UNIVERSITY OF OKLAHOMA
GRADUATE COLLEGE

NOVEL ALGORITHMS FOR INTERPRETABLE AND SEMI-SUPERVISED MACHINE LEARNING

A DISSERTATION
SUBMITTED TO THE GRADUATE FACULTY
in partial fulfillment of the requirements for the
Degree of
DOCTOR OF PHILOSOPHY

by Jay Calder Rothenberger

Norman, Oklahoma

2026

Downloaded from SHAREOK. Persistent Handle URL: https://hdl.handle.net/11244/342655

NOVEL ALGORITHMS FOR INTERPRETABLE AND SEMI-SUPERVISED MACHINE LEARNING

A DISSERTATION APPROVED FOR THE
SCHOOL OF COMPUTER SCIENCE

BY THE COMMITTEE CONSISTING OF

Dr. Dimitrios I. Diochnos, Chair

Dr. Andrew H. Fagg

Dr. Chao Lan

Dr. Sina Khanmohammadi

Dr. Miroslav Kramar

©Copyright by JAY CALDER ROTHENBERGER 2026

All Rights Reserved.

Acknowledgments

My contributions to science would have been impossible without the support of my wife Melissa. Not only has she contributed directly to the production of the text of this document, but she has also made direct contributions to the research that it is composed of. She has supported me mentally, emotionally, and financially though my most difficult endeavors. I can only hope to be able to provide support commensurate with what she has offered me in the next chapter of our lives.

I would like to thank my advisor Dr. Diochnos for his considerable patience and wisdom. I understand at the time of writing more than I did five years ago the investment that is required to mentor someone through their doctoral degree. I have the highest certainty that the opportunity he afforded me will continue to shape the trajectory of my life for decades to come.

I would like to acknowledge the support of my family. It is nonsensical to speak of what I would have been able to do without the help of my family. There is, of course, very little that I would have been able to do at all without that help. Though it goes without saying, it cannot hurt to say it anyway: Without the unyielding support of my family I would have been unable to move 1,200 miles to attend the University of Oklahoma.

Finally, I would like to acknowledge the support of the land on which the University of Oklahoma resides. I have a special connection with this land, as it has been home to multiple generations of my family. Long before the University of Oklahoma was established and before my family arrived here, the land on which the University now resides was the traditional home of the “Hasinais” Caddo Nation and “Kirikirʔi:s” Wichita & Affiliated Tribes. In addition to a home this land also served as a hunting ground, trade exchange point, and migration route for the Apache, Comanche, Kiowa and Osage nations. Today, 39 tribal nations dwell in the state of Oklahoma as a result

of settler and colonial policies that were designed to assimilate Native people. The actions of my government and my university have disproportionately benefited my family and harmed the Native people. Despite our shared history, I have witnessed only open hearts and minds from those same Native people; I have received only kindness. I acknowledge, honor and respect the diverse Indigenous peoples connected to the land on which it is built. I fully recognize, support and advocate for the sovereign rights of all of Oklahoma's 39 tribal nations. This acknowledgment is aligned with my university's core value of creating a diverse and inclusive community. It is an institutional – and personal – responsibility to recognize and acknowledge the people, culture, and history that make up the entire OU Community.

Table of Contents

Acknowledgments iv

List of Tables ix

List of Figures xii

Abstract xv

1 Introduction 1

2 Contributions 7
2.1 List of Contributions 7
2.2 Tables 9

3 Background 11
3.1 Image Classification 11
3.1.1 Supervised Image Classification 11
3.1.2 Semi-Supervised Image Classification 12
3.1.3 Student-Teacher Frameworks 13
3.2 Self-Supervised Learning 14
3.2.1 Pretext Tasks and Contrastive Learning 14
3.2.2 Self-Distillation: DINO 15
3.2.3 Language-Supervised Learning: CLIP and SigLIP 16
3.3 Multi-View Learning 17
3.3.1 Views 18
3.3.2 Co-Training 18
3.3.3 View Construction for Deep Learning 19
3.3.4 Representations for SSL and View Construction 19
3.4 Reinforcement Learning 20
3.4.1 Markov Decision Processes 20
3.4.2 Policy Gradient Methods 21
3.4.3 Policy Gradients for Pseudo-Labeling 21

4 Review of Pseudo-Labeling for Computer Vision 22

4.1 Chapter Introduction . . . 22
4.2 Preliminaries . . . 26
4.2.1 Fuzzy Partitions . . . 26
4.3 Semi-Supervised Regimes . . . 29
4.3.1 Sample Scheduling . . . 30
4.3.2 Weak Supervision . . . 31
4.3.3 Consistency Regularization . . . 33
4.3.4 Multi-Model Approaches . . . 34
4.4 Unsupervised and Self-Supervised Regimes . . . 36
4.4.1 Consistency Regularization . . . 36
4.4.2 Knowledge Distillation . . . 37
4.5 Open Directions Later Explored . . . 38
4.6 Summary . . . 39
4.7 Conclusion . . . 40

5 Meta Co-Training . . . 43
5.1 Chapter Introduction . . . 43
5.2 View Construction . . . 45
5.3 Proposed Method: Meta Co-Training . . . 46
5.4 Experiments . . . 52
5.4.1 View Construction Experiments . . . 53
5.4.2 Co-Training Baseline Experiments . . . 55
5.4.3 Meta Co-Training Experiments . . . 57
5.4.4 Comparison to Ensemble Methods . . . 59
5.4.5 Experiments on Additional Computer Vision Benchmarks . . . 61
5.4.6 Application of Co-Training Methods to Road Condition Classification . . . 62
5.5 Conclusion . . . 75

6 Policy Gradient Pseudo-Labeling . . . 78
6.1 Chapter Introduction . . . 78
6.2 Pseudo-Label Assignment as RL . . . 80
6.2.1 MPL and MCT as RL . . . 80
6.2.2 RLGSSL . . . 83
6.2.3 Drawbacks of Existing RL for SSL Algorithms . . . 84
6.3 Method . . . 85
6.3.1 Student-teacher Framework . . . 85
6.3.2 Example-level Reward Calculation . . . 86
6.3.3 Reinforcement Learning Formulation . . . 87
6.4 Experiments . . . 90
6.5 Conclusion . . . 95
6.6 Future Work . . . 95

7 Explaining Image Classification with Localized Additive Explanations 96
7.1 Chapter Introduction . . . 96
7.2 Related Work . . . 98
7.3 Method . . . 100
7.3.1 Training . . . 101
7.3.2 Inference . . . 105
7.3.3 Trainable Components . . . 106
7.4 Experimental Evaluation . . . 107
7.4.1 Setup for Experiments . . . 107
7.4.2 Datasets . . . 108
7.5 Discussion . . . 110
7.6 Conclusion . . . 112

8 Conclusion . . . 114
8.1 Future Work . . . 115

Reference List . . . 117

List of Tables

2.1 Author's core contributions presented in this dissertation. 9

2.2 Applications of the material presented in this dissertation. 10

2.3 Author's works not included in this dissertation. 10

4.1 Figure 4.1 Label Map 42

5.1 MLP performance on individual views. Top-1 accuracy on different subsets of the ImageNet data are shown.

5.2 Pairwise translation performance of linear probe on the ImageNet dataset. A linear classifier is trained on the output of an MLP that is trained to predict one view (columns) from another view (rows) by minimizing MSE. The top-1 accuracy (%) of the linear classifier is reported. Both the MLP and the linear classifier have access to the entire embedded ImageNet training set. 54

5.3 Linear probe evaluation for views. Top-1 accuracy on different subsets of the ImageNet data are shown. 55

5.4 Performance of different self-supervised and semi-supervised algorithms on ImageNet dataset. An asterisk (*) indicates models that were trained on top of pre-trained frozen backbone models. For all cross-validation experiments the mean testing accuracy as well as the 95% confidence interval are given. The implementation for REACT is not publicly available, so the reported performance is assumed to be the mean. Results that allow rejection of the null hypothesis for both baselines are shown in **bold**. In Chapter 6 I will present PGPL and compare it to MCT as well as the others presented here. PGPL will improve further on the results presented in this chapter. 58

5.5 CT, MCT, and ensemble top-1 accuracy of view combinations on 10% (1%) of ImageNet labels. As CT, MCT and the MLP ensemble do not depend on the order of the views, only all unique combinations are shown. In the case of the MLP ensemble these views are concatenated in order to form a larger unified view. 60

5.6 MLP ensemble performance on individual views. Top-1 accuracy on different subsets of the ImageNet data are shown.

5.7 Co-training and MCT evaluated on the ImageNet dataset using the CLIP | EsViT and DINOv2 | SwAV views. The Deep Ensemble is evaluated on the concatenation of all four views. All three methods are evaluated using 5-fold cross-validation. For all cross-validation experiments the mean testing accuracy as well as the 95% confidence interval are given. Results that allow rejection of the null hypothesis for both baselines are shown in **bold**. 61

5.8 Additional comparisons to REACT. The authors include experiments for zero-shot performance and 10% of available labels. Results shown are for 10% of labels. For Flowers102 this is 1-shot performance. For all cross-validation experiments the mean testing accuracy as well as the 95% confidence interval are given. Results that allow rejection of the null hypothesis for both baselines are shown in **bold**. 62

5.9 Additional comparisons to Semi-ViT using 10% (1%) of the available labels for training. For the iNaturalist only the 1010 most frequent classes were used. For all cross-validation experiments the mean testing accuracy as well as the 95% confidence interval are given. Results that allow rejection of the null hypothesis for both baselines are shown in **bold**. 63

5.10 Class frequencies within the labeled data. Minority classes are oversampled by the labelers, so this class distribution is not the same as the class distribution for the unlabeled data. 68

5.11 Comparison of the best sample mean accuracies for supervised (SL), supervised + self supervised, CT, and MCT on the first four label subsets. Sample mean accuracy increases by applying techniques from self-supervised learning, CT, and MCT. 71

5.12 Comparison of all three models and three algorithms. For all cross-validation experiments the mean testing accuracy as well as the 95% confidence interval are given. Results that allow rejection of the null hypothesis for both baselines are shown in **bold**. 75

6.1 Experimental comparisons on the Food101, and STL-10 datasets. For all cross-validation experiments the mean testing accuracy as well as the 95% confidence interval are given. Results that allow rejection of the null hypothesis for both baselines are shown in **bold**. 91

6.2 Experimental comparisons on the iNaturalist 2021 dataset. For iNaturalist 2021 only the 1010 most frequent classes were used. For all cross-validation experiments the mean testing accuracy as well as the 95% confidence interval are given. Results that allow rejection of the null hypothesis for both baselines are shown in **bold**. 91

6.3 Experimental comparisons on the CIFAR-10 and CIFAR-100 datasets. For all cross-validation experiments the mean testing accuracy as well as the 95% confidence interval are given. Results that allow rejection of the null hypothesis for both baselines are shown in **bold**. 92

6.4 Performance of different methods compared on the ImageNet-1% and ImageNet-10% benchmarks. Self-supervised models used for semi-supervised learning appear on the upper half of the table. Models on the lower half of the table use labeled and unlabeled data during classification training. For all cross-validation experiments the mean testing accuracy as well as the 95% confidence interval are given. Results that allow rejection of the null hypothesis for both baselines are shown in **bold**. 93

7.1 Class frequencies for Cats and Dogs. 109

7.2 Evaluation results. The testing accuracy of the LAX model is compared to its accuracy after the pixels identified as relevant have been masked. Recall that the best accuracy LAX model is the one that has the highest accuracy using $\boldsymbol{p_0}$ for prediction, while the best occluded LAX model is the one that maximizes Q_{LAX} (see Definition 7.3.4) and still uses $\boldsymbol{p_0}$ for prediction. Hence, the accuracy after masking the relevant evidence is equivalent to the accuracy if $\boldsymbol{p_2}$ were used for prediction by either model. For $\boldsymbol{p_2}$ accuracy, closer to $\frac{1}{k}$ is better where applicable. DenseNet and Interpretable DenseNet models do not have a notion of $\boldsymbol{p_2}$ accuracy, so only their test set accuracy is displayed. 109

7.3 Compute time comparison for all methods at 128 by 128 resolution on a batch of size 32 examples. Times recorded on a machine with one Nvidia Ampere A100 GPU. 111

