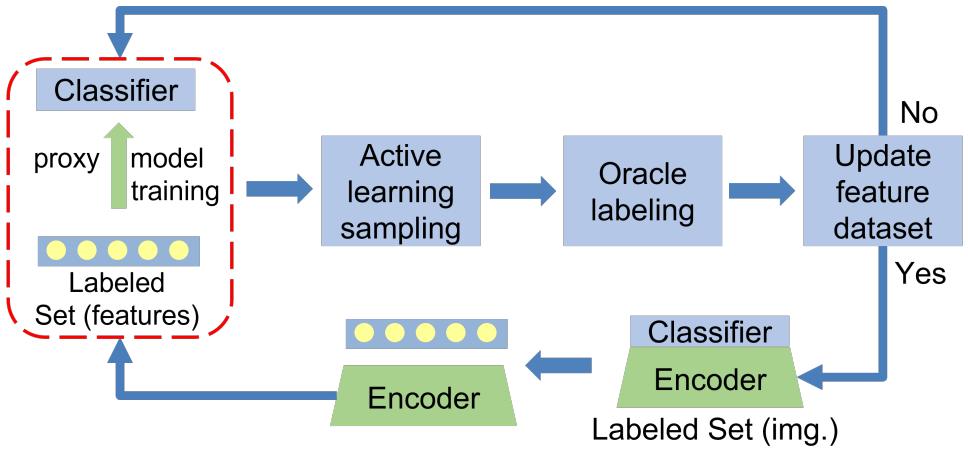


Figure 4.10 – The Data Selection Pipeline in our approach, Aligned Selection via Proxy (ASVP). After acquiring new labels from the oracle, we assess the necessity of updating the pre-computed features. If required, the pre-trained model is fine-tuned, and the pre-computed features are subsequently updated.

Next, we explain when it becomes necessary to update the pre-computed features. As discussed in sec. 4.3, as the number of annotations increases, the quality of fine-tuned features improves gradually, resulting in a higher proportion of samples selected by the proxy model from the redundant region. Also, including the samples from the

redundant region rarely improves the performance of the fine-tuned model. Hence, we assess the ratio of redundant region samples by evaluating the changes in the performance of the fine-tuned model after adding more annotated samples. To assess the changes in the fine-tuned model's performance, we employ LogME-PED (Li et al., 2023b; You et al., 2021) as an indicator.

Specifically, in every active learning iteration, we first compute the PED component. If convergence is achieved within a single iteration, we infer that the fine-tuned features will maintain similarity to the pre-trained features. So, there is no need to calculate the LogME metric and update the pre-computed features. However, if the PED component converges with more than one iteration, we proceed to compute the LogME metric. When the difference between the current LogME score and that from the previous active learning iteration is below a predefined threshold, we deduce that the fixed pre-trained features pose a significant obstacle to the performance improvement of the proxy model, prompting us to fine-tune the model and update the pre-computed features. The algorithm is shown in Alg. 3.

4.4.2 Computational Complexity

We define the average training time for the fine-tuned and the proxy models in each active learning iteration as $T_{tr,full}$ and $T_{tr,proxy}$, respectively. The time for processing all unlabeled samples through the whole pre-trained model and the proxy model to compute active learning metrics and select samples is denoted as $T_{f,full}$ and $T_{f,proxy}$, respectively. The time spent on forwarding all samples through the model to record features is represented by T_{pre}. The number of active learning iteration is given by N_{al}. The total active learning time for standard active learning, $T_{s,full}$, is $N_{al} \times (T_{tr,full} + T_{f,full})$. The total time for SVPp, $T_{s,svpp}$, is $N_{al} \times (T_{tr,proxy} + T_{f,proxy}) + T_{pre}$. For ASVP, the initial pre-computed features are obtained from a pre-trained model. When updating these pre-computed features is necessary, the model is fine-tuned on labeled data, and the pre-computed features are refreshed accordingly. Let N_{pre} denote the number of times the ASVP features are pre-computed, then, the time required for

Algorithm 3 ASVP

Input: Pre-trained backbone $f(\cdot;\theta_p)$, budget B, threshold τ, initial labeled set $L_0 = \{(x_i, y_i)\}_{i=1}^{N}$, initial unlabled set $U_0 = \{(x_i)\}_{i=1}^{M}$
Output: Trained final model $h(f(\cdot))$, updated labeled set L_f
Use pre-trained model $f(\cdot;\theta_p)$ to pre-compute features for all samples in the dataset $L_0 \cup U_0$.
Set $L_f = \{(f(x_i;\theta_p), y_i) \ | x_i \in L_0\}_{i=1}^{N}$, $U_f = \{(f(x_i;\theta_p)) | x_i \in U_0\}_{i=1}^{M}$
for $t = 0$ to $T-1$ **do**
    Train proxy model $h(\cdot)$ using L_f.
    Select a batch of samples S_t from U_f based on the proxy model $h(\cdot)$.
    Annotate samples in S_t and update labeled set $L_f = L_f \cup S_t$, unlabeled set $U_f = U_f \setminus S_t$.
    Compute PED (Li et al., 2023b)
    if PED converges in one iteration **then**
        Skip LogME computation and continue.
    else
        Compute LogME score ℓ_t (You et al., 2021).
        if $|\ell_t - \ell_{t-1}| < \tau$ **then**
            Fine-tune $h(f(\cdot;\theta_p))$ to update pre-computed features.
        end if
    end if
end for
Set $L = \{(x_i, y_i) \ | f(x_i) \in L_f\}_{i=1}^{N+BT}$
if Pre-computed features are updated **then**
    Train $h(f(\cdot;\theta_p))$ using Fine-tuning with L.
else
    Train $h(f(\cdot;\theta_p))$ using LP-FT with L.
end if
Return: Trained final model $h(f(\cdot))$, updated labeled set L

pre-computing features is $N_{pre} \times T_{pre} + (N_{pre} - 1) \times T_{tr,full}$. The total time required by ASVP, denoted as $T_{s,asvp}$, is $N_{al} \times (T_{tr,proxy} + T_{f,proxy}) + N_{pre} \times T_{pre} + (N_{pre} - 1) \times T_{tr,full}$.

4.5 Results

Our method is validated on four datasets: ImageNet-1k (Deng et al., 2009), CIFAR-10 (Krizhevsky et al., 2009), CIFAR-100 (Krizhevsky et al., 2009) and Oxford-IIIT Pet dataset (Parkhi et al., 2012) with five typical active learning strategies. The

experiment setup is clarified in sec. 4.5.1. We propose the new efficient active learning evaluation metric in sec. 4.5.2. The results are shown in sec. 4.5.3 and sec. 4.5.4. The ablation study is shown in sec. 4.5.5. The detailed study of the impact of the position of updating features is shown in sec. 4.5.5. Additionally, given the efficiency of our framework, decreasing the amount of sample selected in each active learning iteration to alleviate the redundant problem of batch active learning strategy is feasible. The results are shown in sec 4.5.6.

4.5.1 Experiment Setup

Active learning strategies. To evaluate the compatibility of our proposed ASVP with existing active learning strategies, four typical active learning strategies are included: (1) Margin: uncertainty-based sampling, one of the SOTA in the context of fine-tuning pre-trained models (Scheffer et al., 2001), (2) Confidence: uncertainty-based sampling (Wang and Shang, 2014), (3) Coreset: diversity-based sampling (Sener and Savarese, 2018), (4) BADGE: combination of uncertainty and diversity (Ash et al., 2020), (5) ActiveFT(al), feature space coverage with features from fine-tuned model (Xie et al., 2023).

Baseline. (1) Standard active learning: training the whole model with fine-tuning (FT) or linear-probing then fine-tuning (LP-FT) (Kumar et al., 2022) in each active iteration, (2) SVPp (Zhang et al., 2024): selecting samples via MLP proxy model based on pre-trained features and training the final model with FT, (3) ActiveFT(pre) (Xie et al., 2023): a single-shot active learning strategy that selects all sample in single iteration based on pre-trained features.

Implementations. We utilized the ResNet-50 model (He et al., 2016), pre-trained on ImageNet using the BYOL-EMAN (Cai et al., 2021) method in all our experiments. For the baseline method (SVPp) and our method (ASVP), we employed 2-layer MLPs as proxy models, where its architecture is Linear + BatchNorm + ReLU + Linear. The hidden layer width matched the input feature dimensions. We performed multiple active learning iterations until the model performance approached

that of training on the entire dataset (upper bound). Specifically, for the ImageNet, CIFAR-10, CIFAR-100, and Pets datasets, we conducted 16, 18, 22, and 10 active learning iterations, respectively. The query step was set to 200 samples per iteration for the Pets dataset. For ImageNet, CIFAR-10, and CIFAR-100, we used varying query steps: for ImageNet, we queried 10,000 samples per iteration until reaching 100,000 samples, after which we queried 50,000 samples per iteration. For CIFAR-10, we queried 200 samples per iteration until reaching 2,000 samples, then increased to 500 samples per iteration. For CIFAR-100, we queried 400 samples per iteration until reaching 6,000 labeled samples, then switched to 2,000 samples per iteration. The difference in LogME-PED scores between consecutive active learning iterations is set as 1 for the threshold of updating pre-computed features.

Fine-tuning, linear probing then fine-tuning (LP-FT), and proxy model training all utilized the SGD with momentum optimizer, with a momentum value of 0.9. For fine-tuning and LP-FT, the batch size in the training stage is 128 and the batch size in the sample selection stage of the active learning is 512.

The hyper-parameters of fine-tuning are shown in table 4.1. For ImageNet, we adopted the fine-tuning parameters following the prior work Cai et al. (2021). Setting the learning rates separately for the classifier head and the backbone. For CIFAR-10, CIFAR-100, and Pets, we followed prior work Cai et al. (2021) to set the learning rate of the classifier as 0.1 and to search other hyper-parameters. The hyper-parameters are detailed in table 4.1. The learning rate of the backbone was searched from (0.01, 0.05, 0.1 0.3) and the weight decay from (0, 0.0001). Additionally, to maintain consistency in training iterations between fine-tuning and LP-FT, we set the number of fine-tuning epochs as the sum of linear probing and fine-tuning epochs in LP-FT training.

For the LP-FT hyper-parameters, we conducted a search within the following ranges: (0.01, 0.1) for the classifier's learning rate, (0.01, 0.05, 0.1, 0.3) for the backbone's learning rate, (0.05,0.1,0.5) for learning rate in linear-probing (LP) stage, and (0, 0.0001) for weight decay. Fine-tuning (FT) training epochs were set at 120. For ImageNet, linear probing (LP) training epochs were searched from (5, 10), while for

Dataset	Learning Rate		Training	Weight
	Classifier	Backbone	Epochs	Decay
ImageNet	0.1	0.01	50	0.0001
CIFAR-10	0.1	0.1	130	0
CIFAR-100	0.1	0.05	150	0
Pets	0.1	0.1	140	0

Table 4.1 – The hyper-parameters of the fine-tuning.

other datasets, LP training epochs were explored from (5, 10, 20, 30). Given the numerous hyper-parameters in LP-FT, we sequentially search the backbone learning rate, classifier learning rate, weight decay, and LP training epochs. The final hyper-parameters are shown in table 4.2, where the weight decay is 0 and the learning rate in the LP stage is 0.5 across all datasets.

Dataset	Learning Rate		Training Epochs	
	Classifier	Backbone	Linear-probing	Fint-tuning
ImageNet	0.01	0.01	5	45
CIFAR-10	0.1	0.01	10	120
CIFAR-100	0.1	0.01	30	120
Pets	0.1	0.1	20	120

Table 4.2 – The hyper-parameters of the linear-probing then fine-tuning.

The proxy model was trained 50 epochs with a learning rate of 0.01 and the weight decay of 0. During the training stage, the batch size is 2048 and in the active learning stage, the batch size is 16384.

4.5.2 Metric

Efficient active learning methods often exhibit a lower accuracy compared to standard active learning approaches. Understanding the additional annotation costs needed to compensate for the accuracy gap caused by efficient active learning allows practitioners to weigh the computational cost savings against increased labeling expenses, aiding their decision to utilize these methods.

To this end, we propose a new evaluation metric: equivalent saving amount. Let N_1 represent the number of samples selected by the active learning strategy, and let A be the accuracy achieved by a model trained with these N_1 labeled samples. We utilize interpolation to estimate the number of samples, N_2, required to achieve accuracy A for a random baseline. We refer to N_2 as the **equivalent number of non-AL labels** in the subsequent discussions. The number of saved samples is $N_2 - N_1$ and **sample saving ratio** is $\frac{N_2 - N_1}{N_2}$. Subsequently, we compute the **average sample saving ratio** across all iterations of active learning to assess the performance of the active learning strategy. Furthermore, the **overall cost** of active learning, including annotation and training costs, equals $N_1 \times P_l + C_{tr}$, where P_l denotes the unit price of annotation and C_{tr} represents the training cost of active learning.

In this chapter, we primarily present the average sample savings ratio, overall costs, and computational time. Additionally, detailed accuracy and the corresponding savings in sample count in each active learning iteration are provided.

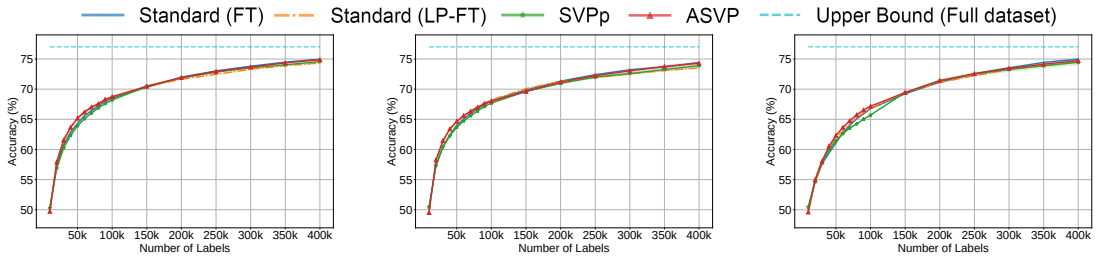
4.5.3 Experiment Results

The average sample savings ratios are presented in table 4.3, with each experimental setting repeated three times to report both the average and standard deviation. The table also illustrates the total cost required for active learning to reach the accuracy achieved by the random baseline at the last active learning iteration (i.e., the accuracy achieved when randomly selecting 400k, 6000, 20000, and 2000 samples from ImageNet, CIFAR-10, CIFAR-100, and Pets datasets, respectively). The training cost is assessed based on AWS EC2 P3 instances (Zhang et al., 2024), while the annotation cost is evaluated using AWS Mechanical Turk, with three annotations per sample to ensure labeling quality. Additionally, the accuracy and corresponding saving in label counts on ImageNet are shown in fig. 4.11 and fig. 4.12. The other results are shown in fig. 4.13-fig. 4.18. In fig. 4.19-fig. 4.22, we illustrate the overall costs (the labeling cost and the active learning training cost) and computational efficiency achieved by the standard active learning method, SVPp, and our proposed ASVP across different

datasets.

Selection Method	Training Method	AL Strategy	ImageNet Avg. Sample Saving Ratio	ImageNet Cost ($)	CIFAR-10 Avg. Sample Saving Ratio	CIFAR-10 Cost ($)	CIFAR-100 Avg. Sample Saving Ratio	CIFAR-100 Cost ($)	Pets Avg. Sample Saving Ratio	Pets Cost ($)	Cost Saving Ratio
Standard	FT	Random	0%	14400	0%	216	0%	720	0%	72	100%
Standard	FT	Margin	30.30±1.09	8752	40.23±0.51	134	14.54±2.46	578	31.67±2.30	49	68%
		BADGE	21.33±0.38	10018	36.18±0.34	128	13.73±2.07	580	33.01±3.17	47	69%
		Confidence	-3.42±0.75	9619	24.99±1.37	142	4.23±2.34	567	27.24±3.28	50	70%
		Coreset	-	-	12.44±2.42	192	-13.90±2.31	722	-4.05±2.31	74	97%
		ActiveFT(al)	-	-	12.39±3.26	186	3.03±1.17	798	-5.20±2.28	97	111%
Standard	LP-FT	Margin	32.22±0.27	9884	37.39±1.28	139	22.31±1.83	509	6.97±2.55	73	76%
		BADGE	24.12±0.54	11901	29.54±0.70	136	19.81±1.28	528	6.74±3.84	68	78%
		Confidence	0.11±1.51	10125	20.10±4.01	138	13.49±2.90	510	9.58±3.41	73	77%
		Coreset	-	-	-4.60±4.09	244	-16.76±1.78	769	-36.93±4.53	110	124%
		ActiveFT(al)	-	-	-2.13±0.11	249	6.38±1.35	755	-29.33±10.58	104	122%
Single-shot	FT	ActiveFT(pre)	-	-	1.40±1.50	235	-0.21±1.41	822	-11.92±1.26	118	129%
	LP-FT		-	-	5.02±4.02	238	8.92±2.34	742	-38.10±2.61	235	180%
SVPp	FT	Margin	25.16±1.47	9094	19.86±2.53	184	3.51±0.77	654	23.48±4.09	50	77%
		BADGE	16.17±1.84	11374	19.35±3.66	156	0.51±0.74	657	-7.78±2.42	85	90%
		Confidence	-8.56±2.88	9678	1.53±1.82	171	-7.93±1.85	644	2.40±7.25	66	82%
		Coreset	-	-	-54.78±8.88	1355	-56.53±3.12	903	-38.63±11.64	87	291%
		ActiveFT(al)	-	-	3.79±4.31	192	-0.50±1.42	713	-11.63±6.85	319	210%
ASVP	FT/LP-FT	Margin	34.38±1.57	8690	32.45±1.13	127	12.82±1.40	590	21.98±0.50	47	67%
		BADGE	25.57±2.54	10305	32.87±0.84	137	10.11±1.13	577	6.10±5.59	55	73%
		Confidence	5.88±0.66	9479	25.67±2.48	167	2.75±1.17	616	14.81±3.17	50	75%
		Coreset	-	-	11.11±1.74	192	-26.16±1.76	767	-46.3±10.26	91	107%
		ActiveFT(al)	-	-	26.67±0.66	155	6.81±1.70	775	-7.74±4.81	47	82%

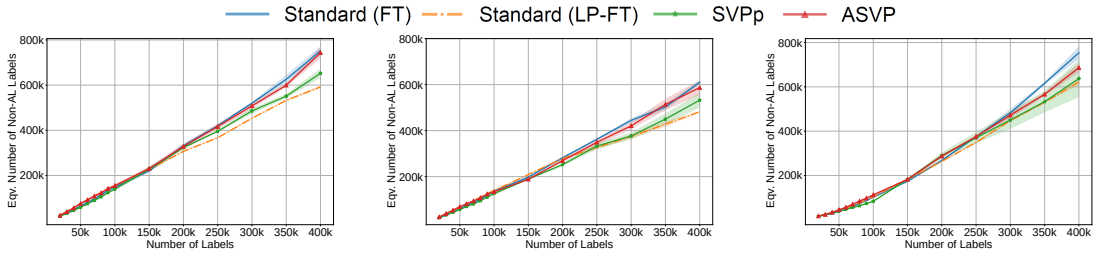
Table 4.3 – Average sample savings ratios, with each configuration repeated three times to calculate the average and standard deviation. The training method indicates how the final model is trained, and the selection method indicates the model used for sample selection. The cost refers to the total cost required for active learning to achieve the accuracy achieved by the random baseline in the last active learning iteration. The cost-saving ratio refers to the average ratio of the cost of active learning to the cost of the random baseline. The best results are shown in red and the second-best results are in blue.



(a) Margin **(b)** BADGE **(c)** Confidence

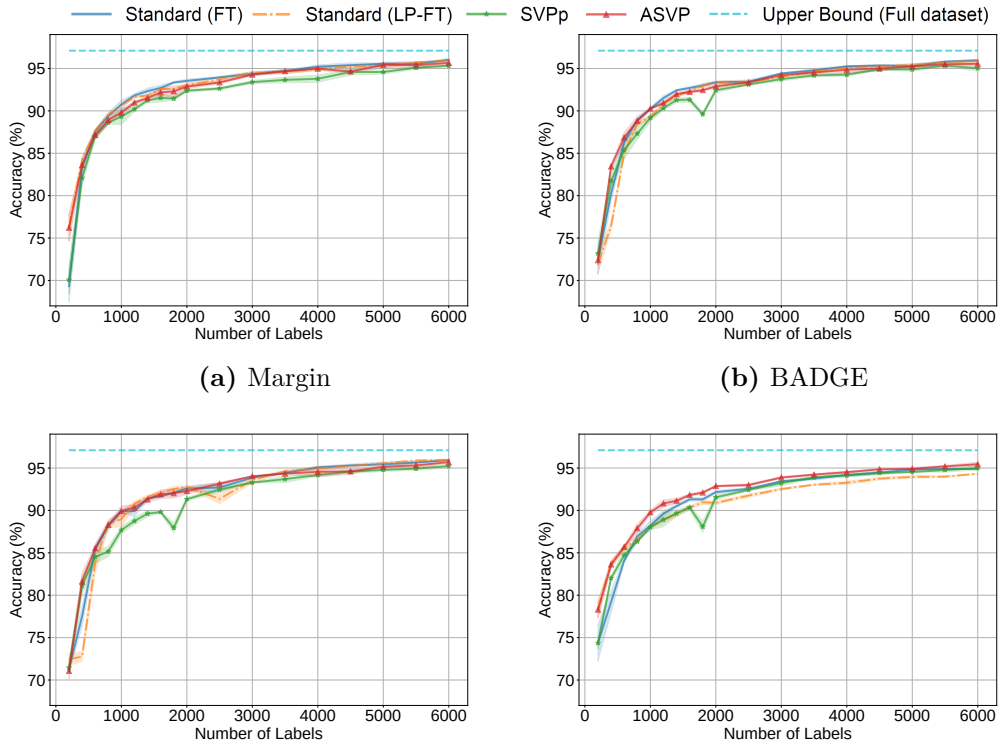
Figure 4.11 – The accuracy of the standard method, SVPp, and our method ASVP on the ImageNet using (a) Margin, (b) BADGE, and (c) Confidence sampling.

Our method consistently outperforms the efficient active learning method, SVPp, in terms of sample saving ratio and overall cost reduction. It surpasses the single-shot



(a) Margin **(b)** BADGE **(c)** Confidence

Figure 4.12 – The equivalent number of non-active learning label amounts comparison of the standard method, SVPp, and our method ASVP on ImageNet using (a) Margin, (b) BADGE and (c) Confidence sampling. The equivalent non-active learning label amount refers to the number of samples selected by a random baseline to achieve the same accuracy as active learning.

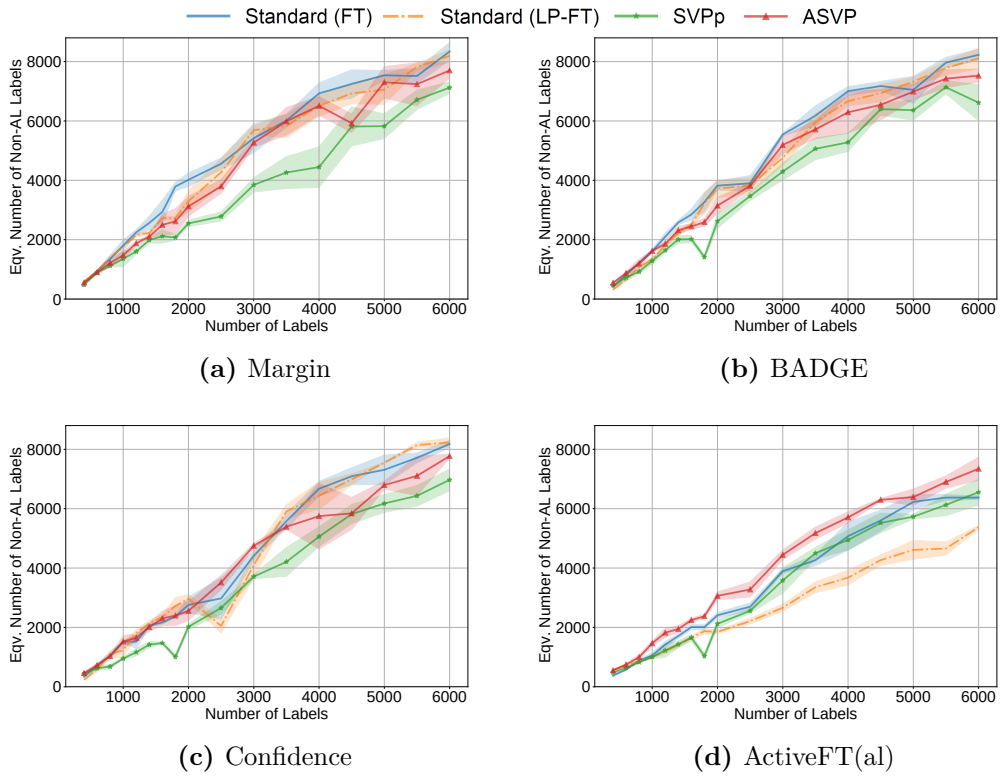


(a) Margin **(b)** BADGE

(c) Confidence **(d)** ActiveFT(al)

Figure 4.13 – The accuracy of the standard method, SVPp, and our method ASVP on the CIFAR-10 using (a) Margin, (b) BADGE, (c) Confidence and (d) ActiveFT(al) sampling.

active learning strategy, ActiveFT(pre), when combined with most active learning strategies. In most cases, our approach offers similar or greater overall cost savings



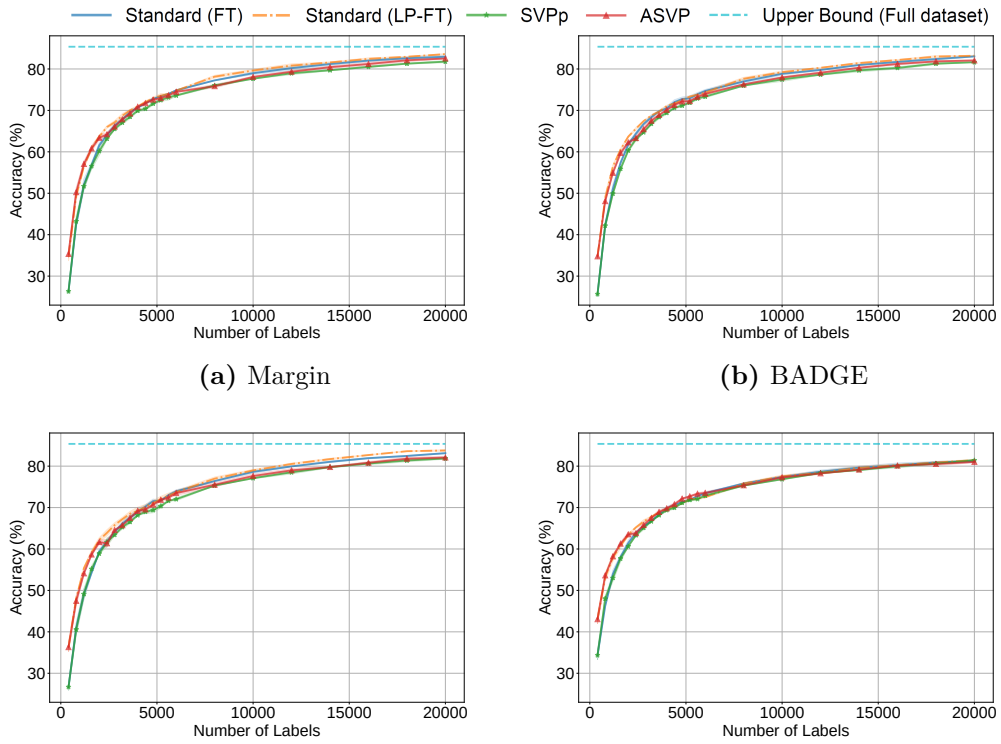
(a) Margin **(b)** BADGE

(c) Confidence **(d)** ActiveFT(al)

Figure 4.14 – The equivalent number of non-active learning label amounts comparison of the standard method, SVPp, and our method ASVP on the CIFAR-10 using (a) Margin, (b) BADGE, (c) Confidence and (d) ActiveFT(al) sampling. The equivalent non-active learning label amount refers to the number of samples selected by a random baseline to achieve the same accuracy as active learning.

compared to standard active learning. This alleviates practitioners' concerns about overall costs when using efficient active learning. Additionally, in standard active learning methods, the lack of a clear criterion for choosing the proper training method may lead to suboptimal results. Specifically, the standard active learning with FT outperforms the standard active learning with LP-FT on CIFAR-10 and Pets, while on ImageNet and CIFAR-100, the standard (LP-FT) method demonstrates better performance. In contrast, our approach, by switching training methods, ensures a relatively favorable performance in practical applications.

Compared to the single-shot active learning method ActiveFT(pre), our approach is compatible with various existing active learning strategies and exhibits lower dependence on the quality of pre-trained features. The reliance of ActiveFT(pre) on



(a) Margin **(b)** BADGE

(c) Confidence **(d)** ActiveFT(al)

Figure 4.15 – The accuracy of the standard method, SVPp, and our method ASVP on the CIFAR-100 using (a) Margin, (b) BADGE, (c) Confidence and (d) ActiveFT(al) sampling.

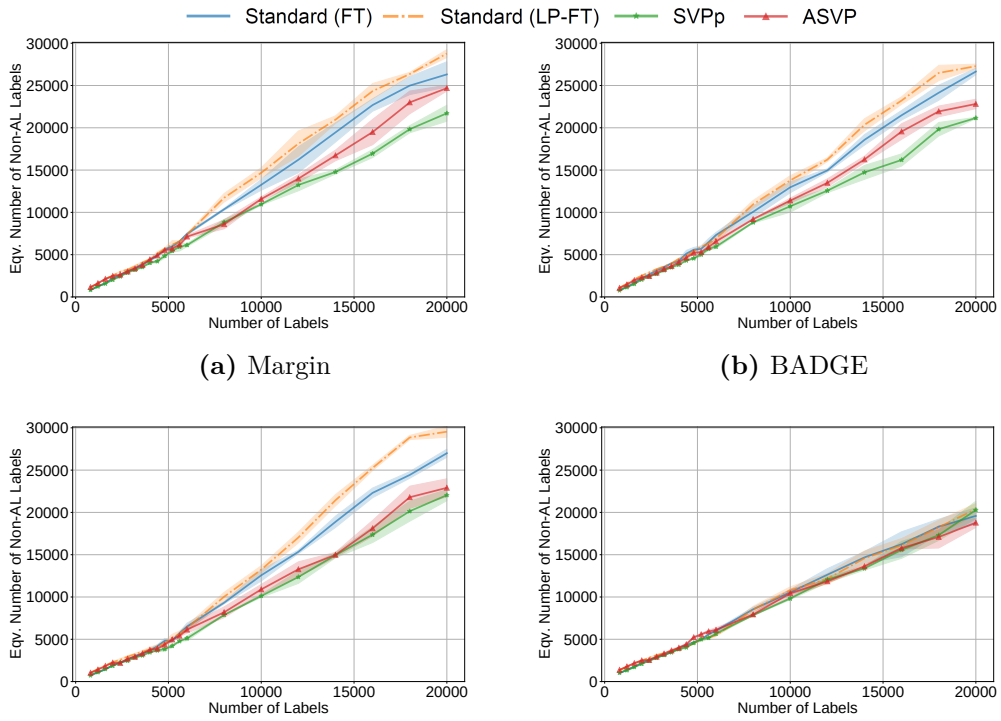
the quality of pre-trained features leads to suboptimal results. This is particularly evident in datasets like Pets, where the pre-trained feature quality is low.

4.5.4 Running Time

	$T_{pre}(hrs)$	$N_{al} \times T_f(hrs)$	$N_{al} \times T_{tr}(hrs)$	$T_s(hrs)$
Standard	-	16.36	80.97	97.33
SVPp	0.92	0.10	0.46	1.48
ASVP	8.60	0.10	0.46	9.16

Table 4.4 – Computation time comparison of various active learning sample selection methods on ImageNet using single V100 GPU.

The computation time of standard active learning, SVPp, and ASVP on ImageNet is presented in table 4.4. The results on other datasets are shown in appendix 4.5.



(a) Margin **(b)** BADGE

(c) Confidence **(d)** ActiveFT(al)

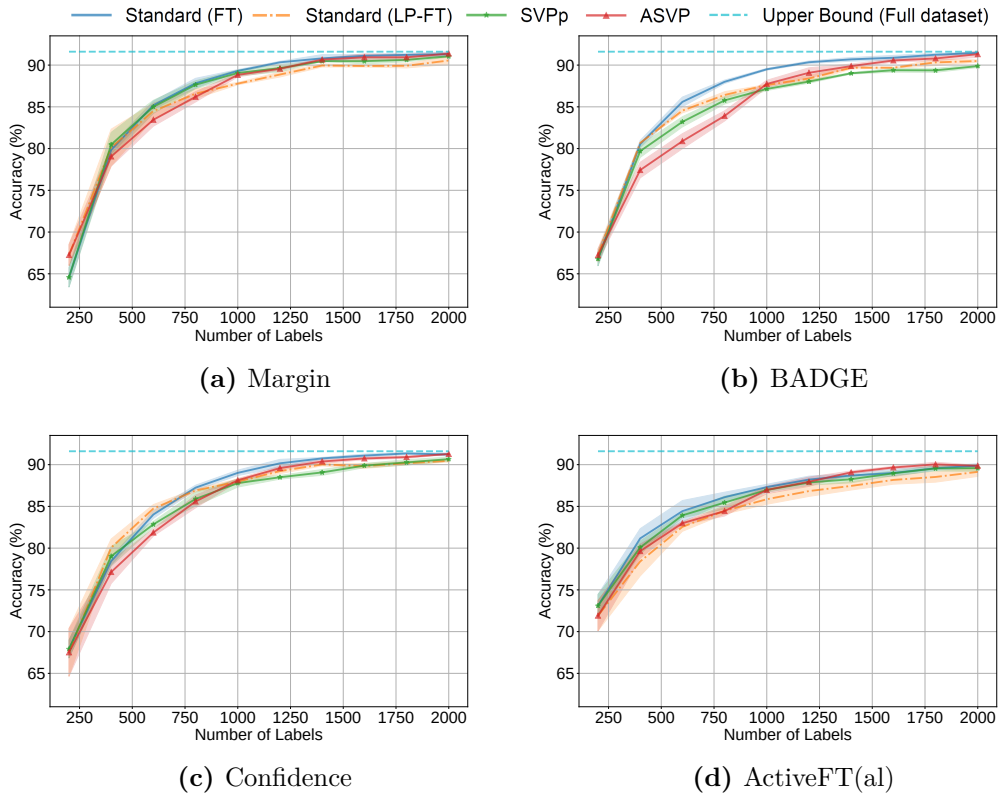
Figure 4.16 – The equivalent number of non-active learning label amounts comparison of the standard method, SVPp, and our method ASVP on the CIFAR-100 using (a) Margin, (b) BADGE, (c) Confidence and (d) ActiveFT(al) sampling. The equivalent non-active learning label amount refers to the number of samples selected by a random baseline to achieve the same accuracy as active learning.

The time for ASVP mainly arises from updating pre-computed features, as the computation time for the proxy model, an MLP, is significantly lower than that of the fine-tuned model. This means that increasing the number of active learning iterations N_{al} has a marginal impact on ASVP's total time. Therefore, this opens up the potential to improve the diversity of selected samples by reducing the number of samples chosen in each active learning iteration.

4.5.5 Detailed Study

Ablation Study

The ablation study was conducted on CIFAR-10, CIFAR-100, and Pets and the results



(a) Margin **(b)** BADGE

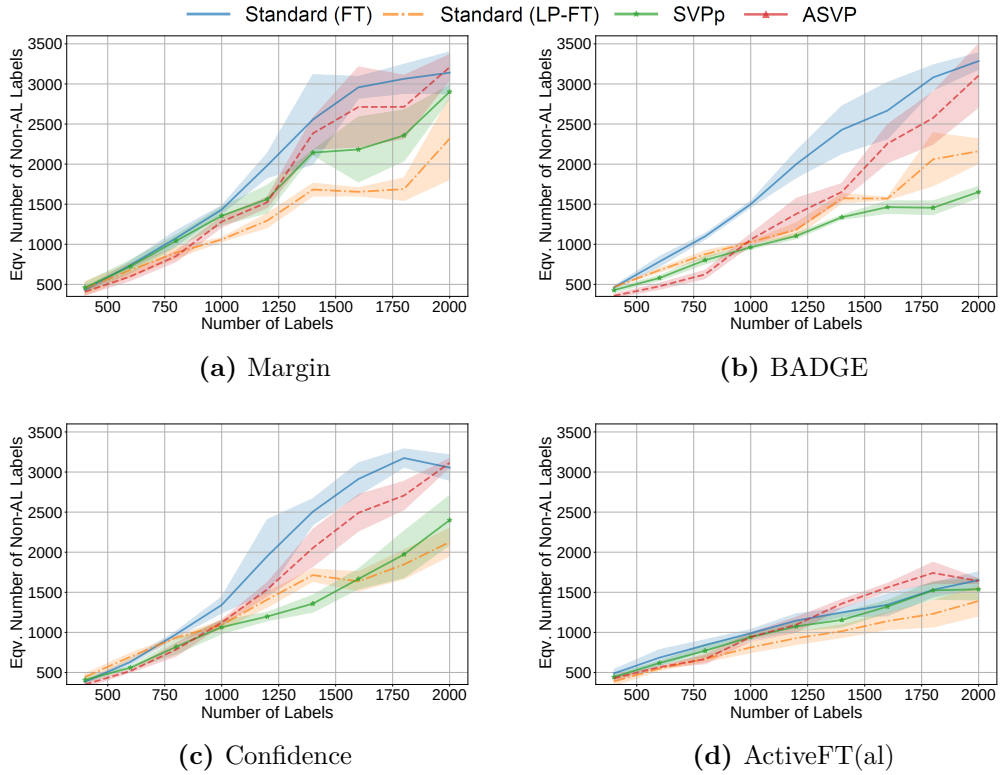
(c) Confidence **(d)** ActiveFT(al)

Figure 4.17 – The accuracy of the standard method, SVPp, and our method ASVP on the Pets using (a) Margin, (b) BADGE, (c) Confidence and (d) ActiveFT(al) sampling.