List of Figures

4.1 Family tree of pseudo-labeling methods. Please see Table 4.1 for a table with section links and references. The methods shown in the figure from left to right are presented in the text in that order. Section 4.3 is devoted to semi-supervised learning methods and Section 4.4 to unsupervised learning methods. Furthermore, the third level split in the figure indicates in which subsection we discuss the various methods. For example, consistency regularization methods in SSL are discussed in Section 4.3.3, while consistency regularization methods in UL are discussed in Section 4.4.1. 22

5.1 Using pre-trained models for constructing views. Unlabeled and labeled sets are embedded using two different pre-trained models. This transforms a single-view dataset into two compressed representations which are used as views for co-training and meta co-training. 46

5.2 At each step $t \in \{1, \dots, T\}$ of MCT the models that correspond to the so-far learnt parameters $\theta_{1;t}$ and $\theta_{2;t}$ play the role of the student and the teacher simultaneously using batches for their respective views. Pseudo-labeling occurs on complementary views so that the teacher can provide the student with labels on an unlabeled batch. Labeled batches may, or may not, use complementary views as the purpose that they serve is to calculate the risk of the student model on the labeled batch and this result signals the teacher model to update its weights accordingly. 49

5.3 Top-1 accuracy of CT iterations on the CLIP and DINOv2 views for a training run on the ImageNet 10% dataset. 56

5.4 Progression of a training run for MCT using the CLIP and DINOv2 views as a function of the training step. Models are trained on 10% of the ImageNet labels. 57

5.5 The top-1 accuracy of the CT predictions for each iteration of CT. 10% of available Food101 labels were used for training. The method exhibited performance improvement over multiple iterations of pseudo-labeling and retraining. 64

5.6 Three New York State Department of Transportation (*NYSDOT*) images labeled by road surface condition with both of their corresponding views. The images on the first row are from the camera site whose location in Figure 5.7 is marked with a heart. Each of the classes in the labeled data are represented for each site. The images along the second row are from the closest camera location to that site. All labeled images in the dataset are classified as having a (a, d) *dry*, (b, e) *snowy*, or (c, f) *wet* road condition. 65

5.7 The camera site whose images have been labeled by human experts is shown with a heart near the center of the figure. The yellow stars along the perimeter of the figure depict the five closest sites to the camera with expert labeled images. Note that there are two camera sites very close to each other on the right-hand side of the figure. In this particular configuration of camera sites the starred site that is closest to the heart site, is the one that is shown at the top part of the figure. The distance between these two sites is slightly less than 0.6 miles. The ground truth labels for this site are assumed to be similar to those at the heart site since it is unlikely that road conditions due to weather are hyper-local events. Image from Google Maps [53]. 67

5.8 Four examples of conditions that are present in the unlabeled data that correspond to input types that are not represented in the labeled training data. Some conditions only present in the unlabeled data can obscure the road surface making it impossible to determine its condition. 68

5.9 The accuracy gained by using methods in combination for the lowest label budgets. At the lowest label budgets the complementary benefits of the self-supervised learning and multi-view semi-supervised learning (CT and MCT) are most pronounced. Table 5.11 has the numeric values. 72

5.10 Comparison of co-training methods' accuracies as a function of their training set sizes. Shaded regions indicate 95% confidence intervals. . . 76

6.1 An overview of my proposed PGPL method. Both models function as student and teacher. As a student, each model learns from the joint pseudo-label of the two models $\hat{Y}$. As a teacher, each model learns using $\mathcal{L}_{rl}$ to produce better pseudo-labels. 85

7.1 Examples from the Cats and Dogs dataset. The LAX explanation is shown in the middle column. The last column shows the explanation overlaid on top of the input image. The probability assigned by the model to each one of the two classes is shown on the right. 98

7.2 Model Architecture. Shown are three prediction steps of the model on a given image. In the first step the model is presented with the image, in the second step it is presented with the image with the area the model indicated for the predicted class masked out, and in the final step the model is presented with the image with the area the model indicated as evidence for all classes removed. 101

7.3 Successful and unsuccessful occlusions from the Cats and Dogs task. Images are ordered top to bottom by their descending prediction probabilities. The first column contains the input image. The second column contains the output of the LAX model and the 'Receptive Field' of the Interpretable DenseNet. The last four columns are XAI methods (Occlusion Sensitivity, Saliency Maps, LIME, SHAP in that order) applied to DenseNet. Time in seconds to produce the explanation for the given instance is included for each method. 110

Abstract

Pseudo-labeling (PL) has historically been considered synonymous with model-provided weak supervision in the context of semi-supervised learning (SSL). In that context, pseudo-labels are simply an inferior version of human supervision. Framing pseudo-labeling this way provides an overly restrictive view of how categorical model outputs can and have been used in prior work. In this dissertation I argue that pseudo-labels can be effectively applied in ways other than as weak supervision. Recognizing the shared use of pseudo-labels across algorithms not historically grouped together yields new insights into how we can use them.

I present a formal definition of pseudo-labels and a taxonomy of pseudo-labeling algorithms. Under this definition the supervisory role of the pseudo-label is not essential; the important aspect of pseudo-labels is that they are the product of a model. That model can be optimized to encourage desired properties of those pseudo-labels such as their utility to the learning process or explainability. My definition allows the conceptualization of new PL algorithms, three of which are presented here. Meta co-training (MCT) is a novel co-training-style PL algorithm for SSL that trains two collaborating models. Each model optimizes its pseudo-labels to reduce the loss of the other model. Policy gradient pseudo-labeling (PGPL) draws connections between prior PL works and reinforcement learning (RL) methods; these connections lead to an RL-inspired method that learns to maximize the value of pseudo-labels during training. Localized additive explanations (LAX) applies PL to the problem of explainable artificial intelligence, training a model to jointly explain and predict.

My contributions support the central claim that PL is a versatile tool that can be applied to machine learning algorithms both in SSL and outside of SSL.

Chapter 1

Introduction

The remarkable success of *deep neural networks* (DNNs) across a wide range of perceptual tasks has reshaped the landscape of *machine learning* (ML). In domains such as *computer vision* (CV), where the manual design of algorithms is particularly difficult, ML has enabled the development of systems that approximate solutions to problems previously considered intractable. The creation of labeled training data is a time-consuming and labor-intensive process. In CV in particular, the annotation of images requires significant human effort. Scaling human effort is expensive in resources and time, and typically large corpora of labeled data are required to train performant ML models. The cost of annotation is a bottleneck that limits the applicability of ML to tasks where large labeled datasets can be assembled, and excludes many scientifically and economically important problems for which such resources are unavailable. Reducing the dependence of ML systems on labeled data is therefore a problem of practical importance.

Semi-supervised learning (SSL) addresses the label scarcity problem directly by learning from both labeled and unlabeled data. Unlabeled data, data that is collected without human annotation or demonstration, are typically abundant and inexpensive relative to labeled data. SSL algorithms exploit the structure present in unlabeled data to inform the training of models with access to limited supervision. In many practical scenarios the amount of supervision a training algorithm has access to is fixed. The practical goal of SSL becomes to increase the performance of models trained using this limited supervision through incorporating unlabeled data in the learning process. This dissertation contributes to the growing body of work addressing the label scarcity problem. The contributions of this dissertation include a thorough review of *pseudo-labeling* (PL) for CV, two novel algorithms for PL, an application of those

algorithms to a practical problem for which they are suited, and a novel method for training interpretable neural networks for image classification.

The use of PL for SSL has been established for several decades now [124]. The canonical PL algorithm for SSL is *self-training* [124]. In self-training a model is first trained from labeled data and then predicts pseudo-labels for an unlabeled set. This unlabeled set along with the pseudo-labels are then added to the labeled set. The model is then re-trained using the combined labeled and pseudo-labeled data. This approach was first applied to learning DNN in Lee [2013]. *Co-training* [14] extends the idea of self-training to multiple 'views.' In co-training each model provides pseudo-labels to the other rather than re-training on their own pseudo-labels. Blum and Mitchell show that under some assumptions about the views used for co-training this procedure always results in more accurate models than supervised training alone. A weakness of self-training is that errors in the teacher's pseudo-labels are reinforced through subsequent training of the student, a phenomenon often referred to as *confirmation bias* [4]. *Meta pseudo-labels* (MPL) [105] addresses this weakness by allowing the teacher network's parameters to be updated based on the student's performance on labeled data. MPL and consistency-regularization approaches such as FixMatch [133] and noisy-student training [157] have driven much of the recent progress in PL-based SSL for image classification. This progress is usually measured as being able to achieve a new best validation accuracy on common CV benchmark tasks using only some of the available labels and treating the rest of the data as unlabeled.

PL is the process of utilizing categorical model outputs as the targets for learning rather than ground-truth supervision; in this dissertation I will refer to algorithms that utilize PL in this way as 'PL algorithms.' Though PL is typically considered a method for SSL, the use of model predictions in place of ground-truth is more broadly applicable. For example, *knowledge distillation* [62] (KD) uses the predictions of a

teacher model to learn a student model. In this setting the student is learned with the help of pseudo-labels from the teacher. PL is also used in many *self-supervised learning* algorithms such as DINO [24, 100, 129] and SimCLR [32, 33]. In KD the pseudo-label is designed to serve as supervision for the student so that it can learn to identify a set of classes learned by the teacher from ground truth. Interestingly, the SSL method above learn to partition their input spaces into classes that do not correspond to a particular ground truth labeling. What is important is not to which category the inputs belong, but rather that the ability to categorize inputs consistently. In the case of SimCLR consistency means to categorize augmented versions of the same image similarly. In the case of DINO consistency means that a 'student' model should predict similarly to a teacher model that is updated as an exponential moving average of the student's parameters. Both are a form of *consistency regularization* [81]. These methods outside of SSL are a first glimpse at what is possible when using PL.

In Chapter 4 I present a review of PL for CV [119] that was published in the *Journal of Artificial Intelligence Research* (JAIR). In this review my co-authors and I present a formal definition of pseudo-labels that unifies several research thrusts in CV. This formal definition provides a principled framework through which pseudo-labeling can be systematically understood, and lets me re-contextualize as PL several techniques not conventionally situated within the pseudo-labeling literature, including self-supervised and unsupervised methods. I present a novel taxonomy of these thrusts and review prior work in the area. Using the formal definition and the taxonomy I enumerate several promising directions for future work in PL. In Chapters 5 and 6 I present algorithms that are motivated by the research gaps identified in Chapter 4.

In Chapter 5 I present *meta co-training* (MCT), a novel multi-view SSL algorithm that was published in the 28th *European Conference on Artificial Intelligence* (ECAI). Further, in this paper I address a fundamental limitation of classical co-training under

conditions of label scarcity. Standard co-training relies on the availability of independent and sufficient views of the data. This assumption is rarely satisfied in practice. My work demonstrates that such views can be constructed inexpensively from pre-trained models. This observation makes the co-training framework broadly applicable to common benchmark CV settings. To overcome the sub-optimal pseudo-labeling behavior that arises when views differ substantially in their information content, MCT extends the Meta Pseudo Labels framework to the two-view setting. MCT trains each constituent model to generate better pseudo-labels for the other through a bi-level optimization procedure. MCT achieves state-of-the-art performance on ImageNet-10% and establishes new benchmarks on several fine-grained image classification datasets.

In Chapter 5 I also present an application of co-training algorithms to road surface condition classification. This dissertation demonstrates the successful application of multi-view SSL methods to this real-world problem. Road surface condition classification is a task that naturally admits multiple views, contains unlabeled instances belonging to classes not present during training, and represents a domain in which automated ML systems offer practical value. In this domain, the human effort of labeling presents a bottleneck, and MCT both reduces the required labeling effort and improves the accuracy of the resulting models. The presence of these conditions and success of co-training algorithms under those conditions demonstrate the undeniable value of those algorithms. Experimental results show that these novel SSL methods reduce the number of labels required to reach 95% accuracy by up to 60% and reduce few-shot error rate by 38.6%, demonstrating that the gains achieved on benchmark tasks translate meaningfully to real-world settings.

In Chapter 6 I present *policy gradient pseudo-labeling* (PGPL), a RL-based SSL algorithm. Building on the insights of MCT, PGPL learns a pseudo-labeling policy that maximizes an example-level reward for each pseudo-label. This dissertation further

demonstrates that prior PL methods, including MCT and *reinforcement learning guided semi-supervised learning* (RLGSSL), learn pseudo-labeling policies that maximize a *batch-level* reward. This coarser objective fails to adequately capture the value of individual pseudo-labels to the learning process. The most important innovation of PGPL is a sensible formulation of the example-level reward and a tractable approximation for computing it. PGPL achieves superior performance across a range of SSL benchmark tasks, demonstrating the effectiveness of fine-grained pseudo-label reward optimization.

A concern that has arisen given the widespread usage of DNNs is the difficulty of explaining their predictions. As DNN models are increasingly relied upon in safety-critical scenarios, treating them only as black boxes' becomes a liability. The field of *explainable artificial intelligence* (XAI) seeks to provide explanations of ML predictions that are understandable to humans. Prior work in XAI has produced a number of 'post-hoc' methods capable of explaining the predictions of DNN models such as GRAD-CAM [125], LIME [112], SHAP [90], Integrated Gradients [137] (IG). There have also been efforts to train DNN models that are interpretable by design, such as *interpretable convolutional neural networks* [174] (ICNN) and ProtoPNet [29]. Post-hoc methods are broadly applicable, but their explanations can be time-consuming to compute [90, 112] or rely on gradient information that only tells part of the story [76]. Models that are interpretable by design can be learned to expose the decision-making process more directly, but usually at the cost of imposing architectural constraints.

In Chapter 7 I present *localized additive explanations* (LAX), a novel PL method for training interpretable neural networks. LAX uses pseudo-labels to identify the regions of an input that are essential to its classification. Specifically, LAX trains a neural network to predict the areas of an image that, if occluded, would prevent the learner from predicting the correct class. By construction these regions are essential to the prediction and thus form a truthful explanation of the model's decision. LAX

is applicable to any image classification problem but requires training a model with a structure that jointly produces predictions and visual explanations. By incorporating explanation generation directly into the prediction step, LAX bypasses the need for an external explanation procedure, achieves higher granularity of evidence localization, and reduces time-to-explanation, at the cost of a small reduction in classification accuracy. LAX is trained against objectives ensuring that the signals identified by the explanation are *correct* (evidence for a class), *complete* (essential evidence for predicting that class), and *exclusive* (not identifying anything other than evidence for that class). When this joint objective is met, the same network produces both an accurate prediction and a truthful explanation of that prediction.

Chapter 2

Contributions

This dissertation includes several contributions to semi-supervised machine learning from an algorithmic and scholarly perspective. Alongside novel algorithms, this dissertation also includes applications of these algorithms to a practical scenario. Finally, this dissertation presents a contribution to interpretable machine learning.

2.1 List of Contributions

1. A Formal Definition of Pseudo-Labels My formal definition of pseudo-labels gives a principled way to identify algorithms that use pseudo-labels. This definition is useful for identifying the similarities between algorithms that serve different purposes but have similar mechanisms. Defining pseudo-labeling as a fuzzy partitioning of the input space illuminates the connections between SSL, self-supervised learning, knowledge distillation, and consistency regularization.

2. A Novel Taxonomy of Pseudo-Labeling Algorithms Having identified several parallel thrusts of PL research falling under the formal definition, I further categorize those methods into sub-categories based on how they are using PL. Pseudo-labels are a versatile tool that are used in various ways throughout these thrusts. The greatest diversity in prior work is present in the SSL literature. This diversity illuminates many varied ways in which PL can be applied outside of SSL.

3. Demonstrating the Effectiveness of Foundation Models for View Construction Standard benchmark tasks for CV do not admit multiple views to use for co-training. There is a body of literature about view construction that is con-

cerned with the creation of suitable views. These methods of view construction include manual feature subsetting [42], automatic feature subsetting [31], random feature subsetting [17], random subspace selection [18], and adversarial examples [108]. I demonstrate that representations learned in a self-supervised way serve as strong views for co-training algorithms. Due to the popularity of self-supervised learning many such models [55, 32, 23, 87, 109, 100, 59] are available to use for constructing views.

4. The Meta Co-Training Algorithm I introduce the MCT algorithm for SSL. I demonstrate its effectiveness on a range of SSL benchmark tasks for computer vision. I use self-supervised learned representations as views form those SSL benchmark tasks.

5. A Case Study that Demonstrates the Applicability and Effectiveness of Co-Training Algorithms The previous works relied on the creation of multiple views using a single view. I also show that there are problems that naturally provide multiple views. Further I demonstrate that these co-training algorithms are effective in real-world scenarios that provide multiple views. I show that they are sufficiently robust to realistic data challenges, such as out-of-distribution unlabeled instances.

6. A Reward Function for Pseudo-Labeling Policies RLGSSL, the only prior work that has explicitly formulated the pseudo-label assignment problem as reinforcement learning, did not use the impact of assigning that pseudo-label on the learning process as the reward. Instead RLGSSL defines the reward as the mean-squared-error (MSE) between the model's prediction and a MixUp [172] label. The reward function used in PGPL directly measures the reduction in loss associated with the assignment of a pseudo-label as the reward that the pseudo-labeling policy maximizes. Unlike the reward for RLGSSL, this reward cannot be optimized using backpropagation. Further, a policy maximizing this reward clearly assigns pseudo-labels such that a model learning

from those labels achieves minimal loss. I present a SHAP-like approximation of this reward that makes its computation tractable.

7. The Policy Gradient Pseudo-Labeling Algorithm I introduce the PGPL algorithm for SSL. I draw connections between prior methods for pseudo-labeling and reinforcement learning and identify avenues of improvement I demonstrate its effectiveness of these improvements on a range of SSL benchmark tasks for computer vision.

8. A Novel Explainable Artificial Intelligence Algorithm for Image Classification I introduce Localized Additive Explanations (LAX). LAX provides verifiably true explanations by training a model to generate occlusion masks that would change its prediction if applied. In this way, LAX is able to provide explanations that are truthful, minimal, and complete representations of the information the model relies on.

2.2 Tables

Table 2.1: Author's core contributions presented in this dissertation.

Short Title	Authors	Status	Reference
A Review of Pseudo-Labeling for Computer Vision	**Rothenberger**, Kage, Andreadis, Diochnos	Journal Pub.	[119]
Meta Co-Training	**Rothenberger**, Diochnos	Conference Pub.	[117]
Policy Gradient Pseudo-Labeling	**Rothenberger**	In Submission	-
Localized Additive Explanations	**Rothenberger**, Wilson Reyes, Diochnos	In Submission	-

Table 2.2: Applications of the material presented in this dissertation.

Short Title	Authors	Status	Reference
Improving Road Surface Classification with Co-Training Algorithms	**Rothenberger**, Le, Sutter, Sulia, Diochnos	Conference Abstract	[120]
Application of Co-Training Methods to Road Condition Classification	**Rothenberger**, Le, Sutter, Diochnos	In Submission	-

Table 2.3: Author's works not included in this dissertation.

Short Title	Authors	Status	Reference
Road Surface Condition Detection with Machine Learning	Sutter, Sulia, Wirz, Bassill, Thorncroft, **Rothenberger**, Pryzbylo, Cains, Radford, Evans	Journal Pub.	[139]
The Role of Lightning in Hail Nowcasting	**Rothenberger**, Grimit, Murphy, Wallace	Conference Abstract	[118]
Foundation Model for Detection of Methane Sources	**Rothenberger**, Crowell, Keely, Cusworth, Howell	Conference Poster	[116]
Classifying Road Surface Conditions with Self-Training	Ferrerra, **Rothenberger**, Wilson Reyes, Sutter, Fagg, Diochnos	Conference Abstract	[47]
Predicting Forecast Error for the HRRR LSTM Neural Networks	Evans, **Rothenberger**, Sulia, Bassill, Thorncroft, Gaudet	In Preparation	[46]

The papers relating to the core contributions of this dissertation are contained in Table 2.1. Papers relating to my contributions along these lines are contained in Table 2.2. Not all of the work performed during my studies fall neatly into either of these categories. In Table 2.3 is included work that is not entirely related to my core contributions. Below is included a list of core contributions.

Chapter 3

Background

This chapter introduces the technical background necessary to understand the contributions of this dissertation. The following sections introduce notation and definitions required to understand four key areas: image classification, self-supervised learning, multi-view learning, and reinforcement learning. These areas are foundational to the content in Chapters 4-7.

3.1 Image Classification

3.1.1 Supervised Image Classification

Image classification is the canonical CV task. Given an input image x_i belonging to an image space $\mathcal{X}$, the goal is to learn a function that assigns x_i a label $\mathcal{Y}_i$ belonging to the label space $\mathcal{Y}$ from a fixed set of C classes, $\mathcal{Y} = \{1, \ldots, C\}$. In the supervised setting, The labeled data $\mathbf{L} = \{(x_i, \mathcal{Y}_i)\}_{i=1}^n$ consists of n image-label pairs. A model $f_\theta : \mathcal{X} \to \Delta^C$ parameterized by a vector of weights θ maps each image to a probability distribution over the C classes, where Δ^C denotes the C-dimensional probability simplex. Occasionally in this dissertation I will avail myself of a convention that is common in many deep learning frameworks: If an argument of the loss is an integer then that argument is to be interpreted as the one-hot-encoding of that integer. The one-hot encoding for $\mathcal{Y}_i$ is $y_i \in \Delta^C$ with the $\mathcal{Y}_i$th coordinate of y_i being 1.

The predicted class for an image x_i is $\hat{\mathcal{Y}}_i \in \arg\max f_\theta(x)$. Training proceeds by minimizing the empirical risk

$$\mathcal{L}(f_\theta; \mathbf{L}) = \frac{1}{|\mathbf{L}|} \sum_{(x_i, y_i) \in \mathbf{L}} \ell(f_\theta(x_i), \mathcal{Y}_i), \tag{3.1}$$

where ℓ is a per-example loss function. For classification, ℓ is typically the cross-entropy loss:

$$\ell(f_\theta(x_i), \mathcal{Y}_i) = -\sum_{k=1}^{C} y_{ik} \cdot \log[f_\theta(x_i)_k]. \tag{3.2}$$

The models minimizing (3.1) are typically approximated through gradient descent. The performance of a classifier trained in this way is limited by both the quality and quantity of the labeled data available. DNNs in particular require large labeled datasets to train performant models. When labeled data are scarce, a model trained only on $\mathbf{L}$ will typically exhibit poor generalization.

3.1.2 Semi-Supervised Image Classification

SSL addresses the label scarcity problem by augmenting labeled data $\mathbf{L}$ with unlabeled data $\mathbf{U}$. For algorithms that utilize multiple representations or *views* of the data, such as *co-training* (CT) [14], *meta co-training* (MCT) [117], and PGPL, we will use superscripts to differentiate between the views. For example, $\mathbf{L}^{(1)}$ for the first view and $\mathbf{L}^{(2)}$ for the second. One can think of views simply as independent representations of the same signal. In our case, these views will always be different learned representations of the same image, though in principle the representations need neither be learned nor be images. To make the correspondence between unlabeled instances clear, unlabeled data $\mathbf{U} = \{(u_j^{(1)}, u_j^{(2)})\}_{j=n+1}^{n+m}$ consists of tuples of views. For convenience sampling batches from $\mathbf{U}$ as will be denoted as $U^{(1)}, U^{(2)} \sim \mathbf{U}$ and from $\mathbf{L}$ as $X^{(i)}, Y^{(i)} \sim \mathbf{L}^{(i)}$. In that case $U^{(1)}$ and $U^{(2)}$ contain the representations from view 1 and view 2, respectively, for a set of instances drawn i.i.d. from $\mathbf{U}$. Similarly $X^{(i)}$ and $Y^{(i)}$ represent the instances and their respective labels for a batch drawn i.i.d. from $\mathbf{L}$.

SSL algorithms typically operate under several assumptions about the relationship between the input distribution and the decision boundary. The *manifold assumption* [28]

states that input points lie on some manifold of a lower dimension than the input dimension and that this lower-dimensional manifold is sufficient to represent the input points. The *cluster assumption* [28] states that data points within the same high-density region of the data distribution (a cluster) share a label. The *low-density separation assumption* [28] asserts that the decision boundary should not pass through high-density regions of $\mathcal{X}$. These assumptions motivate the use of unlabeled data: even without labels, $\mathbf{U}$ reveals the structure of $\mathcal{X}$, which can be used to regularize the learned decision boundary.