	CIFAR-10 (hrs)	CIFAR-100 (hrs)	Pets (hrs)
Standard	6.94	20.96	1.50
SVPp	0.22	0.13	0.03
ASVP	0.85	1.10	0.05

Table 4.5 – Sample selection time of standard active learning methods, SVPp, and our proposed method ASVP on CIFAR-10, CIFAR-100, and Pets. Experiments conducted on single V100 GPU.

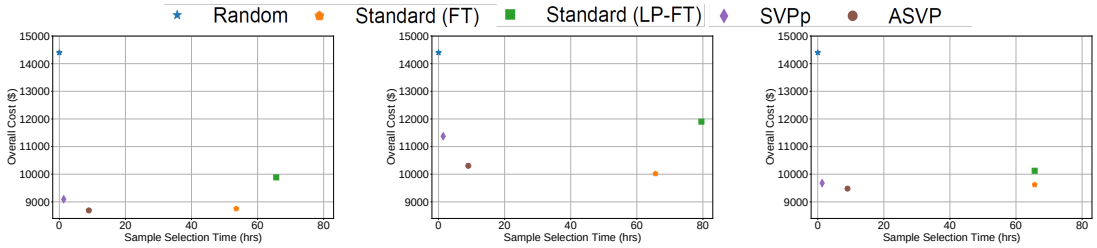
are presented in table 4.6. Across all three datasets, updating pre-computed features (feature alignment) consistently led to a noticeable performance boost. Meanwhile, switching training methods (training alignment) improved performance on all datasets except for the Pets dataset, suggesting that our method may not precisely determine the optimal point for switching training methods. However, considering the practical scenario where we cannot pre-determine whether using FT or LP-FT for training is



(a) Margin **(b)** BADGE

(c) Confidence **(d)** ActiveFT(al)

Figure 4.18 – The equivalent number of non-active learning label amounts comparison of the standard method, SVPp, and our method ASVP on the Pets using (a) Margin, (b) BADGE, (c) Confidence and (d) ActiveFT(al) sampling. The equivalent non-active learning label amount refers to the number of samples selected by a random baseline to achieve the same accuracy as active learning.



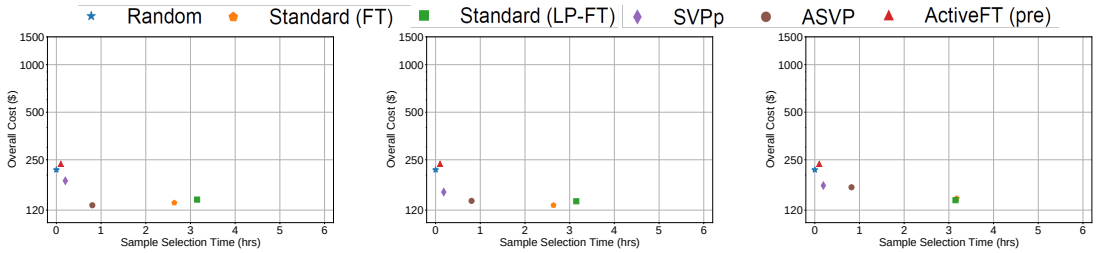
(a) Margin **(b)** BADGE **(c)** Confidence

Figure 4.19 – Computation efficiency and overall cost comparison of the standard active learning method, SVPp, and our method ASVP on ImageNet using (a) Margin, (b) BADGE, and (c) Confidence sampling.

better, our approach still aids in selecting a preferable training method.

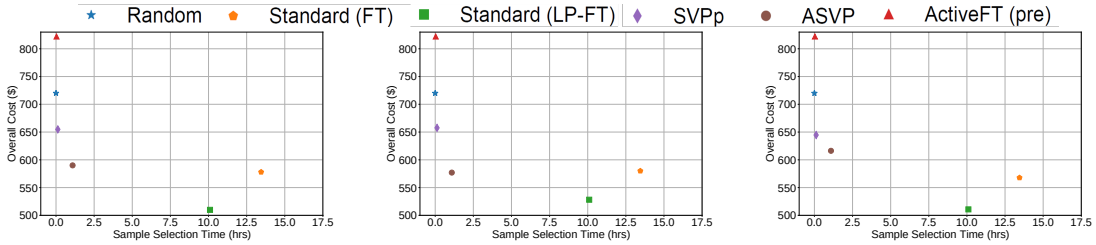
Influence from Position of Updating Features

We examined the impact of the position of updating pre-computed features on CIFAR-



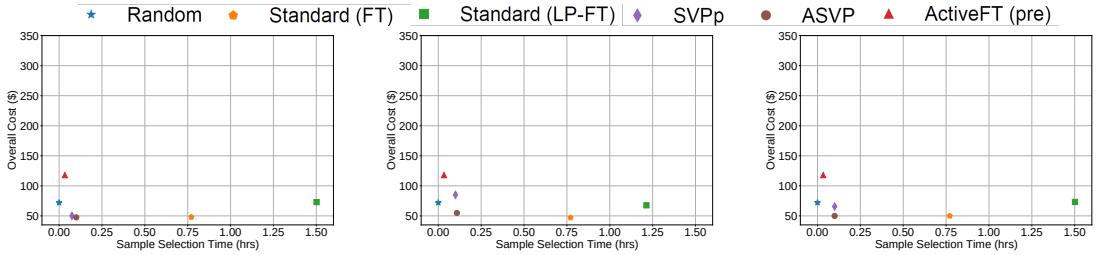
(a) Margin **(b)** BADGE **(c)** Confidence

Figure 4.20 – Computation efficiency and overall cost comparison of the standard active learning method, SVPp, and our method ASVP on CIFAR-10 using (a) Margin, (b) BADGE and (c) Confidence sampling.



(a) Margin **(b)** BADGE **(c)** Confidence

Figure 4.21 – Computation efficiency and overall cost comparison of the standard active learning method, SVPp, and our method ASVP on CIFAR-100 using (a) Margin, (b) BADGE and (c) Confidence sampling.



(a) Margin **(b)** BADGE **(c)** Confidence

Figure 4.22 – Computation efficiency and overall cost comparison of the standard active learning method, SVPp, and our method ASVP on Pets using (a) Margin, (b) BADGE and (c) Confidence sampling.

10. The number of saved samples throughout the entire active learning iteration process is depicted in fig. 4.23, with the average savings ratio presented in table 4.10. The choice of updating pre-computed features does indeed affect the final performance of ASVP. Nevertheless, across the six different positions explored in the experiments, ASVP consistently outperforms the method relying solely on pre-training features

Feature alignment	Training alignment	CIFAR-10	CIFAR-100	Pets
No	No	23.63±4.01	1.36±1.45	23.48±4.09
No	Yes	24.96±2.74	7.69±1.56	18.45±3.89
Yes	No	31.42±2.65	6.98±2.03	27.01±0.51
Yes	Yes	35.03±1.89	13.31±1.86	21.98±0.50

Table 4.6 – Ablation Study. Comparison of the average sample saving ratio of solely updating pre-computed features (training the final model using fine-tuning), solely switching training methods (using pre-trained features), and the complete ASVP approach (both updating pre-computed features and switching training methods) under different datasets with margin sampling.

Feature alignment	Training alignment	CIFAR-10	CIFAR-100	Pets
No	No	14.96±5.89	-2.11±0.85	-7.78±2.42
No	Yes	18.3±4.68	3.81±0.44	-15.61±1.93
Yes	No	30.34±2.06	2.25±2.25	13.93±5.21
Yes	Yes	33.67±0.84	8.16±1.98	6.1±5.59

Table 4.7 – Ablation Study: Comparing the average sample saving ratio using the active learning strategy BADGE.

Feature alignment	Training alignment	CIFAR-10	CIFAR-100	Pets
No	No	-11.54±2.3	-14.43±2.49	2.4±7.25
No	Yes	-9.61±1.43	-7.86±2.16	-0.83±7.34
Yes	No	21.99±3.32	-7.19±1.69	18.05±2.92
Yes	Yes	23.92±5.16	-0.62±1.43	14.81±3.17

Table 4.8 – Ablation Study: Comparing the average sample saving ratio using the active learning strategy confidence.

(SVPp). Furthermore, our proposed method for determining the update location based on LogME-PED proves to be effective. On CIFAR-10, this method selects the update of pre-computed features at 600 labels, ranking as the second-best among the six experimental positions, as shown in table 4.10.

The ASVP method uses the difference in LogME-PED scores between consecutive active learning iterations to decide if updating pre-computed features is required.

Feature alignment	Training alignment	CIFAR-10	CIFAR-100	Pets
No	No	-4.72±5.78	0.25±0.92	-11.63±6.85
No	Yes	-1.89±6.12	5.66±0.97	-14.76±6.74
Yes	No	24.62±1.31	5.86±2.24	-4.61±4.97
Yes	Yes	27.45±1.42	11.28±2.01	-7.74±4.81

Table 4.9 – Ablation Study: Comparing the average sample saving ratio using the active learning strategy ActiveFT(al).

Change at	Saving Ratio	Change at	Saving Ratio
w/o	23.63±4.01		
200	33.13±2.32	800	36.03±1.48
400	32.13±4.60	1000	32.75±1.91
600	35.03±1.89	1200	31.33±1.87

Table 4.10 – The influence of the position of updating pre-computed feature on average sample savings ratios on CIFAR-10. The best results are shown in red and the second-best results are in blue.

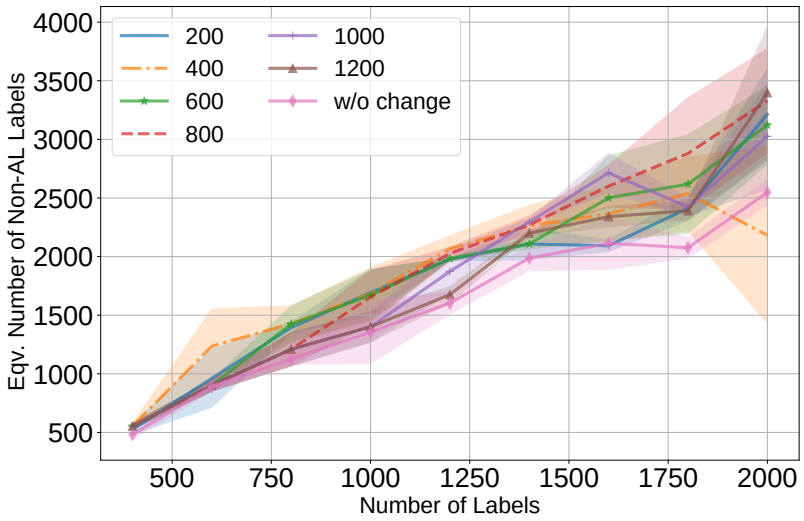


Figure 4.23 – The impact of the position of updating pre-computed features on the savings of label amounts in CIFAR-10 across each active learning iteration. The equivalent non-active learning label amount refers to the number of samples selected by a random baseline to achieve the same accuracy as active learning

The absolute differences in LogME-PED scores are illustrated in fig. 4.24, fig. 4.25, fig. 4.26. When the threshold is defined as 1, the updating positions are detailed in

table 4.11.

	Updating Features	Position
ImageNet	No	$\geq$ 100k
CIFAR-10	Yes	600
CIFAR-100	Yes	2000
Pets	Yes	800

Table 4.11 – Positions of updating pre-computed features. The number of the PED convergence iterations is used to determine if updating pre-computed features is required. The positions are estimated by the absolute difference in LogME-PED scores between consecutive active learning iterations, where a difference smaller than 1 indicates an update.

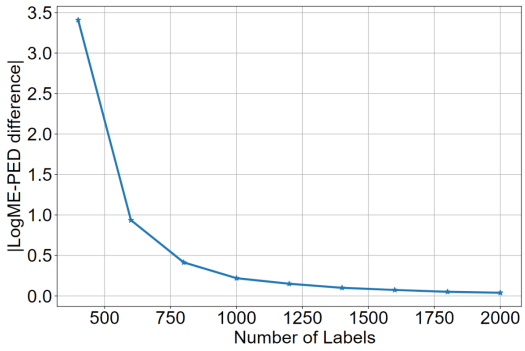


Figure 4.24 – The absolute differences in LogME-PED scores between consecutive active learning iterations on CIFAR-10.

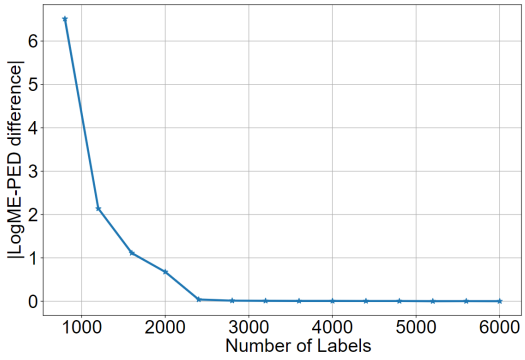


Figure 4.25 – The absolute differences in LogME-PED scores between consecutive active learning iterations on CIFAR-100.

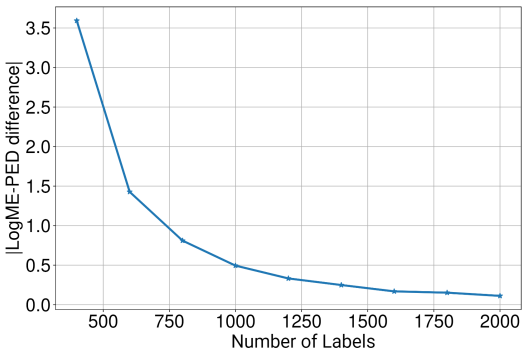


Figure 4.26 – The absolute differences in LogME-PED scores between consecutive active learning iterations on Pets.

4.5.6 Improvement from Small Batchsize

Training deep learning models takes much longer than traditional machine learning models like SVM (Support Vector Machine). When applying active learning, multiple rounds of training deep learning models imply long computation times. As a result, existing methods mostly compromise between active learning performance and sample selection time by selecting a large batch of samples Citovsky et al. (2021); Gal et al. (2017). However, batch selection often leads to the selection of similar samples, weakening active learning performance as mentioned in Chapter 2.3.2. To address this issue, different active learning strategies have been developed to improve the diversity of selected samples Ash et al. (2020); Shui et al. (2020).

This problem can also be addressed through efficient active learning. ASVP trains a simple proxy model during the sample selection stage, resulting in a short computation time. This means that ASVP can select diverse samples by increasing the number of active learning iterations. To demonstrate the effectiveness of ASVP in this regard, we conducted small-batch experiments. Specifically, we conducted experiments on ImageNet, randomly selecting an initial pool of 10,000 samples. Subsequently, we performed 9, 45, 180, and 900 active learning iterations using margin sampling, selecting a total of 100,000 samples. The accuracy of the final model and the total sample selection time for both FT and LP-FT training are presented in table 4.12. The results show a general improvement in active learning performance with an increase in the number of iterations. However, beyond 180 iterations, the performance

improvement diminishes, indicating a limit to the benefits of increasing the number of active learning iterations.

Additionally, the accuracy of SVPp (final model is fine-tuned) with more active learning iterations remains lower than that of the standard active learning pipeline. However, when using LP-FT for final model training and increasing the iteration count to 45 or more, ASVP surpassed the standard active learning method in terms of accuracy.

# AL iterations	Training Method FT	LP-FT	Sampling Time (hrs)
9	16.64±2.06	30.22±1.85	1.1
45	17.66±1.37	31.20±0.40	1.5
180	19.43±0.86	32.00±0.46	3.7
900	18.21±0.67	31.64±0.24	15.8

Table 4.12 – The influence of the number of active learning iterations on average sample savings ratios and sampling time on ImageNet.

4.6 Limitations and Discussions

Due to limited computational resources, this paper evaluates efficient active learning methods' labeling and training costs using a ResNet-50 model with a specific pre-training method, BYOL-EMAN (Cai et al., 2021; Grill et al., 2020). Although ResNet-50 serves as a practical choice for our experiments and provides a reliable baseline, larger models achieve higher accuracy and are often more helpful in practical applications (Zhai et al., 2022). Existing research shows that larger models exhibit better label efficiency, requiring fewer labeled samples to reach the same accuracy as smaller models (Chen et al., 2020b). This implies that in active learning, using larger models may reduce labeling costs while increasing training costs. Therefore, the total cost savings achieved by both standard and efficient active learning methods are closely tied to model size. Evaluating the impact of model size on total cost savings, and exploring efficient active learning methods that can effectively reduce sample

selection time and total cost (labeling and training costs) in the context of larger models, are important directions for future research.

Furthermore, when transferring to downstream datasets (used for active learning), different pre-trained models exhibit varying performance gaps between fine-tuning and linear probing. A smaller gap suggests higher-quality pre-trained features, making it easier for proxy models based on these features to select samples similar to those chosen by fine-tuned models. This means that proxy model-based active learning may compromise less on performance compared to standard active learning. Therefore, exploring how to select suitable pre-trained models is another interesting direction for future research.

4.7 Summary

Active learning on pre-trained models shows label efficiency but demands significant computational time, whereas current efficient active learning methods tend to compromise a notable fraction of active learning performance, resulting in increased overall costs. To address this challenge, this chapter attributes the reduced performance of SVPp to (1) the absence of samples necessary for preserving valuable pre-trained information and (2) an increased selection of redundant samples due to fine-tuned features becoming better than pre-trained features as the annotation increases. Building upon this analysis, we propose ASVP, a two-fold alignment approach to improve the effectiveness of the efficient active learning method with similar or marginally increased computational time. Additionally, we introduce the sample savings ratio as a metric to assess the effectiveness of efficient active learning, providing a straightforward measure for labeling cost savings. Experiments show that our proposed ASVP saves similar or greater total costs in most cases, while still maintaining computational efficiency.