Generally, the SSL objective combines the supervised loss on $\mathbf{L}$ with an unsupervised regularization term on $\mathbf{U}$:

$$\mathcal{L}_{SSL}(f_\theta; \mathbf{L}, \mathbf{U}) = \mathcal{L}(f_\theta; \mathbf{L}) + \lambda \, \mathcal{L}_U(f_\theta; \mathbf{U}), \tag{3.3}$$

where $\mathcal{L}_U$ is some unsupervised loss term and $\lambda > 0$ is a scalar that controls the relative contribution of the two terms. The form of $\mathcal{L}_U$ varies substantially across methods and defines the character of the SSL algorithm.

3.1.3 Student-Teacher Frameworks

A recurring architectural pattern in PL-based SSL is the 'student-teacher framework'. In this setting, a 'teacher' model f_{θ_T} generates pseudo-labels for unlabeled data, and a 'student' model f_{θ_S} is trained on those pseudo-labels. The teacher can take several forms. In Mean Teacher [140], the teacher is an exponential moving average (EMA) of the student's weights, $\theta_{T;i+1} \leftarrow \alpha\theta_{T;i} + (1-\alpha)\theta_{S;i+1}$, producing a more stable target than the student provides for itself. The smoothing coefficient $\alpha \in (0, 1)$ controls how quickly the teacher tracks the student. In practice, values close to one are used so the teacher changes slowly and provides low-variance pseudo-label targets.

Meta Pseudo Labels (MPL) introduces several notions that are important to this

dissertation [105]. MPL refines the student-teacher framework for PL by jointly optimizing the student and the teacher. In standard pseudo-labeling (i.e. [83]), the teacher is first trained from scratch and then generates pseudo-labels for the student without any feedback on how useful those pseudo-labels were. MPL removes this limitation by training the teacher to generate pseudo-labels that maximize the student's performance on the labeled data. This bi-level optimization gives the teacher a feedback signal: pseudo-labels are valued not by the teacher's confidence but by how much they help the student learn to correctly classify a separate labeled set. MPL achieves state-of-the-art results on ImageNet in the semi-supervised regime and was influential in the design of Meta Co-Training.

3.2 Self-Supervised Learning

Self-supervised learning[1] is a paradigm for learning representations from unlabeled data. The core idea is to define a *pretext task* whose 'label' is derived entirely from the structure of the data itself, without any human annotation. By solving a well-chosen pretext task, a model learns to extract features that capture semantically meaningful properties of images. These features can then be used to train classifiers with limited labeled data, making self-supervised pre-training a natural companion to SSL.

3.2.1 Pretext Tasks and Contrastive Learning

Early self-supervised methods for CV defined pretext tasks based on image transformations: predicting the relative position of image patches [43], solving jigsaw puzzles [97], predicting rotation angle [51], and image colorization [175] are representative examples. Although these methods demonstrated that useful features could be learned without

[1]To avoid ambiguity, this dissertation uses *SSL* to refer exclusively to *semi-supervised learning* and refers *self-supervised learning* without abbreviation.

labels, the representations they produced fell substantially short of supervised learning on standard benchmarks.

Contrastive learning methods substantially narrowed this gap. These methods learn representations by training a shared encoder $g_\phi : \mathcal{X} \to \mathbb{R}^d$ to be invariant to a set of data augmentations $\mathcal{T}$. Broadly these data augmentations can be any label-preserving transformations. Common data augmentations include crops, rotations, and affine transformations. Given an image x, two independently sampled augmented views $\tilde{x} = t(x)$ and $\tilde{x}' = t'(x)$ with $t, t' \sim \mathcal{T}$ are constructed, and the encoder is trained so that the representations $z = g_\phi(\tilde{x})$ and $z' = g_\phi(\tilde{x}')$ are similar, while representations of different images are dissimilar. Chen et al. [2020] formalize this through their method SimCLR, which optimizes a loss they call NT-Xent; BYOL [55] eliminates the need for negative pairs using a momentum encoder. These methods established that rich visual representations could be learned at scale without supervision.

3.2.2 Self-Distillation: DINO

A particularly relevant family of methods for this dissertation learns representations through *self-distillation*: the model is its own teacher. *Self-**di**stillation with **no** labels* (DINO) [24] applies the student-teacher framework, introduced in Section 3.1, directly to representation learning. DINO maintains a student encoder g_{ϕ_S} and a teacher encoder g_{ϕ_T}, where the teacher's weights are an exponential moving average of the student's. For a given image x, multiple augmented views are constructed; the student processes all views while the teacher processes a higher-resolution un-augmented image. The student is trained to match the teacher's output distribution. The teacher and student both predict a set of logits to which the softmax function is applied. The softmax function is given by:

$$softmax(z)_i = \frac{\exp(z_i)}{\sum_i \exp(z_i)}. \tag{3.4}$$

The student suffers cross-entropy loss between its predicted distribution and the teacher's. Because the teacher is an exponential moving average of student weights, it provides more stable targets than the student itself over training iterations.

3.2.3 Language-Supervised Learning: CLIP and SigLIP

Complementary to the use of unlabeled images in SSL, one approach for learning visual representations uses the much larger resource of image-text pairs available on the internet. Rather than defining a pretext task over images alone, these methods train a visual encoder g_{ϕ_V} and a text encoder g_{ϕ_T} jointly to align image and text representations. *Contrastive language-image pre-training* (CLIP) [109] trains these encoders on an internet-scale dataset using a contrastive objective: for a batch $\mathcal{B}$ of size B image-text pairs $\{(x_i, c_i)\}_{i=1}^{B}$, the encoders are trained to maximize the cosine similarity (denoted as sim) of the B matching pairs (x_i, c_i) while minimizing the similarity of all $B^2 - B$ non-matching pairs. This loss encourages the model to learn representations of images and text that are similar when the image matches the text description. In this way if one has a string of text and a CLIP model, then one also has a mechanism by to identify images for which the text would be an appropriate caption. The contrastive objective for CLIP is given as:

$$\begin{aligned}\mathcal{L}_{\text{CLIP}} = -\frac{1}{2B}\sum_{i=1}^{B}\Bigg[&\log\frac{\exp(\text{sim}(g_\phi^V(x_i),\, g_\phi^T(c_i))/\kappa)}{\sum_j \exp(\text{sim}(g_\phi^V(x_i),\, g_\phi^T(c_j))/\kappa)} \\ &+ \log\frac{\exp(\text{sim}(g_\phi^T(c_i),\, g_\phi^V(x_i))/\kappa)}{\sum_j \exp(\text{sim}(g_\phi^T(c_i),\, g_\phi^V(x_j))/\kappa)}\Bigg],\end{aligned} \tag{3.5}$$

where $\kappa > 0$ is a temperature parameter. The softmax normalization in (3.5) couples

all pairs in the batch, requiring the model to simultaneously reason about the relative similarity of every image-text pair. At scale, the resulting visual representations generalize remarkably well; a CLIP-trained visual encoder achieves competitive *zero-shot classification* accuracy on many benchmarks. Zero-shot classification refers to the performance of a classification model that has not been trained on a dataset. CLIP achieves zero-shot 'generalization' by comparing image embeddings to text embeddings of class descriptions. Because CLIP has learned to represent text and images similarly, then it can be used to identify images that are similar to text descriptions and in many cases does not need additional training to perform classification.

Sigmoid loss for language-image pre-training (SigLIP) [169] revisits this objective, replacing the softmax-normalized loss with a pairwise sigmoid loss that treats each image-text pair independently:

$$\mathcal{L}_{\text{SigLIP}} = -\frac{1}{B}\sum_{i=1}^{B}\sum_{j=1}^{B}\log\ \sigma(y_{ij}\cdot \text{sim}(g_{\phi_V}(x_i), g_{\phi_T}(c_j))/\kappa)\,, \tag{3.6}$$

where $y_{ij} = +1$ if $i = j$ and $y_{ij} = -1$ otherwise, and σ is the sigmoid function, $\sigma(x) = \frac{1}{1+e^{-x}}$. Because (3.6) requires no global normalization across the batch, it is more computationally efficient than CLIP and scales better to large batch sizes.

3.3 Multi-View Learning

The goal of multi-view learning is to exploit multiple distinct representations, or *views*, of the same underlying data. When different views capture complementary information about an input, learning from their agreement can improve generalization beyond any single view. This section introduces the multi-view learning framework and the co-training algorithm, which is central to the contributions of this dissertation.

3.3.1 Views

A *view* is a representation of the input in some feature space. Let $z^{(1)} \in \mathbb{R}^{d_1}$ and $z^{(2)} \in \mathbb{R}^{d_2}$ be two such representations. Two views are *sufficient* for a classification task if the label y can be predicted from either $z^{(1)}$ or $z^{(2)}$ alone. Two views are *independent* if each view is conditionally independent of the other given the label. This can be understood intuitively as a single view giving no more information about the other than the label would have.

3.3.2 Co-Training

Co-training [14] is the classical multi-view SSL algorithm. Co-training trains two models $f_{\theta_1} : \mathbb{R}^{d_1} \to \Delta^C$ and $f_{\theta_2} : \mathbb{R}^{d_2} \to \Delta^C$, each trained on one view. Co-training iteratively has each model generate pseudo-labels for the unlabeled examples on which it is most confident and adds those pseudo-labeled examples to the training set of the other model. Under the sufficiency and conditional independence assumptions the errors made by one model on one view are independent of those made by the other model on the other view. The two models thereby teach each other by exploiting their complementary information to improve jointly.

At each iteration of co-training, model f_{θ_k} selects the p unlabeled examples from $\mathbf{U}$ on which its confidence $\max_c f_{\theta_k}(z_i^{(k)})$ is highest, assigns them pseudo-labels $\tilde{y}_i \in \arg\max_c f_{\theta_k}(z_i^{(k)})$, and transfers those pseudo-labeled examples to the training set of the other model $f_{\theta_{k'}}$, $k' \neq k$. This exchange is repeated until a stopping criterion is met. Under the independent and sufficient view conditions, Blum and Mitchell proved in Blum and Mitchell (1998) that co-training can drive the error of each individual classifier arbitrarily low using a polynomial number of unlabeled examples.

3.3.3 View Construction for Deep Learning

In practice, the sufficiency and conditional independence assumptions of co-training are rarely satisfied. For CV tasks defined over raw image features, naturally independent views typically are not available. This dissertation demonstrates that representations learned by self-supervised models serve as strong views for co-training. As described in Section 3.3.4, DINO, CLIP, and SigLIP each learn complementary inductive biases over the same images: DINO through visual self-distillation, CLIP and SigLIP through alignment with natural language. Given two such encoders g_{ϕ_1} and g_{ϕ_2}, the representations $z_i^{(1)} = g_{\phi_1}(x_i)$ and $z_i^{(2)} = g_{\phi_2}(x_i)$ define two views of the image x_i that are sufficient and approximately independent. The wide availability of pre-trained self-supervised models of this kind makes view construction inexpensive and the co-training framework broadly applicable.

3.3.4 Representations for SSL and View Construction

Models trained with self-supervised objectives empirically learn representations that generalize well and exhibit strong classification performance on a wide range of benchmark tasks. While these self-supervised 'foundation models' are specific to the data distribution they are trained on, the wide availability of images collected from the internet has made training foundation models for vision, in general, more realistic. Performance varies based on the exact unlabeled training set used, but the wide variety of available trained models and their strong performance on benchmark tasks make them attractive for many applications. Given a pre-trained encoder g_ϕ — whether trained with DINO, CLIP, SigLIP, or another method — a downstream classifier $h_\psi : \mathbb{R}^d \to \Delta^C$ can be trained on the learned representations $\{(g_\phi(x_i), y_i)\}_{i=1}^n$ using only the labeled data $\mathbf{L}$. Classifiers trained in this way require dramatically fewer

labeled examples to achieve competitive performance than those trained on raw image features [169, 110, 24, 100, 129].

3.4 Reinforcement Learning

Reinforcement learning (RL) is a framework for learning a decision-making policy through interaction with an environment. An agent observes a state, takes an action, receives a reward signal, and updates its policy to maximize the cumulative reward it receives over time. Although RL was developed primarily in the context of sequential decision-making problems, specific tools from RL are applicable to optimization problems in which the objective is not differentiable with respect to the policy parameters. This section introduces the RL framework and the policy gradient methods used in this dissertation.