Chapter 5

Training from a Better Start Point: Active Self-Semi-Supevivsed Learning

5.1 Introduction

Part of the success of deep learning models arises from large amounts of labeled data (Deng et al., 2009; He et al., 2016; Liu et al., 2021). However, the high cost associated with acquiring large numbers of labels hinders the widespread application of deep learning models. This challenge is particularly pronounced in domains that demand expert annotations, such as medical images (Yang et al., 2017; Zhang et al., 2022c), or biology images (Alonso et al., 2019; Bewley et al., 2015). In response, researchers have dedicated considerable effort to exploring semi-supervised learning. Recent works (Sohn et al., 2020; Zhang et al., 2021) have shown that these techniques can achieve similar accuracy to supervised learning with fewer annotations.

However, existing semi-supervised learning techniques face bottlenecks in reducing labeled samples (Kim and Lee, 2022). This is due to the common practice of employing model predictions as pseudo-labels for unlabeled samples, which are not always

accurate. Training with these noisy pseudo-labels boosts the model's confidence in incorrect predictions, inducing the model to resist the correct information, and consequently compromising the performance of semi-supervised training. This issue is commonly referred to as confirmation bias (Arazo et al., 2020). To unlock the full potential of semi-supervised learning, many current techniques still rely on a substantial number of annotations to rectify these incorrect pseudo-labels. For instance, prevalent benchmarks in semi-supervised learning typically opt for 25, 100, and 400 annotations per class (Berthelot et al., 2019, 2020; Rizve et al., 2021; Xie et al., 2020), with some recent studies gradually reducing to 4 annotations per class (Li et al., 2021; Zhang et al., 2021). As the number of annotations decreases, the semi-supervised learning struggles to figs/correct erroneous pseudo-labels, resulting in a significant decline in its performance, as illustrated in Fig. 5.1.

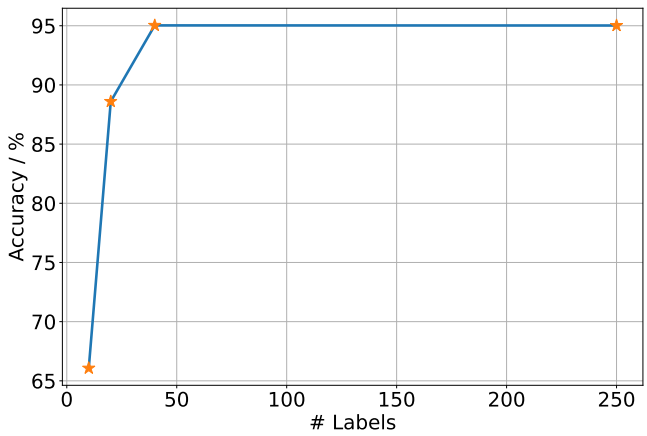


Figure 5.1 – Steep performance drops when reducing annotations. Accuracy trained with state-of-the-art semi-supervised learning algorithm, Flexmatch (Zhang et al., 2021), with different numbers of annotations. Experiments were performed on CIFAR-10. The network architecture adopts WRN-28-2, the details of semi-supervised learning settings follow (Zhang et al., 2021).

Conducting semi-supervised training on top of self-supervised models holds great promise in improving this issue. Self-supervised training yields valuable representations for downstream tasks without any labels (Cai et al., 2021; Caron et al., 2020). Initializing with a self-supervised model allows the semi-supervised model to swiftly generate accurate pseudo-labels and reduce the influence of erroneous pseudo-

labels (Kim et al., 2021a), thereby enhancing final performance. However, with limited labeled samples, the semi-supervised model may not rapidly generate accurate pseudo-labels. The noisy pseudo-labels disrupt valuable information obtained from self-supervised learning (Sec. 5.2), consequently diminishing the performance gains achieved through the initialization with self-supervised learning. In some cases, this initialization might not yield any performance improvements.

Motivated by this observation, we propose a more explicit method to transfer valuable information from the self-supervised model to the semi-supervised model. Specially, we propose Active Self-Semi-Supervised Learning (AS3L) to guide semi-supervised learning using **Prior Pseudo-Labels (PPL)**, which are obtained through label propagation on self-supervised representations (Sec. 5.3.3). To differentiate these from pseudo-labels generated by model predictions during semi-supervised training, we term the ones obtained before such semi-supervised training as PPL. Also, we employ a switching mechanism to integrate PPL and model predictions early in semi-supervised training and remove the PPL later in training (Sec. 5.3.2). This serves both to give the semi-supervised model a good starting point and to prevent PPL from hindering the continuous updating of the pseudo-labels during the semi-supervised training process.

Moreover, given that the accuracy of PPL relies not only on feature quality but also on labeled sample selection, especially when annotations are limited, we propose an active learning strategy. This strategy aims to leverage self-supervised representations to generate as accurate PPL as possible by selecting labeled samples (Sec. 5.3.4).

Validated across four image classification datasets, our AS3L outperforms the state-of-the-art in most cases, particularly in scenarios with limited labels, and showcases accelerated convergence (Sec. 5.4).

The contributions of our work are summarized as follows:

1) In scenarios with scarce labeled data, our observation reveals that semi-supervised learning cannot effectively harness valuable information obtained from self-supervised training by weight initialization.

2) Motivated by the observation, we propose the AS3L framework, a novel approach to bootstrap semi-supervised training from a good initial point consisting of accurate PPL, actively selected labeled samples, and weights initialization from self-supervised training. This method readily integrates with existing semi-supervised learning approaches to extend the success of semi-supervised learning to scenarios with even fewer labeled data. Our proposed method outperforms SOTA algorithms in most cases with limited annotations.

3) We develop an active learning strategy that is tightly coupled to the AS3L framework, which can greatly improve the accuracy of PPL when there are few labeled samples, thereby helping AS3L work well with limited labels.

5.2 Observations

This section explores whether semi-supervised training can effectively harness valuable information acquired from self-supervised pre-training through weight initialization. Here, the framework of fine-tuning self-supervised pre-trained models using semi-supervised training methods is also referred to as SelfMatch (Kim et al., 2021a). Specifically, we initialize a semi-supervised model using weights pre-trained with the self-supervised method, Simsiam (Chen and He, 2021). Subsequently, we conduct semi-supervised training using FlexMatch (Zhang et al., 2021), one of the state-of-the-art semi-supervised learning algorithms. The semi-supervised training is executed on CIFAR-10 (Krizhevsky et al., 2009) with 10 and 40 labeled samples respectively, and on CIFAR-100 (Krizhevsky et al., 2009) with 200 and 400 labeled samples, where all the labeled samples are selected randomly. The network architecture and hyperparameters for semi-supervised training align with the prior work (Zhang et al., 2021).

Self-supervised pre-training weight initialization allows downstream tasks to start training in a well-established pre-trained feature space (Chen et al., 2020a; Grill et al., 2020), thereby improving the performance of downstream tasks. Therefore, the quality of features is a crucial indicator to assess whether semi-supervised training retains

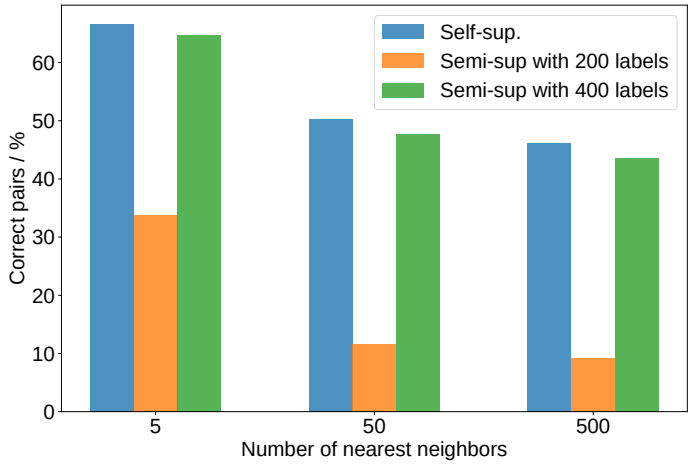


Figure 5.2 – In the CIFAR-100 dataset, consistency of sample labels with neighboring sample labels in self-supervised and semi-supervised feature spaces. The model was initialized with the self-supervised pre-training weights and then further trained using FlexMatch with 200 and 400 labels, respectively.

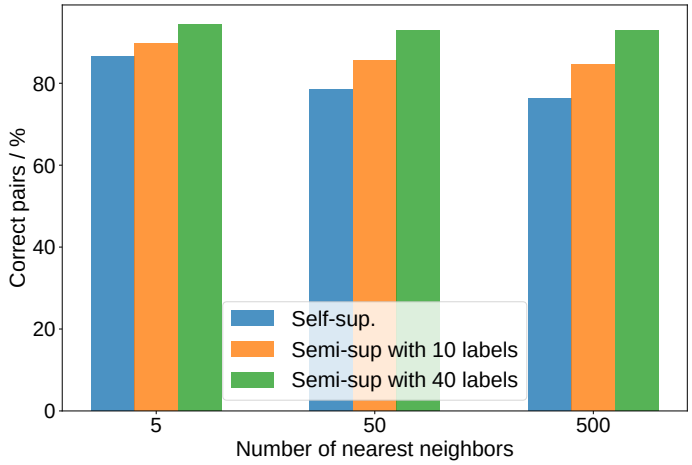


Figure 5.3 – In the CIFAR-10 dataset, consistency of sample labels with neighboring sample labels in self-supervised and semi-supervised feature spaces. The model was initialized with the self-supervised pre-training weights and then further trained using FlexMatch with 10 and 40 labels, respectively.

valuable information obtained from self-supervised training. Following (Van Gansbeke et al., 2020), we compute the degree of consistency between the label of the sample and the label of its nearest neighbors within the feature space. For the CIFAR-100 dataset, with few labeled samples, semi-supervised training undermines the valuable information obtained from self-supervised pre-training. As depicted in Fig. 5.2, the

label consistency between samples and their nearest neighbor samples in the semi-supervised feature space notably declines compared to that in the self-supervised feature space.

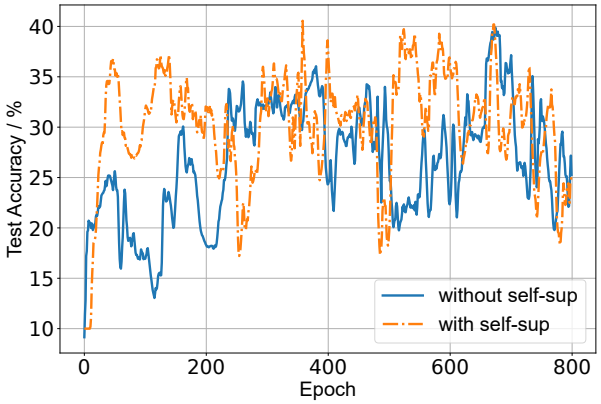


Figure 5.4 – Test Accuracy of the semi-supervised models initialized with self-supervised pre-training and random initialization. Specifically, semi-supervised training utilizing 10 labeled samples on CIFAR-10. Semi-supervised training method employed: FlexMatch.

For the simpler dataset, CIFAR-10, semi-supervised learning can generate features superior to those from self-supervised pre-training, as shown in Fig. 5.3. Yet, in scenarios with fewer annotations (10 labeled samples), semi-supervised training rarely benefits from self-supervised initialization. As illustrated in Fig. 5.4, during the initial phase of semi-supervised training (approximately the first 200 epochs), self-supervised initialization enhances the performance of the semi-supervised model. However, as semi-supervised training progresses, the test accuracy of models initialized randomly and those using self-supervised pre-training initialization becomes nearly identical.

Table 5.1 – Impact of self-supervised pre-training weight initialization on semi-supervised performance. The experiments adopted the semi-supervised training method, FlexMatch.

Self-sup.	CIFAR-10		CIFAR-100	
Initialization	10 labels	40 labels	200 labels	400 labels
None	59.06±19.80	94.86±0.05	30.59±1.69	46.11±2.83
Simsiam	57.03±14.86	94.63±0.34	28.63±4.13	51.24±1.02

Additionally, we demonstrated the impact of initializing with self-supervised pre-training weights on semi-supervised training performance, as shown in Table 5.1, indicating that it does not improve performance when labeled data is limited. In conclusion, weight initialization struggles to effectively transmit valuable information obtained from self-supervised training to the semi-supervised model. To address this issue, we propose to construct PPL, serving as an intermediary to facilitate semi-supervised models in fully leveraging the valuable information acquired from self-supervised pre-training.

5.3 Method

In this section, we begin by framing the problem and introducing the basic framework of semi-supervised learning in section 5.3.1. Subsequently, in sec. 5.3.2, we elaborate on leveraging PPL as an intermediary to facilitate the transfer of valuable information from self-supervised pre-training models to semi-supervised models. And sec. 5.3.3 details the generation of PPL through label propagation on the self-supervised features, f_{self}. Finally, in sec. 5.3.4, we propose an active learning strategy aimed at improving the accuracy of PPL.

5.3.1 Preliminaries on Semi-Supervised Training

Semi-supervised learning trains a model, denoted as $\mathrm{M}(\cdot)$, which comprises a feature extractor and a classification head. This training process leverages both a labeled dataset, $D_l = \{(x_i, y_i)\}_{i=1}^{N_1}$, and an unlabeled dataset, $D_u = \{(x_i)\}_{i=N_1+1}^{N_1+N_2}$. The objective is to achieve superior performance compared to training exclusively on the labeled dataset. Typically, $N_2 \gg N_1$. The loss function for semi-supervised training, as depicted in Eq. (5.1), consists of two components: the cross-entropy loss on labeled samples, L_l, and the consistency loss on unlabeled samples, L_{un}. The hyper-parameter λ governs the trade-off between these two losses. The L_l, as illustrated in Eq. (5.2), represents the cross-entropy loss, CE, on weakly augmented labeled sam-

ples, where weak augmentation, A_w, follows the definition in previous research (Sohn et al., 2020), incorporating operations such as image flipping.

$$L = L_l + \lambda L_{un} \tag{5.1}$$

$$L_l = CE(M(A_w(x_l)), y_l) \tag{5.2}$$

The basic consistency loss, L_{un}, serves two purposes: using high-confidence model predictions as pseudo-labels to train unlabeled data and encouraging consistent predictions for inputs subjected to perturbations (image augmentations). As formulated in Eq. (5.3), it involves performing both weak augmentation, A_w, and strong augmentation, A_s, on unlabeled samples. Then, the model predicts unlabeled samples with weak augmentation and strong augmentation as $y_{pre,w} = M(A_w(x_u))$ and $y_{pre,s} = M(A_s(x_u))$, respectively. Here, μ is a ratio of the number of unlabeled samples and labeled samples in each training batch, and B is batch size. $\mathbb{I}$ is an indicator function used to identify reliable predictions. When the confidence of its input surpasses the specified threshold τ, the $\mathbb{I}$ returns 1; otherwise, it outputs 0. Additionally, the definition of A_s aligns with that in prior research (Zhang et al., 2021), including techniques such as cutout.

$$L_{un} = \frac{1}{\mu B} \sum_{b=1}^{\mu B} \mathbb{I}(max(y_{pre,w}))CE(y_{pre,s}, argmax(y_{pre,w})) \tag{5.3}$$

5.3.2 Active Self-Semi-Supervised Learning Framework

Existing methods typically initialize the model using weights pre-trained through self-supervised learning and then proceed with semi-supervised training to enhance performance. However, as demonstrated in sec. 5.2, as the number of labeled samples diminishes, semi-supervised training quickly disrupts valuable information from self-supervised training, resulting in inferior quality of semi-supervised features (as shown

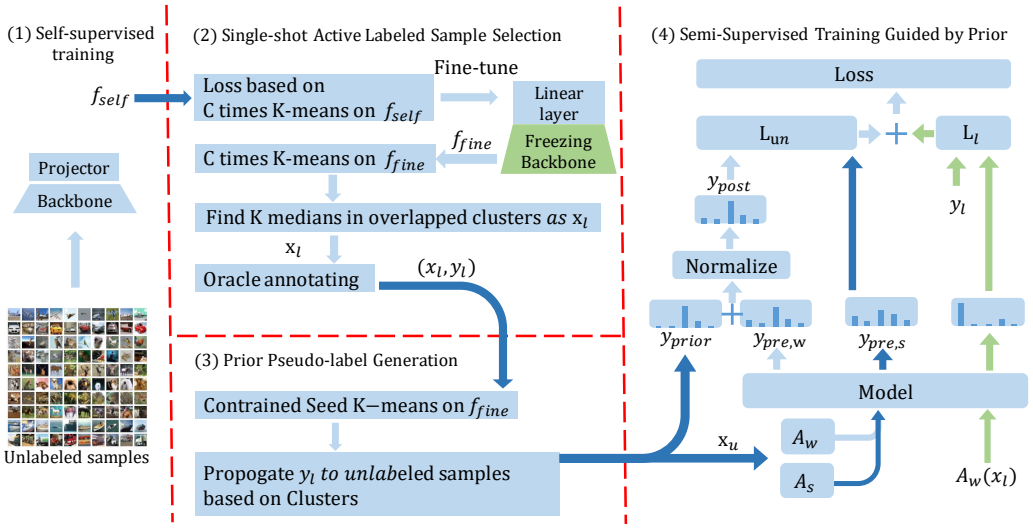


Figure 5.5 – The framework of our Active Self-Semi-Supervised learning(AS3L). AS3L consists of four components: (1) Obtaining self-supervised feature f_{self} follows (Chen and He, 2021); (2) Selecting labeled samples based on f_{self} (Sec. 5.3.4); (3) Label Propagation based on clusters to get PPL y_{prior} (Sec. 5.3.3); (4) Semi-supervised training guided by y_{prior} (Sec. 5.3.2).

fig. 5.2). To better utilize self-supervised pre-training information and improve model performance, we propose extracting and transferring information from pre-trained features to semi-supervised training through pseudo-labels. This approach ensures that valuable information obtained from pre-training is preserved and effectively used, even as the model undergoes weight updates.

As depicted in Fig. 5.5, our framework leverages various techniques. Initially, the entire dataset $D = D_l \cup D_u$ undergoes self-supervised training to generate feature representations f_{self}. Our active learning strategy then selects samples and queries their labels to construct the labeled dataset D_l based on f_{self}. After that, we propagate labels in the pre-trained feature space to obtain a group of pseudo-labels. These pseudo-labels serve as intermediaries to transfer valuable information from self-supervised pre-training to semi-supervised models. Given that these pseudo-labels are obtained before semi-supervised training, we refer to them as prior pseudo-labels (PPL), denoted as y_{prior}. The PPL is used for the unlabeled dataset $D_u = \{(x_i, y_{prior,i})\}_{i=N_1+1}^{N_1+N_2}$.

Finally, a semi-supervised model is trained by combining these PPL with the model predictions, resulting in enhanced overall performance.

$$L_{un} = \frac{1}{\mu B} \sum_{b=1}^{\mu B} \mathbb{I}(max(y_{post}))CE(y_{pre,s}, argmax(y_{post})) \tag{5.4}$$

$$y_{post} = \begin{cases} normalize(y_{prior} + y_{pre,w}) & t \leq T \\ y_{pre,w} & t > T \end{cases} \tag{5.5}$$

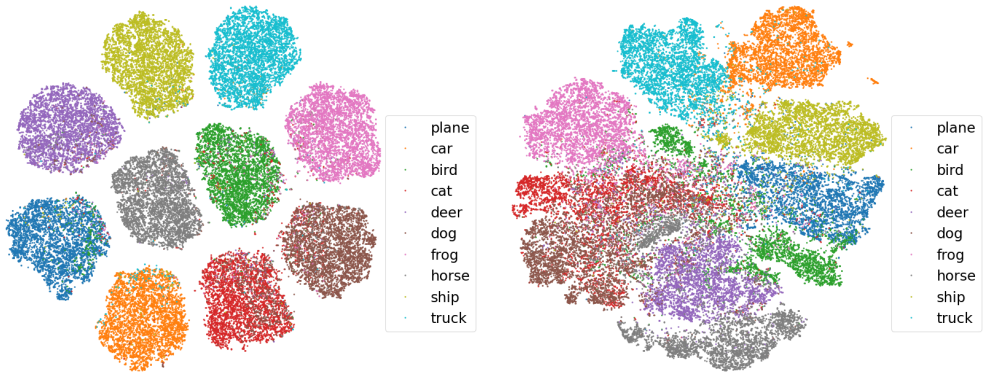
Semi-Supervised Training Guided by PPL: Assuming we have obtained prior pseudo-labels, y_{prior}, through label propagation on f_{self} (which will be discussed in the next section), we integrate y_{prior} into the existing semi-supervised training framework by re-formulating the consistency loss as Eq. (5.4). Instead of using the model prediction, $y_{pre,w}$, as the target for training unlabeled data, we combine y_{prior} with $y_{pre,w}$ to generate a new training target, y_{post}, as given in Eq. (5.5). Since this target is derived from both the semi-supervised model and the prior pseudo-labels, we slightly abuse the term and refer to it as posterior pseudo-labels.

In Eq. (5.5), T denotes a pre-defined switching point. This adjustment ensures that, during the early stages of semi-supervised training (i.e., before T training iterations) when the model prediction is less accurate, y_{prior} helps to guide the semi-supervised training. As the model predictions become more accurate than y_{prior}, the framework uses model predictions as pseudo-labels, resembling a conventional semi-supervised training approach. This transition helps mitigate the impact of PPL inaccuracies on the semi-supervised training. We empirically found that setting a rough switching point T yields favorable results, as discussed in sec. 5.4.3.

5.3.3 Prior Pseudo-label Generation

We generate PPL y_{prior} by applying label propagation based on clusters. Instead of using traditional K-means, we employ constrained seed K-Means (Basu et al., 2002) to leverage labeled sample constraints, which enhances the quality of clustering. The

labels of the labeled samples are propagated to all unlabeled samples within the same cluster and then normalized to derive PPL.



(a) Semi-Supervised feature f_{semi} **(b)** Self-Supervised feature f_{self}

Figure 5.6 – T-SNE visualization of semi-supervised and self-supervised features, where semi-supervised features are trained with 40 labels on CIFAR-10 followed our method. Self-supervised features seem to be more loosely clustered, and the boundaries between clusters are not as well defined. (a) Semi-Supervised feature f_{semi}. (b) Self-Supervised feature f_{self}.

It is worth noting that as the number of clusters K increases, the likelihood of samples within the same cluster sharing the same label also increases. Therefore, to improve the accuracy of PPL, it is advisable to increase K. However, as K rises, the number of samples per cluster decreases, leading to an increase in the number of clusters devoid of any labeled samples. This is particularly evident when K surpasses the number of labeled samples, resulting in a larger proportion of unlabeled samples not being associated with any labels. As a trade-off, we adopt a strategy of using different values of K for a total of C clustering iterations. This approach ensures that the majority of unlabeled samples are encompassed by labeled samples, while those situated farther away from any labeled samples exhibit lower confidence levels.

5.3.4 Single-shot Active Labeled Sample Selection

Our active learning strategy primarily focuses on improving the accuracy of PPL, which facilitates the transfer of valuable information from a self-supervised model to

a semi-supervised model. For this purpose, existing active learning methods based on uncertainty and diversity principles are not suitable, as they tend to select challenging samples that exhibit high uncertainty (Gao et al., 2020) or are distant from the feature space of labeled samples (Sener and Savarese, 2018). These challenging samples often imply difficulty in differentiation within the self-supervised features, f_{self}. In other words, these samples are likely to have different labels from their neighbors (samples with similar features), leading to lower PPL accuracy when using these samples for label propagation.

In contrast, selecting representative samples (Settles, 2009) is more appropriate for our scenario. Since our PPL is derived through clustering-based label propagation, in scenarios with limited annotations, the PPL for all samples within a cluster is determined by the labeled samples within the same cluster. In other words, a relatively small number of labeled samples play a crucial role in determining the PPL for the entire cluster. Therefore, we aim to select labeled samples that well represent the majority of samples within their cluster, ensuring that the selected labeled samples have consistent labels with most samples in their cluster.

Building upon the observations (Sec. 5.4.3) that within the self-supervised feature space, f_{self}, samples located in the same cluster tend to have similar labels, and those near the center of the cluster are more likely to share the same label as the majority of samples in that cluster. Consequently, our active strategy selects samples close to the center of the cluster to query labels.

Fine-tuning features

While self-supervised training has been established as effective in producing high-quality features, f_{self}, with excellent performance in linear evaluation (Chen et al., 2020a; Chen and He, 2021), we experimentally find that clustering directly on f_{self} is not a good choice. One potential reason is that self-supervised features exhibit a distinct distribution compared to features trained with labeled data, as shown in fig. 5.6. Self-supervised features tend to scatter due to their training on finer-grained

proxy tasks, resulting in a larger distance between features of the same class and a smaller distance between features of different classes. This characteristic can impact the performance of the clustering algorithm and, potentially, the accuracy of our pseudo-labels. To achieve a more suitable feature space, we fine-tune these features based on multiple clusters before selecting samples for labeling.

To bring samples within the same cluster closer together, the mean squared error (MSE) loss is employed to force samples in the same cluster to be close to their centers. Also, to improve robustness, we perform C iterations of K-means on f_{self}, with the final loss as defined by eq. 5.6, where N is the number of samples within the whole dataset, $f_{fine,i}$ is i^{th} sample's fine-tuned feature, $f_{center,i,j}$ denotes the cluster center assigned to i^{th} sample during the j^{th} clustering run, where j refers to the specific iteration out of a total of C runs of K-means clustering. During the fine-tuning process, as the loss is defined based on C times clustering with randomness, those samples that are stable within the same cluster become closer, while those attracted by different cluster centers do not approach any specific center, thereby enhancing the clustering results.

$$Loss = \frac{1}{CN} \sum_{i} \sum_{j} \| f_{fine,i} - f_{center,i,j} \|_2^2 \tag{5.6}$$

Additionally, considering the computational cost while still aiming to retain the approximate self-supervised feature structure, we add a single linear layer to the self-supervised trained encoder. During training, the encoder weights are frozen and only the weights of the newly added linear layer are updated. To reduce the computational cost, we pre-compute the self-supervised features for all samples. This allows us to directly use the f_{self} as input during the training of the new linear layer, eliminating the need to forward-pass through the backbone in each training iteration. Finally, we select labeled samples and generate PPL based on the output of the linear layer, denoted as f_{fine}.

Select Labeled Set Based on Multiple Clusters

To improve robustness, conduct C rounds of K-means clustering on f_{fine}. Subsequently, identify samples consistently assigned to the same class across all C clustering rounds. Calculate the mean of the features for these samples to represent the center of that class. Then, select and annotate as the labeled set the samples that are closest to these K class centers.

5.4 Results

5.4.1 Results on CIFAR-10, CIFAR-100 and STL-10

Our method is first evaluated on the common benchmark for semi-supervised learning: CIFAR-10 (Krizhevsky et al., 2009), CIFAR-100 (Krizhevsky et al., 2009), STL-10 (Coates et al., 2011). Both CIFAR-10 and CIFAR-100 contain 50,000 training samples and 10,000 test samples, all with image size of 32×32. STL-10 contains 5,000 labeled training samples, 100,000 unlabeled training samples and 8,000 test samples. The resolution of all samples is 96×96. We experimented with labeled sample sets of various sizes, especially with fewer annotation samples than in previous papers (10 labeled samples for CIFAR-10, 200 labeled samples for CIFAR-100, and 20 labeled samples for STL-10).

Baseline Methods

We compare our method with (1) Semi-supervised learning from scratch algorithms: Fixmatch (Sohn et al., 2020), Flexmatch (Zhang et al., 2021) and CoMatch (Li et al., 2021). (2) Semi-supervised learning initialized with the self-supervised model: SelfMatch (Kim et al., 2021a). SelfMatch refers to a training framework that utilizes standard semi-supervised training methods to fine-tune models initialized with self-supervised pre-training weights. This framework applies to various self-supervised pre-training and semi-supervised training methods. To set a fair baseline, we employ

identical semi-supervised (FlexMatch) and self-supervised training (SimSiam (Chen and He, 2021)) methods for SelfMatch as those utilized in our framework. (3) Plug-in model selection method, Sel (Kim and Lee, 2022), that is specially optimized for semi-supervised training with few annotations. (4) Constructing self-supervised signals to enhance semi-supervised training in scenarios with limited annotations, Less (Lucas et al., 2022). The annotated samples for the aforementioned baseline methods were randomly selected. (5) Active semi-supervised learning method: USL (Wang et al., 2022a) conducts semi-supervised training by actively selecting labeled samples. To establish a fair baseline, we utilize the same self-supervised pre-training weight initialization for our method, SelfMatch, and USL.

Implemention details

Many existing semi-supervised learning methods involve the use of pseudo-labeling techniques. Our method readily integrates with these techniques. In our implementation, we conduct experiments with Flexmatch, one of the state-of-the-art semi-supervised approaches. For the self-supervised pre-training, any self-supervised method can be employed. For this study, we chose SimSiam (Chen and He, 2021), a state-of-the-art self-supervised method, using the same backbone as in the semi-supervised learning stage. We adhere to the hyper-parameter settings proposed by SimSiam (Chen and He, 2021). The network weights obtained from self-supervised learning serve as the initialization for the backbone of the semi-supervised model.

We set network architectures following previous work (Zhang et al., 2021): WRN-28-2 (Zagoruyko and Komodakis, 2016) for CIFAR-10, WRN-28-8 for CIFAR-100 and WRN-37-2 for STL-10. For CIFAR-10, CIFAR-100 and STL-10. We set hyper-parameters following TorchSSL (Zhang et al., 2021): SGD with momentum 0.9, initial learning rate 0.03, λ is 1, μ is 7, the batch size is 64, total training iterations is 1024×1024 and a cosine annealing learning rate scheduler.

The hyper-parameters involved in the active learning and label propagation in this chapter include the number of times clustering was performed, C, and the number

of classes in each clustering, K. Specifically, clustering is performed 6 times (C is set to 6) in both the active sampling strategy and label propagation. For active sampling, the number of classes in each clustering, K, is equal to the number of selected samples. In label propagation, clustering is accomplished using Constrained Seed K-Means (Basu et al., 2002). The minimum value of K is set to match the number of categories in the dataset. For CIFAR-10 and STL-10, K takes values of 10, 20, 30, 40, 50, and 60. For CIFAR-100, K is set to 100, 200, 300, 400, 500, and 600, respectively. The linear layer mentioned in Sec. 5.3.4 has the same dimension as the final layer of the backbone, and fine-tuning features are trained for 40 epochs. Additionally, the T for switching from PPL-guided pseudo-labels to regular pseudo-labels during semi-supervised training is set to the 60th epoch.

Results

The experimental results are shown in Table 5.2. Given the lack of labeled data for semi-supervised training, selecting the best model by validation set is infeasible. Following previous work (Kim and Lee, 2022), we report the median accuracy for the last 20 checkpoints. Notably, "Sel" denotes the use of the metric proposed by Kim et al. (Kim and Lee, 2022) to select the final model for evaluation. Our results are presented in 2 ways: first, the median accuracy of the last 20 models as "Ours" and second, the accuracy of the models selected using "Sel" (Kim and Lee, 2022) as "Ours-Sel", where the total training iteration is 1024×1024. Additionally, to demonstrate the accelerated convergence of our method, we report the accuracy achieved with only 1/4 of the total training iterations, termed as "Ours-Early-Sel".

Effectiveness: Our approach demonstrates significant improvements over most baselines, except for LESS, which shows similar performance under some experimental setups. Notably, the results with LESS were obtained under somewhat unfair conditions. LESS selects labeled samples through random stratified sampling (i.e., randomly selecting an equal number of samples per class), ensuring that all classes are covered even with limited annotations, which requires additional knowledge about the dataset. In contrast, our method selects samples without prior knowledge of the

Table 5.2 – Comparison of accuracy on CIFAR-10, CIFAR-100 and STL-10. All results are averaged over 3 runs. The best results are shown in red and the second best results are shown in blue.

	Active Learning	Self-sup. Initialization	CIFAR-10		CIFAR-100		STL-10	
			10 labels	40 labels	200 labels	400 labels	20 labels	40 labels
Fully-Supervised	-	-	95.38		80.70		-	
FixMatch	No	None	60.01±7.41	85.57±5.21	35.83±1.63	46.04±1.41	43.24±6.32	60.92±5.60
FixMatch-Sel	No	None	65.73±10.32	89.87±4.96	35.78±1.58	46.05±1.28	45.45±3.24	63.93±9.65
FlexMatch	No	None	59.06±19.80	94.86±0.05	30.59±1.69	46.11±2.83	38.31±16.69	54.90±8.06
CoMatch	No	None	65.10±7.81	92.16±4.97	32.51±1.15	41.72±2.04	-	-
LESS	No	None	64.40±10.90	93.20±2.10	42.50±3.20	51.30±2.40	48.98±5.19	64.20±5.10
SelfMatch (w FlexMatch)	No	Simsiam	57.03±14.86	94.63±0.34	28.63±4.13	51.24±1.02	44.78±1.98	57.06±5.99
SelfMatch-Sel(w FlexMatch)	No	Simsiam	59.13±15.43	94.79±0.07	36.58±3.31	53.05±1.85	45.73±4.11	63.14±9.47
USL-Sel (w FlexMatch)	Yes	Simsiam	55.76±6.45	89.22±0.79	25.31±10.32	29.73±4.77	55.90±1.33	61.94±5.14
Ours (w FlexMatch)	Yes	Simsiam	69.08±5.32	94.35±0.20	46.87±2.47	47.37±6.66	49.57±9.32	57.57±9.55
Ours-Sel (w FlexMatch)	Yes	Simsiam	83.80±10.57	94.32±0.09	48.40±5.78	59.16±2.37	51.11±6.88	61.89±7.73
Ours-Early-Sel (w FlexMatch)	Yes	Simsiam	82.17±11.09	93.72±0.22	44.80±5.07	61.50±2.00	46.32±7.77	65.15±2.44

dataset's classes, contributing to the similar results observed between LESS and our method.

A comparison between FlexMatch and SelfMatch (with FlexMatch) indicates that solely relying on self-supervised pre-training initialization does not significantly improve the performance of semi-supervised training in nearly half of the experiments. In contrast, our method, incorporating initialization, PPL, and annotated sample selection, enables a more comprehensive utilization of valuable information from self-supervised pre-training, resulting in superior overall performance.

Fast convergence: Our method demonstrates accelerated convergence by explicitly employing PPL to initiate semi-supervised training. As shown in Table 5.2, our method (Ours-Early-Sel) achieves about 97% accuracy with only 1/4 of the training iterations. It also outperforms baseline methods in most cases with just 1/4 of the training iterations. Additionally, we compared the computational time required for our method to reach the final accuracy of SelfMatch (w FlexMatch), which is initialized with self-supervised pre-trained weights. As illustrated in Table 5.3, our method achieves the final accuracy of SelfMatch in approximately 1/3 of the training time.

5.4.2 Results on ImageNet

We conducted experiments on the large-scale dataset, ImageNet-1k (Deng et al., 2009), which has 1.28 million training images and 50,000 validation images. Following

Table 5.3 – Comparison of practical running time on a single RTX 3090 GPU. SelfMatch time refers to the total runtime of complete semi-supervised training, while our method's time indicates the duration required to reach SelfMatch's final accuracy.

Dataset	CIFAR-10	CIFAR-100
SelfMatch (w FlexMatch)	98.82 (hrs)	353.69 (hrs)
Ours (w FlexMatch)	35.21 (hrs)	121.04 (hrs)

previous works (Wang et al., 2022b; Zhang et al., 2021), we adopted ResNet-50 (He et al., 2016) as backbone. Due to computational resource limitations, we set hyperparameters following the method of training based on self-supervised pre-training weights (Cai et al., 2021; Wang et al., 2022b) instead of the original FlexMatch (Zhang et al., 2021). The total training epoch is 50, with a change point T at the 4th epoch.

FlexMatch and SelfMatch served as our baseline methods. We implemented two versions of the SelfMatch, employing different semi-supervised training approaches: FlexMatch and FixMatch. For FixMatch, we employed a variant known as FixMatch-EMAN, where EMAN (Cai et al., 2021) is utilized to enhance both pre-training and semi-supervised training effectiveness. To ensure fairness, both SelfMatch and our method are initialized with the same pre-trained weights, as shown in Table 5.4. They are respectively pre-trained using BYOL (Grill et al., 2020) and BYOL-EMAN (Cai et al., 2021). The annotated samples for baseline methods were randomly selected, with a difference from existing semi-supervised learning experimental setups. In contrast to the common practice of randomly selecting N labeled samples per class, our experiments were conducted in a more realistic setting. Specifically, we randomly selected annotated samples from the entire dataset.

The same trend is observed in experiments on ImageNet, as detailed in Table 5.4. Our method outperforms SelfMatch significantly, especially when labeled data is limited. This suggests that our method can more effectively leverage pre-trained models to enhance semi-supervised learning performance. Additionally, this improvement is consistent across different semi-supervised training methods, including FlexMatch and FixMatch-EMAN.

Table 5.4 – Comparison of accuracy on ImageNet. The best results are shown in red and the second best results are shown in blue.

Semi-Supervised Method	Self-Supervised Initialization	0.2% labels		1% labels	
		Top-1	Top-5	Top-1	Top-5
Fine-tune	BYOL	26.0	48.7	53.2	68.8
FlexMatch	None	3.9	9.6	19.6	37.9
SelfMatch (w FlexMatch)	BYOL	30.3	48.8	57.3	80.1
SelfMatch (w FixMatch-EMAN)	BYOL-EMAN	32.7	52.7	59.6	81.6
Ours (w FlexMatch)	BYOL	37.3	55.9	58.3	79.9
Ours (w FixMatch-EMAN)	BYOL-EMAN	40.4	60.7	60.4	81.5

5.4.3 Model Detailed Analysis

We further analyze our approach from six aspects: ablation studies for the whole framework, influence of switching point, comparisons of active learning strategies, ablation studies for our active sampling strategy, and the detailed analysis of pseudo-label propagation and clustering results on self-supervised features.