3.4.1 Markov Decision Processes

The standard formalism for RL is the Markov Decision Process (MDP), defined by the tuple $(\mathcal{S}, \mathcal{A}, P, r, \gamma)$. Here $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, $P : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$ is the transition probability function, $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the reward function, and $\gamma \in [0, 1)$ is a discount factor. An agent interacting with an MDP observes a state $s_t \in \mathcal{S}$ at each time step t, selects an action $a_t \sim \pi_\phi(\cdot \mid s_t)$ according to a stochastic policy $\pi_\phi : \mathcal{S} \to \Delta^{|\mathcal{A}|-1}$ parameterized by ϕ, transitions to a new state $s_{t+1} \sim P(\cdot \mid s_t, a_t)$, and receives a scalar reward $r_t = r(s_t, a_t)$. The agent's objective is to find parameters ϕ that maximize the expected discounted reward,

$$J(\phi) = \mathbb{E}_{\tau \sim \pi_\phi} \left[\sum_{t=0}^{T} \gamma^t \, r_t \right], \tag{3.7}$$

where $\tau = (s_0, a_0, r_0, s_1, a_1, r_1, \ldots)$ is a trajectory generated by rolling out the policy π_ϕ

in the environment.

3.4.2 Policy Gradient Methods

Policy gradient methods optimize $J(\phi)$ by computing gradients with respect to the policy parameters ϕ and ascending those gradients. The policy gradient theorem [154] addresses this by providing a tractable expression for $\nabla_\phi J(\phi)$,

$$\nabla_\phi J(\phi) = \mathbb{E}_{\tau \sim \pi_\phi} \left[\sum_{t=0}^{T} \nabla_\phi \log \pi_\phi(a_t \mid s_t) \cdot R_t \right], \tag{3.8}$$

where $R_t = \sum_{t'=t}^{T} \gamma^{t'-t} r_{t'}$ is the discounted sum of future rewards from time step t given a_t. The expectation in (3.8) can be estimated by Monte Carlo sampling: trajectories τ are sampled by rolling out the current policy and the gradient estimate is computed from those samples. This estimator is commonly known as REINFORCE [154].

3.4.3 Policy Gradients for Pseudo-Labeling

The connection between RL and pseudo-labeling becomes clear when pseudo-label assignment is framed as a one-step decision-making problem. In this framing, the state s encodes information about an unlabeled example and the current model, the action a corresponds to the pseudo-label assigned to that example, and the reward $r(s, a)$ measures the benefit of that assignment to the learning process. Because the reward involves the outcome of a subsequent training step, it is challenging to differentiate with respect to the policy parameters. REINFORCE provides a mechanism for estimating this gradient without requiring differentiability of the reward. In Chapter 6 I will explore the implications of and challenges to applying RL in this way.

Chapter 4

Review of Pseudo-Labeling for Computer Vision

4.1 Chapter Introduction

In this chapter we will give a summary of the work published in Rothenberger et al. [2025]. This chapter will focus on introducing, establishing a formal definition and taxonomy for, and summarizing prior work in PL. PL is an approach that has seen widespread adoption across semi-supervised an unsupervised learning algorithms. PL has traditionally been considered a tool for SSL, but as we will see in the following sections it has also seen use in areas like self-supervised learning and knowledge distillation. In Figure 4.1 we provide a taxonomy of PL.

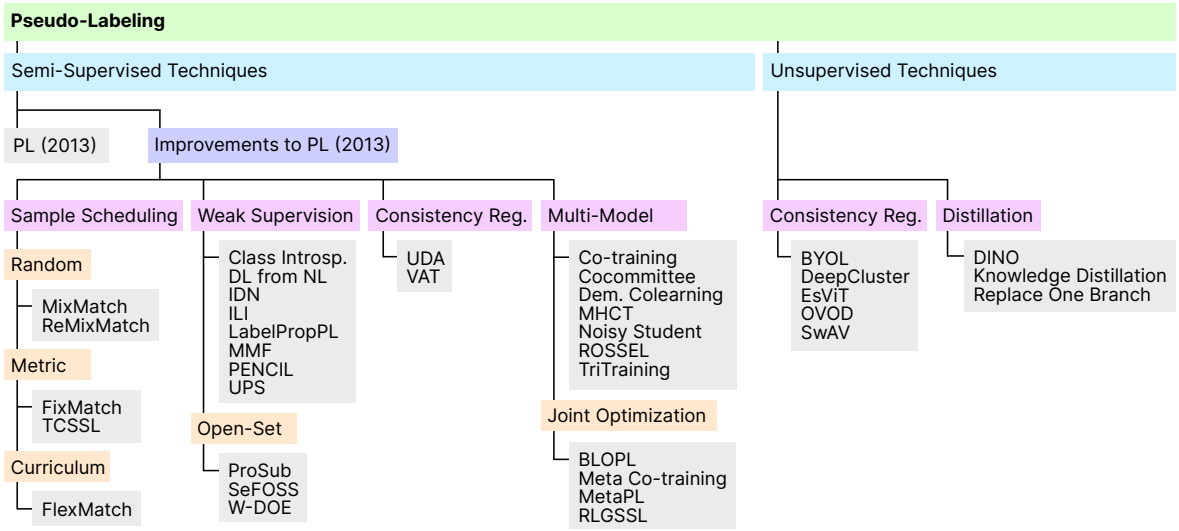


Figure 4.1: Family tree of pseudo-labeling methods. Please see Table 4.1 for a table with section links and references. The methods shown in the figure from left to right are presented in the text in that order. Section 4.3 is devoted to semi-supervised learning methods and Section 4.4 to unsupervised learning methods. Furthermore, the third level split in the figure indicates in which subsection we discuss the various methods. For example, consistency regularization methods in SSL are discussed in Section 4.3.3, while consistency regularization methods in UL are discussed in Section 4.4.1.

We will give a formal definition of *Pseudo-labels* (also known as *proxy labels*) in

Section 4.2, but generally, pseudo-labels can be thought of as model predictions used in the place of ground-truth. Because the idea of using an estimator to provide an approximation of an unknown ground truth is so natural, it is perhaps unsurprising that it has been applied in so many different ways. It is important in our context that we do not mistake the usefulness of pseudo-labels to be upper-bounded by that of additional human supervision. In many of the most interesting applications of pseudo-labels, such as MPL [105], MCT [117], and PGPL, the pseudo-labels have no guarantee of being particularly close to the ground truth labels. While many methods interpret pseudo-labels as weak supervision, such as those in Section 4.3.2, PL is far more interesting and general than that. We show that PL techniques largely share one or more of the following characteristics:

- *self-sharpening*, where a single model is used to create the pseudo-label from a single view of a set of data points,
- *multi-view learning*, where single samples are represented in multiple ways to promote learning generalizable patterns, and
- *multi-model learning*, where multiple models are used during the learning process (e.g., in a student-teacher setting).

We outline directions for future work that lie at the intersection of these areas, and directions illuminated in one area by considering established methods in the other.

Pseudo-labels within semi-supervised learning. Pseudo-labeling was first applied to deep learning for computer vision in Lee [83].[1] Most of the applications of PL to SSL can be viewed as improvements on this paper. We can see in Figure 4.1 that

[1]There are self-training methods that predate the work of Lee [83]; e.g. Yarowsky [164], Nigam et al. [95]. Here we follow the common usage of pseudo-labeling in deep learning literature.

there are many works that build upon the ideas presented in Lee [83]. The methods in this area largely leverage three important ideas that continue to motivate research in PL for SSL: the *low density separation assumption*, the *cluster assumption*, and the *manifold assumption*. The low density separation assumption states that samples are embedded in high-density regions separated by low-density regions. The *cluster assumption* states that points within high density regions share the same class label. The *manifold assumption* states that high-dimensional data lies on a low(er)-dimensional manifold embedded within that space.

Subsequently, the literature includes the notion of *consistency regularization* (CR) and *entropy minimization* (EM) [27, 133, 12, 11, 176, 156, 94, 126, 65]. In CR, examples that have been slightly perturbed (e.g., by an augmentation or adversarial additive perturbation) should lie close together in any embedding space or output distribution space produced by the model. In EM, examples should lie in high density regions alongside those of the same class and far from dense regions containing examples from other classes. Though CR and EM form strong arguments for PL with techniques using the average of predicted probabilities for augmentations to assign pseudo-labels to achieve *sample scheduling* or *curriculum learning* (CL) and most approaches choosing to assign "hard" (one-hot) pseudo-labels to examples to encourage EM [12, 133, 11, 176, 170].

Pseudo-labels within unsupervised learning. Beyond semi-supervised learning, pseudo-labels have been applied in sub-fields of unsupervised learning. Most notably PL has been used in many successful algorithms for self-supervised learning [24, 87, 23, 32, 109, 169, 66, 8]. Knowledge distillation [62, 45, 173, 89, 153] also makes use of pseudo labels to distill the knowledge of one model into another. It is important to note that while some knowledge distillation algorithms or frameworks admit the

use of supervised labels, and thus could be considered semi-supervised, the process of distilling knowledge from one network to another does not generally require supervision. For this reason and for the purpose of this chapter we will consider these within the unsupervised category. These all fall under our definition of PL and the investigation of these methods through the lens of PL illuminates new avenues for investigation.

Pseudo-labeling techniques that are tolerant to label noise. Pseudo-labels are noisy by construction. This has led to the development of mechanisms that provide better pseudo-labels or that can tolerate incorrect pseudo-labels. The study of classifiers under label noise has been a fruitful avenue of discovery and is some of the most mature research related to PL. For example, the work of Angluin and Laird [3] on random classification noise has inspired new learning algorithms in SSL for more than two decades now; e.g., Goldman and Zhou [52] and Zhou and Li [178]. Other than our own there are also excellent surveys on the topic; e.g., Song et al. [134], Han et al. [58], Frénay and Verleysen [49]. Techniques that deal with label noise are deeply embedded into PL methods and one can find noise-tolerant mechanisms for PL approaches in all the categories shown in Figure 4.1. While the investigation of label noise is important and highly influential, it is nevertheless outside the scope of this chapter. We feel that the study of those areas is already well-served by the work cited above.

Novelty While previous work has comprehensively covered semi-supervised learning as a whole, those works devote limited space to PL. PL is also broader than SSL as we discussed above. By focusing exclusively on PL, in this chapter we will achieve greater depth than those prior work. We will provide a formal definition of PL that will allow us to unify disparate methodological threads within PL. We will establish a systematic taxonomy for PL methods. Most importantly, our contribution is distinguished through investigating PL both inside and outside the area of SSL.

Outline. The following sections of this chapter give an overview of PL approaches organized by our taxonomy (see Figure 4.1). In Section 4.2 we provide preliminaries so that we can clarify terms. Of particular importance are the notions of fuzzy sets, fuzzy partitions, and stochastic labels, all of which allow us to define the central notion of this review, that of *pseudo-labels*. Once this common language is established, we proceed in Section 4.3 with a summary of the methods that belong to semi-supervised learning. In Section 4.4 we provide a summary of unsupervised methods, including methods of self-supervision. In Section 4.5 we discuss directions for future work that are relevant to this dissertation. Finally, we conclude this chapter in Section 4.7. For a more complete treatment of the methods, which we enumerate in this chapter, please see Rothenberger et al. [2025].

4.2 Preliminaries

We will introduce an interpretation of pseudo-labeling as fuzzy partitions that provides a formal framework for understanding the methods in the remainder of the text.

4.2.1 Fuzzy Partitions

Fuzzy sets have the property that elements of the universe of discourse Ω belong to sets with some *degree of membership* that is quantified by some real number in the interval $[0, 1]$. For this reason a function $m\colon \Omega \to [0, 1]$ is used so that $m(\omega)$ expresses the *grade* (or, *extent*) to which an $\omega \in \Omega$ belongs to a particular set. The function $m = \mu_{\mathcal{A}}$ is called the *membership function* of the fuzzy set $\mathcal{A}$. Such a fuzzy set $\mathcal{A}$ is usually defined as $\mathcal{A} = (\Omega, \mu_{\mathcal{A}})$ and is denoted as $\mathcal{A} = \{\mu_{\mathcal{A}}(\omega)/\omega \mid \omega \in \Omega\}$.

Example 1 (Illustration of a Fuzzy Set)**.** Let $\mathcal{X}$ be the universe of discourse with just four elements; i.e., $\mathcal{X} = \{a, b, c, d\}$.

Then, $\mathcal{S}_1 = \{0.4/a, 0.1/b, 0.0/c, 1.0/d\}$ is a *fuzzy set* on $\mathcal{X}$.

The intention is to treat the "belongingness" numbers as a measure for the degree of membership in particular sets. For pseudo-labeling we will assume that there is a predetermined number of fuzzy sets $\mathcal{S}_1, \ldots, \mathcal{S}_k$, for some $k \geq 2$ and these will correspond to the k hard labels that are available in a classification problem (more below). Along these lines, an important notion is that of a *fuzzy partition*, which we will use to define pseudo-labeling precisely.

Definition 4.2.1 (Fuzzy Partition)**.** Let $k \geq 2$. Let $\mathcal{I} = \{1, \ldots, k\}$. Let $\mathcal{Q} = (\mathcal{S}_1, \ldots, \mathcal{S}_k)$ be a finite family where for each $i \in \mathcal{I}$, $\mathcal{S}_i$ is a fuzzy set and $\mu_{\mathcal{S}_i}$ is its corresponding membership function. Then, $\mathcal{Q}$ is a *fuzzy partition* if and only if it holds that $(\forall x \in \mathcal{X})\, [\sum_{i \in \mathcal{I}} \mu_{\mathcal{S}_i}(x) = 1]$.

Example 2 (Illustration of a Fuzzy Partition)**.** Let $\mathcal{I} = \{1, 2, 3\}$. Let $\mathcal{X} = \{a, b, c, d\}$ and consider the following fuzzy sets:

$$\begin{cases} \mathcal{S}_1 &= \{\ 0.4/a,\ 0.1/b,\ 0.0/c,\ 1.0/d\ \}, \\ \mathcal{S}_2 &= \{\ 0.3/a,\ 0.6/b,\ 0.6/c,\ 0.0/d\ \}, \text{ and} \\ \mathcal{S}_3 &= \{\ 0.3/a,\ 0.3/b,\ 0.4/c,\ 0.0/d\ \}. \end{cases}$$

The above family of fuzzy sets $\mathcal{S}_1, \mathcal{S}_2, \mathcal{S}_3$ together with the corresponding membership functions $\mu_{\mathcal{S}_1}, \mu_{\mathcal{S}_2}, \mu_{\mathcal{S}_3}$ whose values are shown in equation above, provides a fuzzy partition since, for every $x \in \mathcal{X}$, it holds that $\sum_{i \in \mathcal{I}} \mu_{\mathcal{S}_i}(x) = 1$.

More information on fuzzy sets is available at Zadeh [167] and Ruspini [122].

4.2.1.1 Deeper Discussion on Labels

Below we provide more information on labels and pseudo-labels. In addition, Δ^C will continue to be the probability simplex of dimension C.

Definition 4.2.2 (Stochastic Labels)**.** Let $|\mathcal{Y}| = C$. *Stochastic labels* are vectors $y \in \Delta^C$ where the positions of the vector defines a probability distribution over the classes $\mathcal{Y}_i \in \mathcal{Y}$, e.g., $y = (p_1, \ldots, p_C)$.

Remark 1 (Stochastic Labels from Fuzzy Partitions)**.** In Example 2, the fuzzy sets $\mathcal{S}_1$, $\mathcal{S}_2$, and $\mathcal{S}_3$ correspond to three different labels. By taking vertical cuts of these three fuzzy sets for each one of the four instances ($a, b, c, d \in \mathcal{X}$) one obtains a vector of non-negative values that sum to one. This vector could specify a valid discrete probability distribution, and such a distribution would be the stochastic label associated with the respective instance. For example, the stochastic label of instance b would be $(0.1, 0.6, 0.3)$. I make this remark to illustrate the connection between our definition of pseudo-labels and previous interpretations of pseudo-labels, which often use the language of discrete probability distributions.

Definition 4.2.3 (Pseudo-Labels)**.** Given a fuzzy partition $\mathcal{Q}$ over $\mathcal{X}$, pseudo-labels are vectors $y \in \Delta^C$ where each position of the vector defines the degree of membership in a unique fuzzy set in $\mathcal{Q}$ (corresponding to a label). For an instance $x \in \mathcal{X}$ we have $y = (\mu_{\mathcal{S}_1}(x), \ldots, \mu_{\mathcal{S}_C}(x))$. We write $y = \mathcal{Q}(x)$.

Using the notation introduced earlier, for some instance x, the pseudo-label takes the form $f_\theta(x)$. The crucial point is that a pseudo-label is obtained as a result of a predictive model $f_\theta(x)$ applied on the instance x, indicating the probability that x has belonging to different classes according to f_θ. Ideally (if we are using pseudo-labels as weak supervision), generated pseudo-labels are similar to ground-truth labels for the various instances $x \in \mathcal{X}$. Good pseudo-labels are similar to the ground truth to the extent that the ground truth is not significantly different from the labeled dataset $\mathbf{L}$ (e.g., because of noise). Because each element of the pseudo-label vector indicates membership in a particular class, the pseudo-label can also be one-hot (e.g., see instance

d in Example 2). The difference between the ground truth label and the pseudo-label in that case is that the latter was generated by the fuzzy partition using the instance.

Definition 4.2.4 (Pseudo-Labeling)**.** Utilizing pseudo-labels generated by at least one fuzzy partition $\mathcal{Q}$ of the input space $\mathcal{X}$ inferred from only instances and ground truth labels to provide supervision during the training of a machine learning algorithm is called *pseudo-labeling*.

Pseudo-labeling is the process of generating/inferring pseudo-labels. In self-training (e.g. [83]) the fuzzy class partition defined by f_θ is used to assign pseudo-labels to unseen instances. These instances are then used to optimize the network further.

Definition 4.2.5 (Pseudo-Labeled Examples)**.** Pseudo-labeled examples are tuples $(x, \mathcal{Q}(x))$.

The examples that are formed by the tuples of instances and pseudo-labels are the pseudo-examples that are used during retraining in self-training.

4.3 Semi-Supervised Regimes

The classic formulation of PL techniques as applied to deep neural networks is defined in Lee [83], where PL is presented as a fine-tuning stage in addition to normal training. At a high level, the model f_θ is initially trained over samples from the labeled set $\mathbf{L}$, and after a fixed number of epochs θ is frozen and pseudo-examples derived from f_θ and $\mathbf{U}$ are added into the training steps, yielding an effective SSL technique.

More specifically, this formulation of PL derives labels for samples drawn from $\mathbf{U}$ by "sharpening" the predictions of the partially-trained network, effectively reinforcing the partially-trained model's "best guess" for a particular sample. The loss function $\mathcal{L}$ being optimized has a supervised component ℓ_{S} and a nearly-identical unsupervised

component ℓ_{U}; in every case cross-entropy loss is used to penalize the model's predictions against, respectively, the true labels, or the pseudo-labels obtained by the sharpening process:

$$\mathcal{L} = \underset{\text{supervised}}{\mathcal{L}(f_\theta, \mathbf{L})} + \alpha(t)\mathcal{L}(f_\theta, (\underset{\text{unsupervised}}{\mathbf{U}, \operatorname{argmax}_k f_\theta(\mathbf{U})_k})). \tag{4.1}$$

Crucially, the unsupervised segment is moderated by $\alpha(t)$: an epoch-determined annealment parameter. This starts at zero and then ramps linearly up to a ceiling value. According to Lee [83], careful consideration is required in choosing the scheduling of α, as setting it too high will disturb training of labeled data and too low will not have an impact on the training process at all. This is the basic foundation upon which most fine-tuning PL methods are built.

In the remainder of this section we will provide an overview of pseudo-labeling methods in computer vision for semi-supervised learning. We will discuss methods based on scheduling unlabeled instances (Subsection 4.3.1), methods based on performing well under weak supervision (Subsection 4.3.2), methods of consistency regularization (Subsection 4.3.3), and approaches that use multiple models. The last of which is closely related to the joint optimization/selection of pseudo-labels (Subsection 4.3.4). For more details please see Rothenberger et al. [2025].

4.3.1 Sample Scheduling

An inherent prerequisite to solving the problem of generating pseudo-labels for samples is determining for which samples to generate pseudo-labels. The precise sample schedule can greatly impact the performance of the final pseudo-labeling algorithm [5].

The simplest way to choose unlabeled samples to label is to sample them randomly from the unlabeled set. Pseudo-labeling techniques are vulnerable to *confirmation bias*, where initially-wrong predictions are reinforced by the PL process [5]. One solution is

MixMatch [12], a hybrid PL technique combining several dominant methods within the SSL space with the goal of preventing overconfident predictions. MixMatch consists of several steps: a *data augmentation* step, a *label guessing* step (the PL component), and a *sample mixing* step. MixMatch has inspired several successful derivative algorithms such as *ReMixMatch* [11], *FixMatch* [133], and FlexMatch [170]. While most methods use confidence as the metric by which to select PL, *Time-Consistent Self-Supervision for Semi-Supervised Learning* (TCSSL) [176] uses a "time consistency" metric that is the exponential moving average of the divergence of the discrete output distribution of the model for each sample.

A natural solution to choosing a suitable set of examples to pseudo-label is *curriculum learning* (CL) [10]. In this paradigm, the model is first allowed to learn on "easy" instances of a class or concept and then trained on progressively "harder" instances. One such CL approach is Cascante-Bonilla et al. [2020]. Functionally, it is a self-training framework where, within each epoch, unlabeled data is sorted based on the confidence of prediction. Curriculum learning is also the motivation for FlexMatch [170].

4.3.2 Weak Supervision

Subtly different from metric-based selection of unlabeled examples, we can also imagine assigning a label based on a *weak supervision*. Weak supervision differs from the metrics in the previous section as they only provide a label rather than a continuous score.

An early approach in this space was that of Zhu and Ghahramani [179], which used a kNN to assign a label based on its geodesic distance to other samples in the embedding space. This approach was extended to deep neural networks in Iscen et al. [65]. Shi et al. [126] improves further with a loss that encourages all examples of one class assignment to be embedded far from those of other classes and examples within an assignment to be drawn together. *Iterative Label Improvement* (ILI) [56] employs a self-training

method where the model, post-initial training, may adjust any sample's label based on confidence levels, emphasizing the inherent noise in pseudo-labels. Additionally, ILI is extended through *Iterative Cross Learning* (ICL) [166], where models trained on dataset partitions assign pseudo-labels to other segments.

One strategy for assessing the value of weak supervision is to estimate the label noise of a classifier. *Uncertainty-Aware Pseudo-Label Selection* (UPS) [114] seeks to achieve good weak supervision by training a calibrated network for confidence filtering on the unlabeled set. Continuing in the theme of directly accounting for label noise, Tong Xiao et al. [142] adds a probabilistic model that predicts the clean label for an instance. More recent work, PENCIL [165] handles uncertainty by modeling the label for an image as a distribution among all possible labels and jointly updates the label distribution and network parameters during training. PENCIL extends DLDL [50] by moderating the label noise learning. Another approach in this space is *instance dependent noise* (IDN) by Wang et al. [149], which directly models label noise by relating noisy labels to instances.

An additional strategy for weak supervision selection is *Class Introspection* [69], which uses an explainability method over a classifier in order to embed the inputs into a space separated by their learned explanations.

Open-Set Recognition Open-set recognition (OSR) and pseudo-labeling, while distinct, share a connection in weakly-supervised tasks. OSR aims to identify known classes while detecting and rejecting unknowns (out-of-distribution instances) [9], a challenge that becomes more complex without strong labels. In the context of weakly-supervised OSR, pseudo-labeling can be employed to create initial, albeit noisy, labels for the known classes in unlabeled data. These pseudo-labels help define boundaries for known classes, which is crucial for OSR to distinguish them from unknowns. Thus,

pseudo-labeling provides a form of implicit supervision, enabling OSR models to learn and differentiate classes even with limited or absent explicit labels.

Approaches in this space such as SeFOSS [146], ProSub [147], and W-ODE [150] aim to learn from all data (in and out of distribution) in **U**. These methods utilize scores to determine whether or not examples are outliers in the training distribution to appropriately assess uncertainty to their pseudo-labels.