Ablation Experiment of Framework

Ablation experiments were performed in the CIFAR-10 with 10 labels. We maintained the same hyper-parameters, except for reducing the total training iterations to 112×2304, equivalent to 2304 epochs. Two components of our method are evaluated: active labeled set selection and PPL. The results are shown in Table 5.5. (1) When working with a limited number of labels, the active learning component yields more stable and accurate results in semi-supervised training. (2) Whether for randomly selected or actively chosen annotated samples, the use of PPL significantly improves the performance of semi-supervised learning.

Influence of switching point

To assess the impact of the PPL switch point, T, on semi-supervised training, we conducted four experiments on the CIFAR-10 dataset with 10 and 40 labeled samples,

Table 5.5 – Ablation Study on CIFAR-10. Accuracy is the average over 3 runs.

Ablation	Accuracy(avg. ± std.)
Random	58.17 ± 13.01
Random + PPL	68.01 ± 15.96
Active	76.65 ± 5.29
Active + PPL	84.43 ± 5.45

respectively. The experiments included: (1) Without PPL, (2) Employing PPL as in eq. 5.5 with T set to 5 epochs, (3) T set to 60 epochs, and (4) Direct PPL, i.e., T equal to the total number of training iterations. The results are depicted in fig. 5.7 and fig. 5.8.

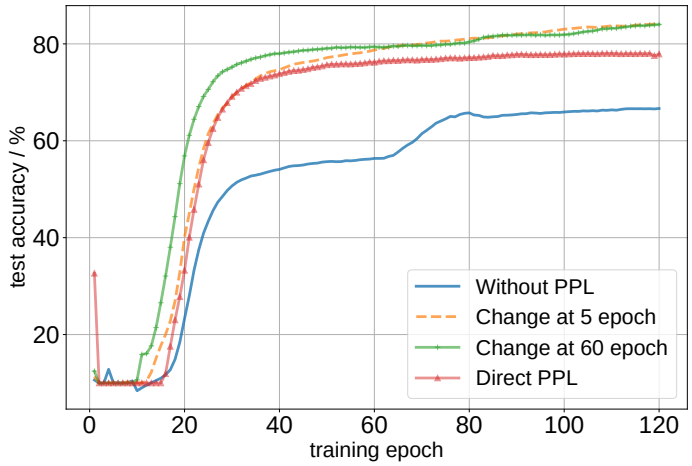


Figure 5.7 – Test accuracy of semi-supervised training in CIFAR-10 with 10 labels. In all experiments, the same self-supervised pre-training weight initialization is applied. Without PPL indicates the absence of PPL. Change at 5 epoch and change at 60 epoch refer to the utilization of PPL according to eq. 5.5, with T as 5 epoch and 60 epoch, respectively. Direct PPL signifies using PPL in all training iterations.

PPL enhances semi-supervised training accuracy in the early stages, but continuous use of PPL (experiment setting (4)) hinders the sustained update of pseudo-labels, thereby impacting the final semi-supervised performance. For instance, in fig. 5.8, the semi-supervised training accuracy of experiment setting (1) surpasses that with continuous PPL (setting (4)) after training epochs exceed about 70. Hence, a switching mechanism, as represented by eq. 5.5, is necessary for guiding the semi-supervised

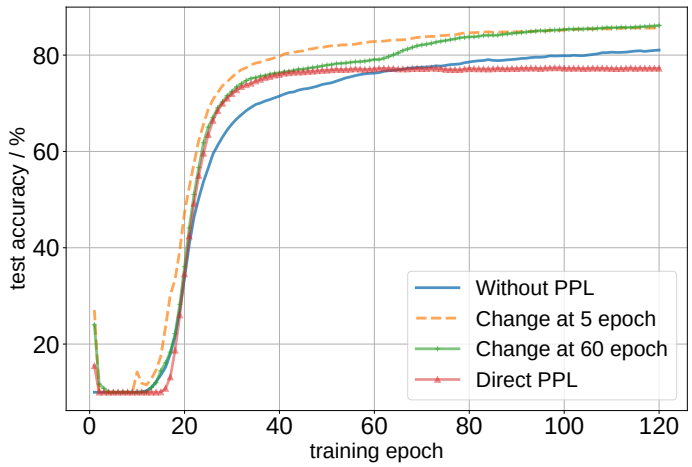


Figure 5.8 – Test accuracy of semi-supervised training in CIFAR-10 with 40 labels. In all experiments, the same self-supervised pre-training weight initialization is applied. Without PPL indicates the absence of PPL. Change at 5 epoch and change at 60 epoch refer to the utilization of PPL according to eq. 5.5, with T as 5 epoch and 60 epoch, respectively. Direct PPL signifies using PPL in all training iterations.

training process.

The specific placement of the switch point minimally affects the ultimate performance of semi-supervised training. Although setting a larger T (i.e., using PPL in more training iterations) yields higher accuracy in the early stages of semi-supervised training with limited labeled data as shown in fig. 5.7, and setting a smaller T helps avoid the constraint imposed by fixed PPL on the process of updating pseudo-labels in scenarios with more labeled data as shown in fig. 5.8, after the switch at point T, the semi-supervised training accuracy becomes consistent quickly for different T settings. Therefore, the impact of setting a roughly appropriate switch point on the final performance of semi-supervised training is minimal.

Ablation Study of Our Active Sampling Strategy

We conducted ablation experiments to compare the effects of various components in our proposed active learning strategy on CIFAR-10. In these experiments, 'K-medoids' refers to clustering only once, while 'multi-clustering' involves clustering 6 times, as described in Sec. 5.4.1.

As shown in Table 5.6, fine-tuning the features results in significant improvements, providing better class coverage and more accurate pseudo-labels. Especially for the case with fewer annotations, where multi-clustering and fine-tuning features yield greater benefits.

Table 5.6 – Ablation study of proposed active learning strategy on CIFAR-10. Accuracy reported is the average over 3 runs.

	Accuracy of y_{prior}		Class Coverage	
# Labels	10	40	10	40
K-medoids on f_{self}	44.90±1.88	69.62±0.83	7.2±1.4	10.0±0.0
Multi-cluster on f_{self}	61.94±5.42	72.54±1.42	9.0±0.7	10.0±0.0
K-medoids on f_{fine}	64.70±6.27	72.72±2.77	9.2±0.8	10.0±0.0
Multi-cluster on f_{fine}	71.41±4.10	74.91±1.66	9.5±0.5	10.0±0.0

We investigated the impact of setting different clustering times, C, on the samples selected by our proposed active learning strategy. The experiment is implemented on CIFAR-10 with 10 labels. As shown in fig. 5.9 and fig. 5.10, we note that our strategy is robust to the number of clustering C. Across different values of C, our active learning strategy consistently selects samples that comprehensively cover all classes, even in scenarios with very few annotations. The choice of C primarily influences the accuracy of the PPL. The larger C is, the smaller the variance, and the accuracy of pseudo-labels can be slightly improved.

Influence of Active Sampling Strategy

We compare different active learning strategies on CIFAR-10, CIFAR-100, and STL-10, with all hyper-parameters consistent with those mentioned in Sec. 5.4.1, except for the training iterations. In these experiments, all models are trained for 112×2304 iterations. Many active learning strategies involve multiple rounds of iteration for selecting annotated samples, often resulting in high training costs, contradicting the fundamental goal of active semi-supervised learning, which aims to reduce the overall cost. Therefore, we choose active learning methods that allow for the selection of annotated samples in a single iteration as our baseline. Specially, random, K-

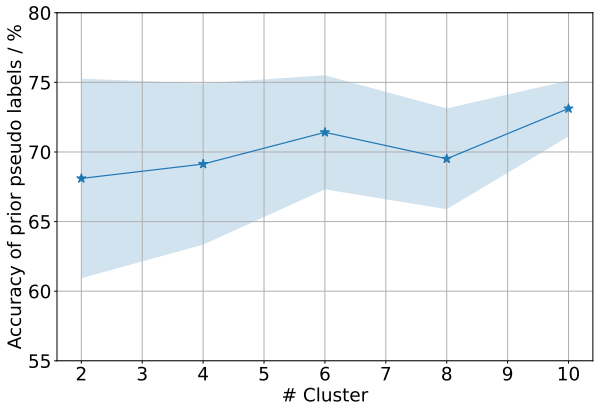


Figure 5.9 – The impact of the number of clusters C in an active learning strategy on the accuracy of the prior pseudo-label.

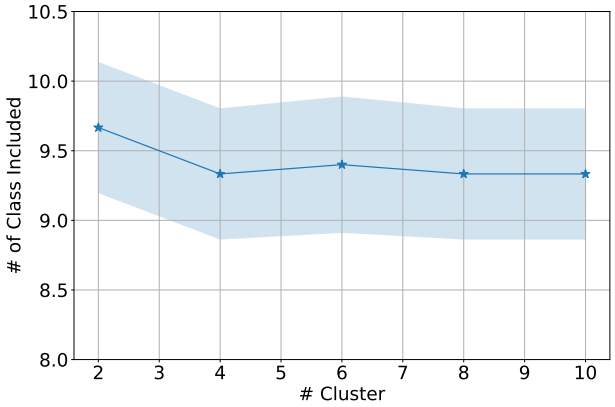


Figure 5.10 – The impact of the number of clusters C in an active learning strategy on the class coverage.

medoids (Sener and Savarese, 2018) and Coreset-greedy (Sener and Savarese, 2018) are selected. As shown in tab. 5.7, our method consistently outperforms other active learning strategies. K-medoids outperforms random sampling in most cases, whereas Coreset often performs worse. This discrepancy arises because Coreset selects samples that are least similar to the features of the existing labeled samples, while K-medoids selects samples closest to the cluster centers. Consequently, Coreset is more likely to select outliers, which decreases the performance of semi-supervised training. This observation aligns with previous work Mittal et al. (2019), which found that some

active learning strategies can have adverse effects when the number of labeled samples is limited.

Table 5.7 – Comparison of active sampling strategies. The accuracy reported is the average over 3 runs. The best results are shown in red and the second best results are shown in blue.

Dataset	CIFAR-10		CIFAR-100		STL-10	
Size of Labeled Set	10	40	200	400	20	40
Random	58.17±13.01	82.81±8.10	38.84±3.09	58.42±2.37	51.14±6.79	63.33±5.97
K-medoids	47.26±6.62	87.42±5.42	43.39±4.00	59.10±0.36	51.03±4.89	68.29±2.93
Coreset	31.92±3.14	86.19±10.78	27.59±4.63	47.77±3.22	45.75±3.95	51.41±3.74
Proposed	84.43±5.19	94.25±0.43	51.22±3.06	61.07±0.12	61.39±5.43	70.26±3.60

Prior Pseudo-label Propagation

For PPL generation, we compare our method with LLGC (Zhou et al., 2003), which is a typical baseline for label spreading, with the hyper-parameters of LLGC following (Iscen et al., 2019). Additionally, we investigate the impact of selecting labeled samples using different active learning methods on the accuracy of pseudo-labels, as presented in Table 5.8. The results confirm that the choice of labeled sample selection strategy has a large impact on the accuracy of pseudo-labels. Samples selected by our active learning strategy result in more accurate pseudo-labels across different label propagation methods. Our pseudo-label propagation method outperforms LLGC when the number of labeled samples is close to the number of true classes, but shows slightly weaker performance than LLGC when more labels are available.

Furthermore, we investigate the impact of the number of classes in clustering, K, in our label propagation. The experiments include three settings with different sizes of K, as summarized in Table 5.9. Expected calibration error (ECE) (Naeini et al., 2015) is employed to assess how well y_{prior} is calibrated to the true accuracy, where smaller values indicate less miscalibration. The results confirm that using different K in label propagation is a good compromise between accuracy and calibration.

Table 5.8 – Accuracy of prior pseudo-label comparison of different active strategies and label propagation methods. The accuracy reported is the average over 3 runs. The best results are shown in red and the second best results are shown in blue.

Dataset		CIFAR-10		
# Labels		10	20	40
Propagation	Sampling			
LLGC	Random	53.42±5.36	59.91±2.58	69.59±2.35
LLGC	Coreset	31.57±6.31	56.06±3.35	74.47±5.17
LLGC	K-medoids	45.53±2.96	62.28±5.39	71.51±1.36
LLGC	USL	53.90±0.56	63.47±3.19	70.82±1.76
LLGC	Proposed	62.94±3.47	71.61±1.64	75.50±1.40
Proposed	Random	52.12±9.10	60.32±4.67	69.40±2.49
Proposed	Coreset	37.43±3.24	59.56±3.30	73.82±3.01
Proposed	K-medoids	44.90±1.88	62.12±6.97	69.62±0.83
Proposed	USL	54.20±0.43	57.62±5.10	67.61±1.86
Proposed	Proposed	71.41±4.10	72.63±0.96	74.91±1.66

Table 5.9 – Ablation study of K on CIFAR-10, accuracy of y_{prior} and Expected calibration error (ECE) reported is the average over 3 runs.

	ECE			Accuracy of y_{prior}		
# Labels	10	20	40	10	20	40
K=[10,10,10,10,10,10]	0.278±0.039	0.166±0.038	0.112±0.021	70.50±4.13	70.66±4.05	70.81±2.13
K=[10,20,30,40,50,60]	0.237±0.041	0.087±0.014	0.067±0.013	71.41±4.10	72.63±0.96	74.91±1.66
K=[10,60,60,60,60,60]	0.233±0.040	0.133±0.037	0.082±0.011	70.53±4.09	70.46±2.82	73.67±1.67

Clustering Results on Self-Supervised Features

We analyzed clustering results in self-supervised feature space, as shown in Fig. 5.11 and Fig. 5.12. The self-supervised features were clustered five times, with the number of clusters equal to the actual number of classes in the datasets. The samples were divided into 10 bins based on the distance between the sample features and the cluster centers. The proportion of sample labels in each bin matching the dominant labels in the clusters was then calculated. The result shows that samples near the center of the cluster are more likely to share the same label as the dominant label in the cluster.

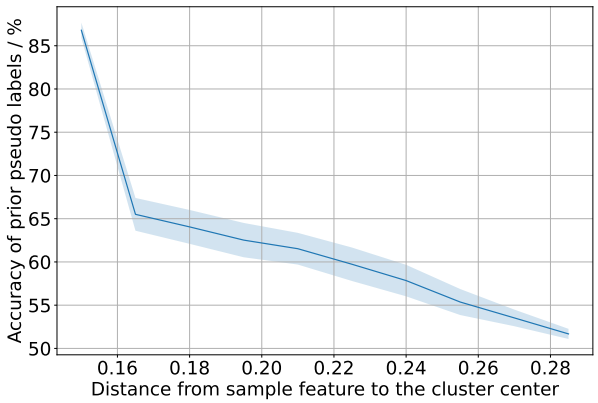


Figure 5.11 – The relationship between distance and dominant labels in self-supervised feature space on CIFAR-10, where the model is self-supervised training by Simsiam (Chen and He, 2021).

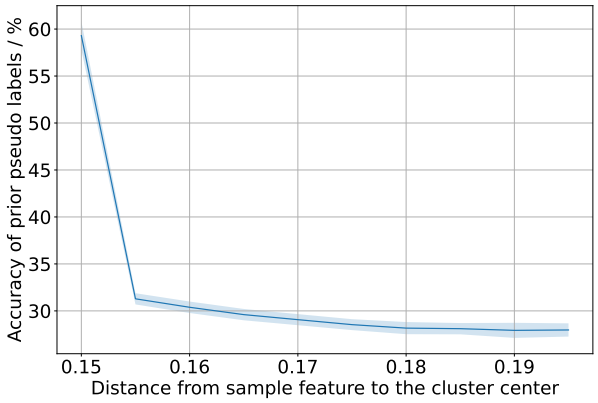


Figure 5.12 – The relationship between distance and dominant labels in self-supervised feature space on CIFAR-100, where the model is self-supervised training by Simsiam (Chen and He, 2021).

5.5 Limitations

Our method primarily relies on prior pseudo-labels generated through clustering on pre-trained features to enhance semi-supervised training performance. This approach faces limitations in scenarios where clustering cannot effectively distinguish between different classes, such as in multi-label learning where a single image contains multiple objects. Additionally, the fixed pseudo-label switching time T may hinder perfor-

mance in nonstationary environments, such as stream learning.

5.6 Summary

In this chapter, we observed that in scenarios with limited labeled samples, semi-supervised training often quickly compromises the well-established feature representations obtained through self-supervised training. Weight initialization, in such cases, fails to effectively transfer valuable information from self-supervised pre-training to the semi-supervised model, resulting in a reduction of the benefits gained from combining self-supervised and semi-supervised training. Motivated by this observation, we propose using PPL as an intermediate step to assist semi-supervised training in acquiring valuable information from self-supervised pre-training. Additionally, to more fully exploit self-supervised pre-training information, we introduce a novel active learning strategy to enhance the accuracy of PPL. By combining weight initialization, PPL, and the active learning strategy, our framework provides a more effective starting point for semi-supervised training. Experimental validation demonstrates the effectiveness of our approach in enhancing the performance of semi-supervised learning under limited labeled data conditions. Furthermore, our method readily integrates with existing semi-supervised learning approaches that include pseudo-labeling techniques, showcasing its broad applicability.

Chapter 6

Conclusion

6.1 Summary

The abundance of data is crucial for the success of deep learning models (Sorscher et al., 2022; Tay et al., 2021), but acquiring a large amount of data, especially annotated data, can be costly and slow. This impedes the application of deep learning models in many fields (Alonso et al., 2019; Budd et al., 2021; Norouzzadeh et al., 2021). Recently, the development of self-supervised pre-training techniques provided a new cornerstone for reducing the need for human annotations (Feichtenhofer et al., 2022; Oquab et al., 2024; Radford et al., 2021). This development transforms the common practice of training deep learning models from scratch to pre-training with unlabeled data, followed by fine-tuning with a small amount of labeled data.

In this thesis, we recognize the unique and complementary contributions of pre-training methods, active learning, and semi-supervised learning to the enhancement of label efficiency. Building upon this recognition, we proposed novel strategies for active learning and semi-supervised learning, tailored for scenario training on top of pre-trained models. The goal is to alleviate the dependency on human annotations and subsequently reduce overall costs.