4.3.3 Consistency Regularization

There are two representative techniques that inject consistency regularization in a way that specifically addresses the case where pairs of instances that differ only slightly receive differing model predictions. These are *Unsupervised Data Augmentation* (UDA) by Xie et al. [156] and *Virtual Adversarial Training* (VAT) by Miyato et al. [94]. It is possible to apply these methods without any supervision at all, and in that case they are still pseudo-labeling, but they would be unsupervised rather than semi-supervised. UDA is even explicitly called *unsupervised* data augmentation, but because historically these two works are specifically for semi-supervised applications we include them in this section rather than the next.

In both of the above cases the pseudo-label of an augmented instance is inferred by its close proximity to the original sample. This distance is either a p-norm in the case of VAT or a semantic notion of distance as in UDA. The fuzzy partition is thus constituted of fuzzy sets that are the union of spheres of a chosen radius around points of the same class, or that are sets of augmented versions of an instance. In the case of VAT the labels that are assigned are one-hot, but this is a subset of the admissible fuzzy partitions. Adversarial training is a special case of consistency regularization in which the consistency is enforced locally for each instance. Adversarial training is an effective method of consistency regularization, but it incurs the additional cost of a

gradient update to the input sample.

4.3.4 Multi-Model Approaches

In multi-model approaches more than one network is used for training one singular model, combining the strengths of the individual component networks. Meta Pseudo Label (MPL) by Pham et al. [105], discussed in Section 4.3.4 of joint-optimization selection, is an example where a teacher model generates labels and a student model processes labeled data and jointly optimizes both.

Several multi-model approaches re-train iteratively similar to PL [83], but make use of multiple models. Noisy Student Training [157] is an effective technique in this category where iteratively larger models are iteratively self-trained. The student-teacher model is not the only multi-model architecture that relies on pseudo-labels for SSL. A representative technique in this area is *co-training* [14] in which two models are trained on disjoint feature sets that are both sufficient for classification. In co-training, pseudo-labels are iteratively incorporated into the training set just as with self-training, except labels added to one model's training set are generated by the other model. In Qiao et al. [2018] the authors present a study where two similar deep neural networks are used. One key problem in co-training is that of constructing the views when they are not available [135, 148]. It is relatively uncommon to have a dataset in which there are two independent and sufficient views for classification. Furthermore, it is possible that the views we have are independent, but not sufficient. Multiple works attempt to address the issue of insufficiency. Wang and Zhou [151] shows that this is still an acceptable setting in which to perform co-training provided that the views are "diverse." There are several works addressing the problem of the lack of independent views [177, 57]. These approaches are ensembles where each classifier issues a vote on the class of unlabeled samples, and the majority vote is assigned as the pseudo-label. In

the same vein, *Tri-Training* employs three classifiers in order to democratically select pseudo-labels for $\mathbf{U}$ after being trained on $\mathbf{L}$ [178]. While this is a simple solution, it comes with a $3\times$ increase in compute cost. An improvement is *Robust Semi-Supervised Ensemble Learning* [160], which formulates the PL process as an ensemble of low-cost supervisors whose predictions are aggregated together with a weighted SVM-based solver. *Multi-Head Co-Training* (MHCT) [30] uses a shared network block to create an initial representation that is then evaluated with multiple classification heads. When an unlabeled sample is selected, the most confident pseudo-label is assigned based on a consensus between a majority of the heads.

Joint-Optimization Selection Unlike iterative-retraining methods of pseudo-labeling, there is a group of methods that treat pseudo-label assignment as an optimization problem. *Meta Pseudo Labels* (MPL) [105] optimizes a teacher to provide good labels to unlabeled points that train a student. Meta-Semi [152] optimizes which pseudo-labels to use for training. Similarly, Bi-Level Optimization for Pseudo-Labeling Based Semi-Supervised Learning (BLOPL) [60] considers pseudo-labels as latent variables of which the learner's weights are a function. Rather than optimizing the presence or absence of pseudo-labels, BLOPL optimizes soft pseudo-labels for every unlabeled instance. These soft pseudo-labels are treated as latent variables and the model that predicts them is optimized to predict well on the labeled set. Reinforcement Learning Guided Semi-Supervised Learning [61] poses this optimization of pseudo labels as a reinforcement learning problem and learns a policy to provide good soft pseudo-labels to the learner. RLGSSL is not a method that requires two models, but we include it in this section as it is the only anomaly in that sense. *Meta Co-Training* (MCT) [117] combines the ideas of Meta Pseudo Labels [105] and Co-Training [14]. Meta Co-Training introduces a novel bi-level optimization over multiple views to determine optimal assignment of

pseudo labels for co-training.

4.4 Unsupervised and Self-Supervised Regimes

It is perhaps surprising to consider pseudo-labeling as a definition for unsupervised learning settings, where traditional, fixed class labels are not available. While pseudo-labeling is usually presented in the context of semi-supervised training regimes, there are very close analogues to PL in the domain of unsupervised learning. Within this section, we will discuss discriminative self-supervised learning. These methods operate as CR methods (Section 4.4.1), and response-based knowledge distillation (Section 4.4.2). Both of these forms of knowledge distillation methods are forms of pseudo-labeling. This section is an abbreviated version of the content in Rothenberger et al. [2025]. Please see Rothenberger et al. [2025] for more details.

4.4.1 Consistency Regularization

Unsupervised consistency regularization approaches show significant similarities with pseudo-labeling approaches as they tend to assign labels to points even if these labels have no relation to any ground truth. For example, EsViT [87], DINO [24], BYOL [55], and SwAV [23] all use some form of label assignment as a representation learning *pretext task*, i.e., a task that prepares a model for transfer learning to a downstream task [67]. These approaches all result in assigning instances to a particular cluster or class that has no defined meaning, but is rather inferred from the data in order to accomplish the minimization of the self-supervised loss. The goal of the clustering or classification is clear: to place images that share an augmentation relationship into the same cluster or class, and optionally a different class from all other images. The partition of the latent space however is inferred, and it is also fuzzy as membership probabilities may not be one-hot. Thus, these techniques clearly satisfy our definition.

4.4.2 Knowledge Distillation

Knowledge distillation [62] describes the process of transferring the knowledge of a larger network into a smaller one. The significant attention brought to this technique has yielded a variety of techniques [54]. In this section we are concerned with *response-based* knowledge distillation; i.e., techniques for knowledge distillation that transfer knowledge through pseudo-labels assigned by the larger model. For a more comprehensive treatment of knowledge distillation see Gou et al. [2021].

The formulation in Hinton et al. [62] is a response-based framework in which a pre-trained larger model transfers its knowledge to a smaller model through the use of pseudo-labels. While in their algorithm the smaller model also learns from the original training labels and is thus semi-supervised, the process of distilling the knowledge from the larger model is unsupervised.

As mentioned in Caron et al. [24], the DINO framework for self-supervised learning is interpreted as a framework for knowledge distillation between a student model and a mean teacher (a teacher model that is an average of the student model's weights). This is self-distillation with *no supervised labels*. The distillation process occurs entirely using PL. DINO can be seen as a special case of the *replace one branch* approach [45] to knowledge distillation with a slight modification of including a mean teacher. In fact, this is exactly how the smaller published DINO models are created.

In all cases a fuzzy partition parameterized by a neural network is inferred from data and available at the beginning of the distillation process. This is the teacher model, or the model that is being distilled. In the case of Hinton's original formulation this fuzzy partition clearly corresponds to division of the input space into classes informed by real labels. It is also possible to incorporate real labels in the distillation process in that formulation, however pseudo-labels are still used in the training process. In

the case of DINO and RoB the labels do not necessarily correspond to any ground truth quantity. DINO distillation and RoB also define fuzzy partitions of the input space into categories and the goal of learning in the knowledge distillation regime is the same: minimize the average cross entropy over a set of unlabeled instances and their associated pseudo-labels between the student and the teacher.

4.5 Open Directions Later Explored

In Rothenberger et al. [2025] we identified a range of similarities across categories of PL algorithms. Having identified the ways in which these different algorithms that use pseudo-labels are similar, we discussed unexplored directions. Below I include the open directions that are relevant for this dissertation and that we will build upon in later chapters. For a more complete treatment of exciting open directions in PL please see Rothenberger et al. [2025].

Self-Supervised Regularization of Semi-Supervised Learning In semi-supervised learning and knowledge distillation there is a risk of representation collapse of the learner. One recent method, *self-supervised semi-supervised learning* [11], incorporates a self-supervised loss term to prevent that representation collapse. Despite the simplicity they are able to improve on supervised baselines. Their approach and others such as Rothenberger and Diochnos [2023] show that the benefits for learning with few labels from pseudo-labeling-based self-supervised learning and pseudo-labeling-based semi-supervised learning are complementary. Combining the benefits of knowledge distillation and semi-supervised learning with the strong performance of representations learned with self-supervised loss terms is an exciting avenue of research for pseudo-labeling.

Meta Learning to Determine Pseudo-Label Assignment In Section 4.3.4 we identified several methods for semi-supervised learning that formulate pseudo-label assignment as a meta-learning problem or bi-level optimization. Several recent pseudo-labeling methods for semi-supervised learning [105, 117, 60, 61] leverage bi-level frameworks for pseudo-label assignment. RLGSSL [61] we find to be particular interesting because it interprets this bi-level optimization as reinforcement learning. This interpretation opens many new potential directions for exploration based on existing work in reinforcement learning. Developing new ways of formulating and optimizing these bi-level frameworks is an exciting direction for self-supervised learning. To our knowledge bi-level optimization has not been applied in this way to either knowledge distillation or self-supervised learning. Due to the promising results of pseudo-labeling methods based on bi-level optimizations for semi-supervised learning we believe similar algorithms can find success in the areas of knowledge distillation and self-supervised learning.

4.6 Summary

In Section 4.1 we gave an overview of theoretical motivations for pseudo-labeling. We provided a taxonomy of methods for pseudo-labeling in these areas and discussed their relation to the important motivating ideas of sample scheduling, label propagation, weak supervision, consistency regularization, multi-model learning, and knowledge distillation. This taxonomy was shown in Figure 4.1. In Section 4.2 we introduced a definition for pseudo-labels. We showed how this definition allows for a unified interpretation of methods from semi-supervised learning and self-supervised learning. In Section 4.3 we presented many different methods for semi-supervised learning that make use of pseudo-labels. In Section 4.4 we showed how pseudo-labels are used in unsupervised and self-supervised ways. In Section 4.5 we discussed open directions that were illuminated

by our definition of pseudo-labels. Finally, in Table 4.1 we provide a glossary of the different techniques that are shown in the taxonomy of Figure 4.1. Table 4.1 provides the paper that proposed each technique and the section in which they appear in this review.

We find that there are many exciting new opportunities for research. The prospect of applying dataset subsetting methods on data collected from the wild is particularly exciting, and so is the application of reinforcement learning algorithms/bi-level optimization frameworks and uncertainty quantification. We are excited to see how new applications of pseudo-labels continue to shape the computer vision landscape.

4.7 Conclusion

The key observations made in this chapter are that pseudo-labels are not exclusively used in the context of SSL, and that pseudo-labels are not just a inferior version of supervision. I showed examples from prior work such as that of Li and et al. [2022] and Duval et al. [2023], which use PL in a non-SSL setting. In these cases the pseudo-labels were not used as weak supervision, but rather they were optimized to have certain properties. Even within SSL we can see examples such as UDA [156] and VAT [94]. In UDA and VAT the pseudo-labels are used for consistency regularization within an SSL framework rather than as a proxy for human supervision. This is in contrast to methods such as PL [83] and co-training [14] where pseudo-labels are used as a proxy for supervision to iteratively re-train models. It is because pseudo-labels are model outputs that we are able to optimize them to have properties like consistency across augmentations or between a student and teacher.

Within SSL there are methods such as MCT [117], BLOPL [60] and RLGSSL [61] that make use of PL outside of just consistency regularization or as weak supervision. These methods explicitly optimize pseudo-label assignments to achieve other objectives.

In the case of MCT and BLOPL this objective is achieving more accurate classification when learning from pseudo-labels, and in the case of RLGSSL the objective is a proxy for generalization.