In **Chapter 3**, we focus on the effectiveness of active learning strategies when training

on top of pre-trained models. The utilization of pre-trained models significantly reduces the demands of labeled data, pushing active learning into a low-budget regime. However, within this range, the types of samples that can enhance model performance differ significantly from those selected by most existing high-budget active learning strategies. Also, we observe an intensified phase transition phenomenon, indicating that the effective range of existing low-budget active learning strategies is too narrow for practical use. We argue that this issue is attributed to the upper bound of the empirical risk in the active learning pool. To address this, we propose using NTK and CPL to directly estimate the empirical risk. Experimental results show that our proposed active learning strategy, NTKCPL, outperforms existing methods and has a wider effective budget range. The introduction of this strategy enables practitioners to benefit from the combination of pre-training and active learning.

In **Chapter 4**, we delve into the efficiency and overall costs (labeling cost and training cost) of active learning when training on top of pre-trained models. Since pre-trained models are often large-scale models, the training costs are significantly higher than those of small models typically used in traditional active learning. Active learning requires multiple training iterations to select labeled samples, indicating that computational time and costs cannot be overlooked when employing active learning on top of pre-trained models. We observe that existing efficient active learning methods reduce active learning time but increase overall costs (primarily from the additional training costs). This dilemma forces practitioners to navigate between computational efficiency and overall costs. To address this challenge, we propose a novel efficient active learning framework that enhances the accuracy of efficient active learning by updating pre-trained features and adopting appropriate training methods. This framework alleviates the dilemma faced by practitioners, significantly reducing the time required for standard active learning without increasing overall costs.

In the aforementioned two chapters, we investigated active learning on top of pre-trained models, where unlabeled data is utilized in the pre-training phase with a pretext task. Subsequently, in the active learning phase, only labeled data is employed to train the target task head and fine-tune the feature extractor. When the pre-trained

features fall short of meeting the requirements of the target task, more adjustments to the feature extractor are often necessary.

However, a small amount of labeled data is insufficient for effective tuning of the feature extractor, which may lead to a decrease in model generalization ability. Therefore, we attempt to introduce semi-supervised learning to fine-tune the feature extractor using unlabeled data in **Chapter 5**.

The challenge lies in the low accuracy of pseudo-labels during the initial stage of semi-supervised learning, particularly in scenarios with limited annotations. Consequently, there is a difficulty in effectively transferring valuable pre-training information through weight initialization. The combination of self-supervised and semi-supervised learning fails to yield performance gains. To address this, we propose using prior pseudo-labels as intermediaries to transfer information. Initially, we cluster on pre-trained features to generate prior pseudo-labels, which are then used to guide the initial stage of semi-supervised learning. Additionally, considering the impact of labeled data selection on the accuracy of prior pseudo-labels, we introduce a new active learning strategy to improve the accuracy of prior pseudo-labels. Ultimately, through a combination of weight initialization, prior pseudo-labels, and actively labeled data, we significantly improve the model's performance in scenarios with very few annotations.

In this thesis, we introduce novel active learning strategies, an efficient active learning framework, and active semi-supervised learning methods. These contributions empower practitioners to derive benefits from pre-trained models and various label-efficient learning methods, thereby further reducing the required amount of labeled data and overall costs for model training. We believe that these advancements can extend the benefits of deep learning to more domains sensitive to annotation costs. We anticipate that future research in the following directions will contribute to enhancing the practical applicability of label-efficient training methods.

6.2 Future Work

6.2.1 Cost-aware Learning

With the advancement of pre-training methods, various label-efficient learning approaches (Cai et al., 2021; Emam et al., 2021; Wang et al., 2022b) have increasingly integrated pre-trained models. While this integration improves label efficiency, it significantly raises computational costs. Recent studies have shown that larger models are particularly effective at leveraging unlabeled data (Chen et al., 2020b; He et al., 2022), further emphasizing the computational burden. As a result, the earlier assumption that savings in annotation costs would substantially outweigh the increased computational costs in label-efficient learning is no longer reasonable.

To address this, future research should prioritize cost-aware learning, focusing on developing methods that train effective models while minimizing total costs. This entails not only designing cost-efficient algorithms but also establishing suitable metrics to evaluate overall cost savings comprehensively.

Additionally, exploring tools to approximate the effects of incorporating additional labeled data or extending training is a promising direction. Methods that can predict how neural network outputs change after such updates—particularly in practical settings like pre-trained model fine-tuning or semi-supervised learning—can be highly valuable. For instance, reliable estimates of these changes could guide practitioners in deciding whether to continue semi-supervised training or allocate resources to active learning for additional annotations. Advancements in this area will significantly contribute to the development of cost-aware learning strategies.

6.2.2 Trustworthy Label-Efficient Learning for Scientific Research

Label-efficient learning serves as a critical tool for deploying deep learning models in domains where annotation is expensive, a common occurrence in scientific research

fields where data labeling often requires specialized domain knowledge (Alonso et al., 2019; Tuia et al., 2022). However, when extending label-efficient learning methods to scientific research fields, it is essential not only to consider improving label efficiency but also to ensure that these methods do not introduce biases affecting the reliability of model predictions (Li et al., 2023a). Currently, the majority of label-efficient learning methods rely on the exploitation of deep learning models, such as active learning leveraging model predictions to identify informative samples and semi-supervised learning using noisy model predictions as pseudo-labels for training. These approaches contribute to the accumulation of biases (Chen et al., 2022; Farquhar et al., 2020), diminishing the trustworthiness of the final model.

Therefore, conducting a comprehensive assessment of the impact of label-efficient learning methods on the trustworthiness of model predictions, and designing novel approaches that improve label efficiency while ensuring model trustworthiness, merits further research.

6.2.3 Multi-Modal Pre-training for Label-Efficient Learning

Recently, multi-modal pre-training (Radford et al., 2021) has attracted extensive attention. In contrast to models pre-trained on images, multi-modal pre-training, particularly for visual-language models, naturally embeds images with similar semantics into proximity within the feature space, significantly improving the performance of deep learning models across tasks such as zero-shot learning (Zheng et al., 2021), few-shot learning (Zhang et al., 2022a), and open-set learning (Zhu et al., 2023). Of course, challenges persist, including semantic misalignment between the pre-trained model's semantics and those of the target tasks and inconsistencies in data distribution. Therefore, fine-tuning (Wei et al., 2023) or prompting (Lu et al., 2022; Zhou et al., 2022) the multi-modal pre-training models using labeled data is still necessary. However, the differences between multi-modal pre-training models and visual pre-training models introduce new challenges for label-efficient learning. These include identifying and correcting inaccurate semantics correspondences within the pre-

trained models (Wang et al., 2024), determining the form of additional annotations, and efficiently embedding existing knowledge in a label-efficient and training-efficient manner.

6.2.4 Life-long Learning

The label-efficient learning methods discussed in this thesis assume a scenario where no models trained for similar tasks are already available. Therefore, practitioners have to train models from scratch, involving the selection of new labeled data in active learning and fine-tuning a pre-trained model (on a different task) using task-relevant labels in semi-supervised learning. This stands in contrast to the current reality, where the prolific development of deep learning has resulted in numerous models trained for various tasks. With the increasing availability of models trained from similar tasks, the challenge lies in efficiently utilizing a small number of new labels to select the most suitable existing models. Moreover, exploring the adaptation of these models to new target tasks (Mundt et al., 2023) and novel datasets is worth further investigation.

List of References

Ibrahim M Alabdulmohsin, Behnam Neyshabur, and Xiaohua Zhai. Revisiting neural scaling laws in language and vision. *Advances in Neural Information Processing Systems*, 35:22300–22312, 2022.

Inigo Alonso, Matan Yuval, Gal Eyal, Tali Treibitz, and Ana C Murillo. Coralseg: Learning coral segmentation from sparse annotations. *Journal of Field Robotics*, 36(8):1456–1477, 2019.

Eric Arazo, Diego Ortego, Paul Albert, Noel E O'Connor, and Kevin McGuinness. Pseudo-labeling and confirmation bias in deep semi-supervised learning. In *2020 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2020.

Sanjeev Arora, Simon S Du, Wei Hu, Zhiyuan Li, Russ R Salakhutdinov, and Ruosong Wang. On exact computation with an infinitely wide neural net. *Advances in neural information processing systems*, 32, 2019.

Jordan T. Ash, Chicheng Zhang, Akshay Krishnamurthy, John Langford, and Alekh Agarwal. Deep batch active learning by diverse, uncertain gradient lower bounds. In *International Conference on Learning Representations*, 2020.

Mahmoud Assran, Mathilde Caron, Ishan Misra, Piotr Bojanowski, Florian Bordes, Pascal Vincent, Armand Joulin, Mike Rabbat, and Nicolas Ballas. Masked siamese networks for label-efficient learning. In *European Conference on Computer Vision*, pages 456–473. Springer, 2022.

Haoping Bai, Meng Cao, Ping Huang, and Jiulong Shan. Self-supervised semi-supervised learning for data labeling and quality evaluation. *arXiv preprint arXiv:2111.10932*, 2021.

Adrien Bardes, Jean Ponce, and Yann Lecun. Vicreg: Variance-invariance-covariance regularization for self-supervised learning. In *ICLR 2022-International Conference on Learning Representations*, 2022a.

Adrien Bardes, Jean Ponce, and Yann LeCun. Vicregl: Self-supervised learning of local visual features. *Advances in Neural Information Processing Systems*, 35: 8799–8810, 2022b.

Horace B Barlow et al. Possible principles underlying the transformation of sensory messages. *Sensory communication*, 1(01):217–233, 1961.

Sugato Basu, Arindam Banerjee, and Raymond Mooney. Semi-supervised clustering by seeding. In *In Proceedings of 19th International Conference on Machine Learning (ICML-2002*. Citeseer, 2002.

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

David Berthelot, Nicholas Carlini, Ian Goodfellow, Nicolas Papernot, Avital Oliver, and Colin A Raffel. Mixmatch: A holistic approach to semi-supervised learning. *Advances in Neural Information Processing Systems*, 32, 2019.

David Berthelot, Nicholas Carlini, Ekin D. Cubuk, Alex Kurakin, Kihyuk Sohn, Han Zhang, and Colin Raffel. Remixmatch: Semi-supervised learning with distribution matching and augmentation anchoring. In *International Conference on Learning Representations*, 2020.

Michael Bewley, Ariell Friedman, Renata Ferrari, Nicole Hill, Renae Hovey, Neville Barrett, Ezequiel M Marzinelli, Oscar Pizarro, Will Figueira, Lisa Meyer, et al. Australian sea-floor survey data, with images and expert annotations. *Scientific data*, 2(1):1–13, 2015.

Zalán Borsos, Mojmir Mutny, and Andreas Krause. Coresets via bilevel optimization for continual learning and streaming. *Advances in neural information processing systems*, 33:14879–14890, 2020.

Zalán Borsos, Marco Tagliasacchi, and Andreas Krause. Semi-supervised batch active learning via bilevel optimization. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3495–3499. IEEE, 2021.

Samuel Budd, Emma C Robinson, and Bernhard Kainz. A survey on active learning and human-in-the-loop deep learning for medical image analysis. *Medical Image Analysis*, 71:102062, 2021.

Zhaowei Cai, Avinash Ravichandran, Subhransu Maji, Charless Fowlkes, Zhuowen Tu, and Stefano Soatto. Exponential moving average normalization for self-supervised and semi-supervised learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 194–203, 2021.

Razvan Caramalau, Binod Bhattarai, and Tae-Kyun Kim. Sequential graph convolutional network for active learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9583–9592, 2021.

Mathilde Caron, Piotr Bojanowski, Armand Joulin, and Matthijs Douze. Deep clustering for unsupervised learning of visual features. In *Proceedings of the European conference on computer vision (ECCV)*, pages 132–149, 2018.

Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. *Advances in neural information processing systems*, 33:9912–9924, 2020.

Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9650–9660, 2021.

Paola Cascante-Bonilla, Fuwen Tan, Yanjun Qi, and Vicente Ordonez. Curriculum labeling: Revisiting pseudo-labeling for semi-supervised learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pages 6912–6920, 2021.

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

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

Baixu Chen, Junguang Jiang, Ximei Wang, Pengfei Wan, Jianmin Wang, and Mingsheng Long. Debiased self-training for semi-supervised learning. *Advances in Neural Information Processing Systems*, 35:32424–32437, 2022.

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

Ting Chen, Simon Kornblith, Kevin Swersky, Mohammad Norouzi, and Geoffrey E Hinton. Big self-supervised models are strong semi-supervised learners. *Advances in neural information processing systems*, 33:22243–22255, 2020b.

Xinlei Chen and Kaiming He. Exploring simple siamese representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15750–15758, 2021.

Xinlei Chen, Haoqi Fan, Ross Girshick, and Kaiming He. Improved baselines with momentum contrastive learning. *arXiv preprint arXiv:2003.04297*, 2020c.

Xinlei Chen, Zhuang Liu, Saining Xie, and Kaiming He. Deconstructing denoising diffusion models for self-supervised learning. *arXiv preprint arXiv:2401.14404*, 2024.

Gui Citovsky, Giulia DeSalvo, Claudio Gentile, Lazaros Karydas, Anand Rajagopalan, Afshin Rostamizadeh, and Sanjiv Kumar. Batch active learning at scale. *Advances in Neural Information Processing Systems*, 34:11933–11944, 2021.

Adam Coates, Andrew Ng, and Honglak Lee. An analysis of single-layer networks in unsupervised feature learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 215–223. JMLR Workshop and Conference Proceedings, 2011.

Cody Coleman, Christopher Yeh, Stephen Mussmann, Baharan Mirzasoleiman, Peter Bailis, Percy Liang, Jure Leskovec, and Matei Zaharia. Selection via proxy: Efficient data selection for deep learning. In *International Conference on Learning Representations*, 2019.

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

Qinyi Deng, Yong Guo, Zhibang Yang, Haolin Pan, and Jian Chen. Boosting semi-supervised learning with contrastive complementary labeling. *Neural Networks*, 170:417–426, 2024.

Carl Doersch, Abhinav Gupta, and Alexei A Efros. Unsupervised visual representation learning by context prediction. In *Proceedings of the IEEE international conference on computer vision*, pages 1422–1430, 2015.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2020.

Zeyad Ali Sami Emam, Hong-Min Chu, Ping-Yeh Chiang, Wojciech Czaja, Richard Leapman, Micah Goldblum, and Tom Goldstein. Active learning at the imagenet scale. *arXiv preprint arXiv:2111.12880*, 2021.

Sebastian Farquhar, Yarin Gal, and Tom Rainforth. On statistical bias in active learning: How and when to fix it. In *International Conference on Learning Representations*, 2020.

Christoph Feichtenhofer, Yanghao Li, Kaiming He, et al. Masked autoencoders as spatiotemporal learners. *Advances in neural information processing systems*, 35: 35946–35958, 2022.

Zhanzhou Feng and Shiliang Zhang. Evolved part masking for self-supervised learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10386–10395, 2023.

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

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

Spyros Gidaris, Praveer Singh, and Nikos Komodakis. Unsupervised representation learning by predicting image rotations. In *ICLR 2018*, 2018.

Chengyue Gong, Dilin Wang, and Qiang Liu. Alphamatch: Improving consistency for semi-supervised learning with alpha-divergence. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 13683–13692, 2021.

Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *Communications of the ACM*, 63(11):139–144, 2020.

Yves Grandvalet and Yoshua Bengio. Semi-supervised learning by entropy minimization. *Advances in neural information processing systems*, 17, 2004.

Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Guo, Mohammad Gheshlaghi Azar, et al. Bootstrap your own latent-a new approach to self-supervised learning. *Advances in Neural Information Processing Systems*, 33:21271–21284, 2020.

Guan Gui, Zhen Zhao, Lei Qi, Luping Zhou, Lei Wang, and Yinghuan Shi. Improving barely supervised learning by discriminating unlabeled samples with super-class. *Advances in Neural Information Processing Systems*, 35:19849–19860, 2022.

Jiannan Guo, Haochen Shi, Yangyang Kang, Kun Kuang, Siliang Tang, Zhuoren Jiang, Changlong Sun, Fei Wu, and Yueting Zhuang. Semi-supervised active learning for semi-supervised models: exploit adversarial examples with graph-based virtual labels. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2896–2905, 2021.

Guy Hacohen and Daphna Weinshall. How to select which active learning strategy is best suited for your specific problem and budget. *Advances in Neural Information Processing Systems*, 36, 2024.

Guy Hacohen, Avihu Dekel, and Daphna Weinshall. Active learning on a budget: Opposite strategies suit high and low budgets. In *International Conference on Machine Learning*, pages 8175–8195. PMLR, 2022.

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

Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9729–9738, 2020.

Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 16000–16009, 2022.

Joel Hestness, Sharan Narang, Newsha Ardalani, Gregory Diamos, Heewoo Jun, Hassan Kianinejad, Md Mostofa Ali Patwary, Yang Yang, and Yanqi Zhou. Deep learning scaling is predictable, empirically. *arXiv preprint arXiv:1712.00409*, 2017.

Ahmet Iscen, Giorgos Tolias, Yannis Avrithis, and Ondrej Chum. Label propagation for deep semi-supervised learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5070–5079, 2019.

Arthur Jacot, Franck Gabriel, and Clément Hongler. Neural tangent kernel: Convergence and generalization in neural networks. *Advances in neural information processing systems*, 31, 2018.

Jaeho Kang, Kwang Ryel Ryu, and Hyuk-Chul Kwon. Using cluster-based sampling to select initial training set for active learning in text classification. In *Advances in Knowledge Discovery and Data Mining: 8th Pacific-Asia Conference, PAKDD 2004, Sydney, Australia, May 26-28, 2004. Proceedings 8*, pages 384–388. Springer, 2004.

Siddharth Karamcheti, Ranjay Krishna, Li Fei-Fei, and Christopher D Manning. Mind your outliers! investigating the negative impact of outliers on active learning for visual question answering. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 7265–7281, 2021.

Byoungjip Kim, Jinho Choo, Yeong-Dae Kwon, Seongho Joe, Seungjai Min, and Youngjune Gwon. Selfmatch: Combining contrastive self-supervision and consistency for semi-supervised learning. *arXiv preprint arXiv:2101.06480*, 2021a.

Jiwon Kim, Youngjo Min, Daehwan Kim, Gyuseong Lee, Junyoung Seo, Kwangrok Ryoo, and Seungryong Kim. Conmatch: Semi-supervised learning with confidence-guided consistency regularization. In *European Conference on Computer Vision*, pages 674–690. Springer, 2022.

Noo-Ri Kim and Jee-Hyong Lee. Propagation regularizer for semi-supervised learning with extremely scarce labeled samples. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14401–14410, 2022.

Yoon-Yeong Kim, Kyungwoo Song, JoonHo Jang, and Il-Chul Moon. Lada: Look-ahead data acquisition via augmentation for deep active learning. *Advances in Neural Information Processing Systems*, 34:22919–22930, 2021b.

Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

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

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. Technical Report TR-2009, University of Toronto, 2009.

Ananya Kumar, Aditi Raghunathan, Robbie Jones, Tengyu Ma, and Percy Liang. Fine-tuning can distort pretrained features and underperform out-of-distribution. In *International Conference on Learning Representations*, 2022.

Samuli Laine and Timo Aila. Temporal ensembling for semi-supervised learning. In *International Conference on Learning Representations*, 2016.

Gustav Larsson, Michael Maire, and Gregory Shakhnarovich. Learning representations for automatic colorization. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part IV 14*, pages 577–593. Springer, 2016.

Dong-Hyun Lee et al. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In *Workshop on challenges in representation learning, ICML*, volume 3, page 896. Atlanta, 2013.

Jaehoon Lee, Lechao Xiao, Samuel Schoenholz, Yasaman Bahri, Roman Novak, Jascha Sohl-Dickstein, and Jeffrey Pennington. Wide neural networks of any

depth evolve as linear models under gradient descent. *Advances in neural information processing systems*, 32, 2019.

David D Lewis and Jason Catlett. Heterogeneous uncertainty sampling for supervised learning. In *Machine learning proceedings 1994*, pages 148–156. Elsevier, 1994.

Bo Li, Peng Qi, Bo Liu, Shuai Di, Jingen Liu, Jiquan Pei, Jinfeng Yi, and Bowen Zhou. Trustworthy ai: From principles to practices. *ACM Computing Surveys*, 55 (9):1–46, 2023a.

Junnan Li, Caiming Xiong, and Steven CH Hoi. Comatch: Semi-supervised learning with contrastive graph regularization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9475–9484, 2021.

Xiaotong Li, Zixuan Hu, Yixiao Ge, Ying Shan, and Ling-Yu Duan. Exploring model transferability through the lens of potential energy. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5429–5438, 2023b.

Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10012–10022, 2021.

Yuning Lu, Jianzhuang Liu, Yonggang Zhang, Yajing Liu, and Xinmei Tian. Prompt distribution learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5206–5215, 2022.

Thomas Lucas, Philippe Weinzaepfel, and Gregory Rogez. Barely-supervised learning: Semi-supervised learning with very few labeled images. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 1881–1889, 2022.

Carsten Lüth, Till Bungert, Lukas Klein, and Paul Jaeger. Navigating the pitfalls of active learning evaluation: A systematic framework for meaningful performance assessment. *Advances in Neural Information Processing Systems*, 36, 2023.

Rafid Mahmood, Sanja Fidler, and Marc T Law. Low-budget active learning via wasserstein distance: An integer programming approach. In *International Conference on Learning Representations*, 2022.

Ambuj Mehrish, Navonil Majumder, Rishabh Bharadwaj, Rada Mihalcea, and Soujanya Poria. A review of deep learning techniques for speech processing. *Information Fusion*, page 101869, 2023.

Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014.

Ishan Misra and Laurens van der Maaten. Self-supervised learning of pretext-invariant representations. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 6707–6717, 2020.

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

Takeru Miyato, Shin-ichi Maeda, Masanori Koyama, and Shin Ishii. Virtual adversarial training: a regularization method for supervised and semi-supervised learning. *IEEE transactions on pattern analysis and machine intelligence*, 41(8): 1979–1993, 2018.

Mohamad Amin Mohamadi, Wonho Bae, and Danica J Sutherland. Making look-ahead active learning strategies feasible with neural tangent kernels. In *Advances in Neural Information Processing Systems*, 2022.

Mohamad Amin Mohamadi, Wonho Bae, and Danica J Sutherland. A fast, well-founded approximation to the empirical neural tangent kernel. In *International Conference on Machine Learning*, pages 25061–25081. PMLR, 2023.

Ali Mottaghi and Serena Yeung. Adversarial representation active learning. *arXiv preprint arXiv:1912.09720*, 2019.

Martin Mundt, Yongwon Hong, Iuliia Pliushch, and Visvanathan Ramesh. A wholistic view of continual learning with deep neural networks: Forgotten lessons and the bridge to active and open world learning. *Neural Networks*, 160:306–336, 2023.

Prateek Munjal, Nasir Hayat, Munawar Hayat, Jamshid Sourati, and Shadab Khan. Towards robust and reproducible active learning using neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 223–232, 2022.

Mahdi Pakdaman Naeini, Gregory Cooper, and Milos Hauskrecht. Obtaining well calibrated probabilities using bayesian binning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 29, 2015.

Islam Nassar, Munawar Hayat, Ehsan Abbasnejad, Hamid Rezatofighi, and Gholamreza Haffari. Protocon: Pseudo-label refinement via online clustering and prototypical consistency for efficient semi-supervised learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11641–11650, 2023.

Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y. Ng. Reading digits in natural images with unsupervised feature learning. In *NIPS Workshop on Deep Learning and Unsupervised Feature Learning 2011*, 2011.

Mehdi Noroozi and Paolo Favaro. Unsupervised learning of visual representations by solving jigsaw puzzles. In *European conference on computer vision*, pages 69–84. Springer, 2016.

Mohammad Sadegh Norouzzadeh, Dan Morris, Sara Beery, Neel Joshi, Nebojsa Jojic, and Jeff Clune. A deep active learning system for species identification and counting in camera trap images. *Methods in ecology and evolution*, 12(1):150–161, 2021.

Augustus Odena, Christopher Olah, and Jonathon Shlens. Conditional image synthesis with auxiliary classifier gans. In *International conference on machine learning*, pages 2642–2651. PMLR, 2017.

Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy V. Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel HAZIZA, Francisco Massa, Alaaeldin El-Nouby, Mido Assran, Nicolas Ballas, Wojciech Galuba, Russell Howes, Po-Yao Huang, Shang-Wen Li, Ishan Misra, Michael Rabbat, Vasu Sharma, Gabriel Synnaeve, Hu Xu, Herve Jegou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. DINOv2: Learning robust visual features without supervision. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856.

Omkar M Parkhi, Andrea Vedaldi, Andrew Zisserman, and CV Jawahar. Cats and dogs. In *2012 IEEE conference on computer vision and pattern recognition*, pages 3498–3505. IEEE, 2012.

Deepak Pathak, Philipp Krahenbuhl, Jeff Donahue, Trevor Darrell, and Alexei A Efros. Context encoders: Feature learning by inpainting. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2536–2544, 2016.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021.

Pengzhen Ren, Yun Xiao, Xiaojun Chang, Po-Yao Huang, Zhihui Li, Brij B Gupta, Xiaojiang Chen, and Xin Wang. A survey of deep active learning. *ACM computing surveys (CSUR)*, 54(9):1–40, 2021.

Yi Ren, Shangmin Guo, Wonho Bae, and Danica J Sutherland. How to prepare your task head for finetuning. In *The Eleventh International Conference on Learning Representations*, 2022.

Mamshad Nayeem Rizve, Kevin Duarte, Yogesh S Rawat, and Mubarak Shah. In defense of pseudo-labeling: An uncertainty-aware pseudo-label selection framework for semi-supervised learning. In *International Conference on Learning Representations*, 2021.

Nicholas Roy and Andrew McCallum. Toward optimal active learning through monte carlo estimation of error reduction. *ICML, Williamstown*, 2:441–448, 2001.

Tobias Scheffer, Christian Decomain, and Stefan Wrobel. Active hidden markov models for information extraction. In *International symposium on intelligent data analysis*, pages 309–318. Springer, 2001.

Mariia Seleznova and Gitta Kutyniok. Neural tangent kernel beyond the infinite-width limit: Effects of depth and initialization. In *International Conference on Machine Learning*, pages 19522–19560. PMLR, 2022.

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

Burr Settles. Active learning literature survey. *Technical Report TR-1648*, 2009.

Rasha Sheikh, Andres Milioto, Philipp Lottes, Cyrill Stachniss, Maren Bennewitz, and Thomas Schultz. Gradient and log-based active learning for semantic segmentation of crop and weed for agricultural robots. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1350–1356. IEEE, 2020.

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

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

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

Kihyuk Sohn, David Berthelot, Nicholas Carlini, Zizhao Zhang, Han Zhang, Colin A Raffel, Ekin Dogus Cubuk, Alexey Kurakin, and Chun-Liang Li. Fixmatch: Simplifying semi-supervised learning with consistency and confidence. *Advances in Neural Information Processing Systems*, 33:596–608, 2020.

Ben Sorscher, Robert Geirhos, Shashank Shekhar, Surya Ganguli, and Ari Morcos. Beyond neural scaling laws: beating power law scaling via data pruning. *Advances in Neural Information Processing Systems*, 35:19523–19536, 2022.

Swabha Swayamdipta, Roy Schwartz, Nicholas Lourie, Yizhong Wang, Hannaneh Hajishirzi, Noah A Smith, and Yejin Choi. Dataset cartography: Mapping and diagnosing datasets with training dynamics. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9275–9293, 2020.

Hui Tang and Kui Jia. Towards discovering the effectiveness of moderately confident samples for semi-supervised learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14658–14667, 2022.

Hui Tang, Lin Sun, and Kui Jia. Stochastic consensus: Enhancing semi-supervised learning with consistency of stochastic classifiers. In *European Conference on Computer Vision*, pages 330–346. Springer, 2022.

Antti Tarvainen and Harri Valpola. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. *Advances in neural information processing systems*, 30, 2017.

Yi Tay, Mostafa Dehghani, Jinfeng Rao, William Fedus, Samira Abnar, Hyung Won Chung, Sharan Narang, Dani Yogatama, Ashish Vaswani, and Donald Metzler. Scale efficiently: Insights from pretraining and finetuning transformers. In *International Conference on Learning Representations*, 2021.

Toan Tran, Thanh-Toan Do, Ian Reid, and Gustavo Carneiro. Bayesian generative active deep learning. In *International Conference on Machine Learning*, pages 6295–6304. PMLR, 2019.

Devis Tuia, Benjamin Kellenberger, Sara Beery, Blair R Costelloe, Silvia Zuffi, Benjamin Risse, Alexander Mathis, Mackenzie W Mathis, Frank van Langevelde, Tilo Burghardt, et al. Perspectives in machine learning for wildlife conservation. *Nature communications*, 13(1):792, 2022.

Wouter Van Gansbeke, Simon Vandenhende, Stamatios Georgoulis, Marc Proesmans, and Luc Van Gool. Scan: Learning to classify images without labels. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part X*, pages 268–285. Springer, 2020.

Vikas Verma, Kenji Kawaguchi, Alex Lamb, Juho Kannala, Arno Solin, Yoshua Bengio, and David Lopez-Paz. Interpolation consistency training for semi-supervised learning. *Neural Networks*, 145:90–106, 2022.

Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th international conference on Machine learning*, pages 1096–1103, 2008.

Kiri Wagstaff, Claire Cardie, Seth Rogers, Stefan Schrödl, et al. Constrained k-means clustering with background knowledge. In *Icml*, volume 1, pages 577–584, 2001.

Dan Wang and Yi Shang. A new active labeling method for deep learning. In *2014 International joint conference on neural networks (IJCNN)*, pages 112–119. IEEE, 2014.

Shuoyuan Wang, Jindong Wang, Guoqing Wang, Bob Zhang, Kaiyang Zhou, and Hongxin Wei. Open-vocabulary calibration for vision-language models. *arXiv preprint arXiv:2402.04655*, 2024.

Xudong Wang, Long Lian, and Stella X Yu. Unsupervised selective labeling for more effective semi-supervised learning. In *European Conference on Computer Vision*, pages 427–445. Springer, 2022a.

Xudong Wang, Zhirong Wu, Long Lian, and Stella X Yu. Debiased learning from naturally imbalanced pseudo-labels. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14647–14657, 2022b.

Yixuan Wei, Han Hu, Zhenda Xie, Ze Liu, Zheng Zhang, Yue Cao, Jianmin Bao, Dong Chen, and Baining Guo. Improving clip fine-tuning performance. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5439–5449, 2023.

Ziting Wen, Oscar Pizarro, and Stefan Williams. Ntkcpl: Active learning on top of self-supervised model by estimating true coverage. *arXiv preprint arXiv:2306.04099*, 2023.

Ziting Wen, Oscar Pizarro, and Stefan B. Williams. Feature alignment: Rethinking efficient active learning via proxy in the context of pre-trained models. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856.

Ziting Wen, Oscar Pizarro, and Stefan Williams. Active self-semi-supervised learning for few labeled samples. *Neurocomputing*, 614:128772, 2025.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 conference on empirical methods in natural language processing: system demonstrations*, pages 38–45, 2020.

Zhirong Wu, Yuanjun Xiong, Stella X Yu, and Dahua Lin. Unsupervised feature learning via non-parametric instance discrimination. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3733–3742, 2018.

Qizhe Xie, Zihang Dai, Eduard Hovy, Thang Luong, and Quoc Le. Unsupervised data augmentation for consistency training. *Advances in Neural Information Processing Systems*, 33:6256–6268, 2020.

Yichen Xie, Han Lu, Junchi Yan, Xiaokang Yang, Masayoshi Tomizuka, and Wei Zhan. Active finetuning: Exploiting annotation budget in the pretraining-finetuning paradigm. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 23715–23724, 2023.

Zhenda Xie, Zheng Zhang, Yue Cao, Yutong Lin, Jianmin Bao, Zhuliang Yao, Qi Dai, and Han Hu. Simmim: A simple framework for masked image modeling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9653–9663, 2022.

Hai-Ming Xu, Lingqiao Liu, Hao Chen, Ehsan Abbasnejad, and Rafael Felix. Progressive feature adjustment for semi-supervised learning from pretrained models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3292–3302, 2023.

Lei Yang, Qingqiu Huang, Huaiyi Huang, Linning Xu, and Dahua Lin. Learn to propagate reliably on noisy affinity graphs. In *European Conference on Computer Vision*, pages 447–464. Springer, 2020.

Lin Yang, Yizhe Zhang, Jianxu Chen, Siyuan Zhang, and Danny Z Chen. Suggestive annotation: A deep active learning framework for biomedical image segmentation. In *International conference on medical image computing and computer-assisted intervention*, pages 399–407. Springer, 2017.

Ofer Yehuda, Avihu Dekel, Guy Hacohen, and Daphna Weinshall. Active learning through a covering lens. In *Advances in Neural Information Processing Systems*, 2022.

Donggeun Yoo and In So Kweon. Learning loss for active learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 93–102, 2019.

Kaichao You, Yong Liu, Jianmin Wang, and Mingsheng Long. Logme: Practical assessment of pre-trained models for transfer learning. In *International Conference on Machine Learning*, pages 12133–12143. PMLR, 2021.

Kaichao You, Yong Liu, Ziyang Zhang, Jianmin Wang, Michael I Jordan, and Mingsheng Long. Ranking and tuning pre-trained models: a new paradigm for

exploiting model hubs. *The Journal of Machine Learning Research*, 23(1): 9400–9446, 2022.

Michelle Yuan, Hsuan-Tien Lin, and Jordan Boyd-Graber. Cold-start active learning through self-supervised language modeling. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7935–7948, 2020.

Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. In *British Machine Vision Conference 2016*. British Machine Vision Association, 2016.

Jure Zbontar, Li Jing, Ishan Misra, Yann LeCun, and Stéphane Deny. Barlow twins: Self-supervised learning via redundancy reduction. In *International Conference on Machine Learning*, pages 12310–12320. PMLR, 2021.

Xiaohua Zhai, Avital Oliver, Alexander Kolesnikov, and Lucas Beyer. S4l: Self-supervised semi-supervised learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1476–1485, 2019.

Xiaohua Zhai, Alexander Kolesnikov, Neil Houlsby, and Lucas Beyer. Scaling vision transformers. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12104–12113, 2022.

Xiaohang Zhan, Jiahao Xie, Ziwei Liu, Yew-Soon Ong, and Chen Change Loy. Online deep clustering for unsupervised representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 6688–6697, 2020.

Beichen Zhang, Liang Li, Shijie Yang, Shuhui Wang, Zheng-Jun Zha, and Qingming Huang. State-relabeling adversarial active learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8756–8765, 2020.

Bowen Zhang, Yidong Wang, Wenxin Hou, Hao Wu, Jindong Wang, Manabu Okumura, and Takahiro Shinozaki. Flexmatch: Boosting semi-supervised learning with curriculum pseudo labeling. *Advances in Neural Information Processing Systems*, 34, 2021.

Jifan Zhang, Yifang Chen, Gregory Canal, Arnav Mohanty Das, Gantavya Bhatt, Stephen Mussmann, Yinglun Zhu, Jeff Bilmes, Simon Shaolei Du, Kevin Jamieson, et al. Labelbench: A comprehensive framework for benchmarking adaptive label-efficient learning. *Journal of Data-centric Machine Learning Research*, 2024.

Renrui Zhang, Wei Zhang, Rongyao Fang, Peng Gao, Kunchang Li, Jifeng Dai, Yu Qiao, and Hongsheng Li. Tip-adapter: Training-free adaption of clip for

few-shot classification. In *European Conference on Computer Vision*, pages 493–510. Springer, 2022a.

Renyu Zhang, Aly A Khan, Robert L Grossman, and Yuxin Chen. Scalable batch-mode deep bayesian active learning via equivalence class annealing. In *The Eleventh International Conference on Learning Representations*, 2022b.

Wenqiao Zhang, Lei Zhu, James Hallinan, Shengyu Zhang, Andrew Makmur, Qingpeng Cai, and Beng Chin Ooi. Boostmis: Boosting medical image semi-supervised learning with adaptive pseudo labeling and informative active annotation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20666–20676, 2022c.

Ye Zheng, Xi Huang, and Li Cui. Visual language based succinct zero-shot object detection. In *Proceedings of the 29th ACM International Conference on Multimedia*, pages 5410–5418, 2021.

Dengyong Zhou, Olivier Bousquet, Thomas Lal, Jason Weston, and Bernhard Schölkopf. Learning with local and global consistency. *Advances in neural information processing systems*, 16, 2003.

Kaiyang Zhou, Jingkang Yang, Chen Change Loy, and Ziwei Liu. Learning to prompt for vision-language models. *International Journal of Computer Vision*, 130(9):2337–2348, 2022.

Jia-Jie Zhu and José Bento. Generative adversarial active learning. *arXiv preprint arXiv:1702.07956*, 2017.

Muzhi Zhu, Hengtao Li, Hao Chen, Chengxiang Fan, Weian Mao, Chenchen Jing, Yifan Liu, and Chunhua Shen. Segprompt: Boosting open-world segmentation via category-level prompt learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 999–1008, 2023.

Yu Zhu, Jinghao Lin, Shibi He, Beidou Wang, Ziyu Guan, Haifeng Liu, and Deng Cai. Addressing the item cold-start problem by attribute-driven active learning. *IEEE Transactions on Knowledge and Data Engineering*, 32(4):631–644, 2019.