It is these directions that we will explore in the remaining chapters. We will see ways in which we can take advantage of the fact that pseudo-labels are model predictions to encouraged them to have certain properties. In the next chapter I will explore how my contribution MCT uses pseudo-labels to train a pair of models that collaborate to teach each other. In the following chapter I will build on this idea by improving the teacher feedback signal. Then, in the next chapter I will show how PL can be applied to XAI, which is neither SSL nor one of the non-SSL PL domains described in this chapter. In all three chapters I will explore the ways to apply PL without requiring that the pseudo-labels serve as weak supervision that approximates some notion of ground-truth well.

Table 4.1: Figure 4.1 Label Map

Label	Reference	Section
BLOPL	Heidari and Guo [60]	4.3.4
BYOL	Grill and et al. [55]	4.4.1
Class Introspection	Kage and Andreadis [69]	4.3.2
Co-training	Blum and Mitchell [14]	4.3.4
Cocommittee	Hady and Schwenker [57]	4.3.4
DINO	Caron et al. [24]	4.4.2
DL from NL	Tong Xiao et al. [142]	4.3.2
DeepCluster	Caron et al. [22]	4.4.1
Dem. Colearning	Zhou and Goldman [177]	4.3.4
EsViT	Li and et al. [87]	4.4.1
FixMatch	Sohn et al. [133]	4.3.1
FlexMatch	Zhang et al. [170]	4.3.1
IDN	Wang et al. [149]	4.3.2
ILI	Haase-Schütz et al. [56]	4.3.2
Knowledge Distillation	Hinton et al. [62]	4.4.2
LabelPropPL	Iscen et al. [65]	4.3.2
MHCT	Chen et al. [30]	4.3.4
MMF	Shi et al. [126]	4.3.2
Meta Co-training	Rothenberger and Diochnos [117]	4.3.4
MetaPL	Pham et al. [105]	4.3.4
MixMatch	Berthelot et al. [12]	4.3.1
Noisy Student	Xie et al. [157]	4.3.4
OVOD	Minderer et al. [93]	4.4.1
PENCIL	Yi et al. [165]	4.3.2
PL (2013)	Lee [83]	4.3
ProSub	Wallin et al. [147]	4.3.2
RLGSSL	Heidari et al. [61]	4.3.4
ROSSEL	Yan et al. [160]	4.3.4
ReMixMatch	Berthelot et al. [11]	4.3.1
Replace One Branch	Duval et al. [45]	4.4.2
SeFOSS	Wallin et al. [146]	4.3.2
SwAV	Caron et al. [23]	4.4.1
TCSSL	Zhou et al. [176]	4.3.1
TriTraining	Zhou and Li [178]	4.3.4
UDA	Xie et al. [156]	4.3.3
UPS	Rizve et al. [114]	4.3.2
VAT	Miyato et al. [94]	4.3.3
W-DOE	Wang et al. [150]	4.3.2

Chapter 5

Meta Co-Training

5.1 Chapter Introduction

Not unrelated to and not to be confused with the idea of ***semi****-supervised learning*, is ***self****-supervised learning* In self-supervised learning a model is trained with an objective that is evident from the data itself; it does not require human-generated labels. Self-supervised learning was popularized by the BERT model [40] for generating word embeddings for natural language processing (NLP) tasks. BERT generates its embeddings by solving the *pretext* task of masked word prediction. Pretext tasks present unsupervised objectives that are not directly related to any particular supervised objective but rather are solved in the hope of learning a suitable representation to more easily solve a downstream task. These pretext learners are often referred to as *foundation models*. Several pretext tasks have been proposed for CV to train associated foundation models that solve them [59, 32, 169, 87, 23, 24]. The learned representations for images are often much smaller than the images themselves. As a consequence, this yields the additional benefit of reduced computational cost when using the learned representations.

SSL algorithms [83, 14, 108, 157, 105, 133, 156, 12] involve generating pseudo-labels to serve as weak supervision on unlabeled data and then learning from those pseudo-labels. This weak supervision can either be used to perform *consistency regularization*, or *entropy minimization*, or both. Consistency regularization enforces that different augmented versions of the same instance are assigned the same label. Entropy minimization enforces that all instances are assigned a label with high confidence. Typically consistency regularization is achieved by minimizing a consistency loss between pairs of examples that ought to have the same label [156, 12, 133, 11], and entropy minimization

is achieved by iterative re-training [83, 14, 108] or sharpening labels [105, 12, 133, 11].

One particular class of SSL algorithms are co-training [14] algorithms in which two different "views" of the data must be obtained or constructed, and then two different models must be trained and re-trained iteratively. These two views yield models that capture different patterns in their input and thus, informally speaking, are more likely to fail independently. This independence of failure is leveraged so that each model can provide useful labels to the other. Standard benchmark problems in machine learning do not present two views of the problem to be leveraged for co-training. If one hopes to apply co-training, one needs to construct two views from the single view that they have access to. I show that constructing views that satisfy the conditions necessary for co-training is relatively simple. Training different models on the two constructed views can boost performance. Unfortunately, the classical *co-training* algorithm (CT) fails to utilize pseudo-labels effectively on benchmark tasks.

I propose a novel SSL method that more effectively leverages pseudo-labels than previous methods. I call the algorithm *Meta* Co-Training (MCT), because it leverages two views to produce good pseudo-labels as part of a bi-level optimization. In MCT each model is trained to provide better pseudo-labels to the other model, given only its view of the data and the performance of the other model on a labeled set. Similarly to CT, MCT utilizes multiple views to train models that are diverse. These models have an advantage when teaching (and learning from) each-other because they will learn different patterns from their differing input data. I show that this approach provides an improvement over both CT and the current state-of-the-art (SoTA) method [88] on the ImageNet-10% dataset, as well as it establishes new SoTA few-shot performance on several other fine-grained image classification tasks.

Chapter Contributions

1. I propose *Meta Co-Training*, an SSL algorithm that appears to be more robust than co-training and does not require iterative re-training from initialization.

2. I conduct experiments on a range of benchmark image classification tasks to evaluate the performance of MCT.

3. I demonstrate that not only is MCT effective on benchmark tasks but also that it provides significant utility on real-world datasets.

4. I make the implementation publicly available:
   `https://github.com/JayRothenberger/Meta-Co-Training`

Outline of the Chapter In Section 5.2 I discuss the construction of views using pre-trained self-supervised learning models. In Section 5.3 I describe the proposed method, Meta Co-training. In Section 5.4 I present experimental findings on ImageNet [39] and discuss how MCT compares to relevant baseline approaches. I perform additional experiments on Flowers102 [96], Food101 [16], FGVCAircraft [92], iNaturalist [63] datasets. In Section 5.4.6 I present an application of co-training algorithms to a real-world dataset. In Section 5.5 I conclude with a summary and directions for future work. The interested reader can find additional information (e.g., details on the training recipe) in Rothenberger and Diochnos [2023].

5.2 View Construction

A view, in the context of co-training methods, is a subset of the input features. A view typically has two properties: *(i)* it is sufficient for predicting the desired quantity, and *(ii)* it is conditionally independent of other views given the label. Previous methods of view construction for co-training include manual feature subsetting [42], automatic feature subsetting [31], random feature subsetting [17], random subspace selection [18],

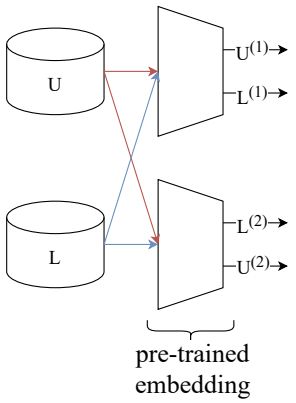


Figure 5.1: Using pre-trained models for constructing views. Unlabeled and labeled sets are embedded using two different pre-trained models. This transforms a single-view dataset into two compressed representations which are used as views for co-training and meta co-training.

and adversarial examples [108]. I used self-supervised learning to construct views for the CT and MCT experiments. However, if one *has access* to a dataset where the instances have two different views, then one can still obtain useful representations with this approach using the same self-supervised model or none at all.

While my is more widely applicable, it is particularly well-suited to computer vision. There are a plethora of competitive learned representations for images [55, 32, 23, 87, 109, 100, 59]. Choosing foundation model representations as views makes my approach lightweight enough to be feasible with limited hardware resources.

5.3 Proposed Method: Meta Co-Training

I propose a novel semi-supervised learning algorithm called *Meta Co-Training (MCT)*, which is described below. MCT operates within a bi-level student-teacher optimization framework that is inspired by Meta Pseudo Labels (MPL) [105], which I extend to two views to perform co-training [14]. A student is optimized to replicate the labels that its teacher gives to a set of unlabeled points while a teacher is optimized to provide labels to the same unlabeled points such that its student learns to predict well on labeled

data. MCT is said to be a co-training algorithm because it operates on two views of the data. However, unlike the original co-training method [14], MCT does not train multiple models for each view sequentially; CT incrementally expands the labeled set with pseudo-labels and re-trains the models for each view, whereas MCT simply learns to generate better labels without expanding the labeled set. In MCT, each view has exactly one model that acts as both a teacher for the other view and a student for its own view. That is to say that the model learning on view 1 will take an unlabeled instance from view 1 as input and provide the pseudo-label prediction to the student for the same unlabeled instance in view 2. The student, the model learning on view 2, will use the pseudo-label to learn to predict similarly.

The lower of the two levels of the optimization is the student optimization (Equation 5.2). The student parameters θ_S are optimized as a function of the teacher parameters θ_T:

$$\mathcal{L}_{student}(f_{\theta_S}; (\mathbf{U}, f_{\theta_T}(x_j))) = \frac{1}{m} \sum_{j=n+1}^{n+m} \ell(f_{\theta_S}(x_j), \mathcal{Y} \sim f_{\theta_T}(x_j)) \text{, and} \tag{5.1}$$

$$\theta'_S = \theta_S^{\mathrm{PL}}(\theta_T) \in \operatorname{argmin}_{\theta_S} \mathcal{L}_{student}(f_{\theta_S}; (\mathbf{U}, f_{\theta_T}(\mathbf{U}))) \,. \tag{5.2}$$

The teacher optimization (Equation 5.3) forms the upper level of the optimization:

$$\theta'_T \in \operatorname{argmin}_{\theta_T} \mathcal{L}\left(f_{\theta'_S}; \mathbf{L}\right) \,. \tag{5.3}$$

All together, the objective of MCT from the perspective of the current view model as the teacher is:

$$\min_{\theta_T} \mathcal{L}_{student}(f_{\theta_S}; (\mathbf{U}, f_{\theta_T}(\mathbf{U}))) + \mathcal{L}\left(f_{\theta'_S}; \mathbf{L}\right) \,. \tag{5.4}$$

It is cumbersome to speak of teachers and students when each model is both. The

objective for MCT is given in Equation 5.5 as jointly optimizing the following objectives with f_{θ_1} the model learning on view 1 and f_{θ_2} the model learning on view 2:

$$\min_{\theta_1,\theta_2} \mathcal{L}_{student}\left(f_{\theta_1}; (\mathbf{U}, f_{\theta_2}(\mathbf{U}))\right) + \mathcal{L}\left(f_{\theta_2'}; \mathbf{L}\right) + \mathcal{L}_{student}\left(f_{\theta_2}; (\mathbf{U}, f_{\theta_1}(\mathbf{U}))\right) + \mathcal{L}\left(f_{\theta_1'}; \mathbf{L}\right). \tag{5.5}$$

Co-training algorithms are traditionally evaluated using the joint prediction of the constituent models; i.e., the re-normalized element-wise product of the individual predictions.

An overview of MCT is provided in Figure 5.2. At each step $t \in \{1, \ldots, T\}$ of MCT the models that correspond to the so-far learnt parameters $\theta_{1;t}$ and $\theta_{2;t}$ participate in a student-teacher framework in which each model plays the role of the student for their view and the teacher for the other model *simultaneously*. The representations obtained from the view construction process of Section 5.2 form different views that are used by MCT.

By training two models on two views to collaborate MCT can take advantage of the aspects of MPL and CT that make each algorithm successful. CT is more effective when the two constituent models fail (predict incorrectly) independently. During MCT the teacher model will receive positive feedback precisely when it makes a prediction that the other model would not have made but that results in an improvement. MCT trains models in a way that encourages each teacher to express independence while it teaches each student to incorporate the improvement. Naturally, once the two models predict similarly then the gradient of the student parameters with respect to the teacher's labels is zero and they both cease to improve. This, however, is also when MPL ceases to improve and it is unlikely to have models that are as diverse as those trained from independent views.

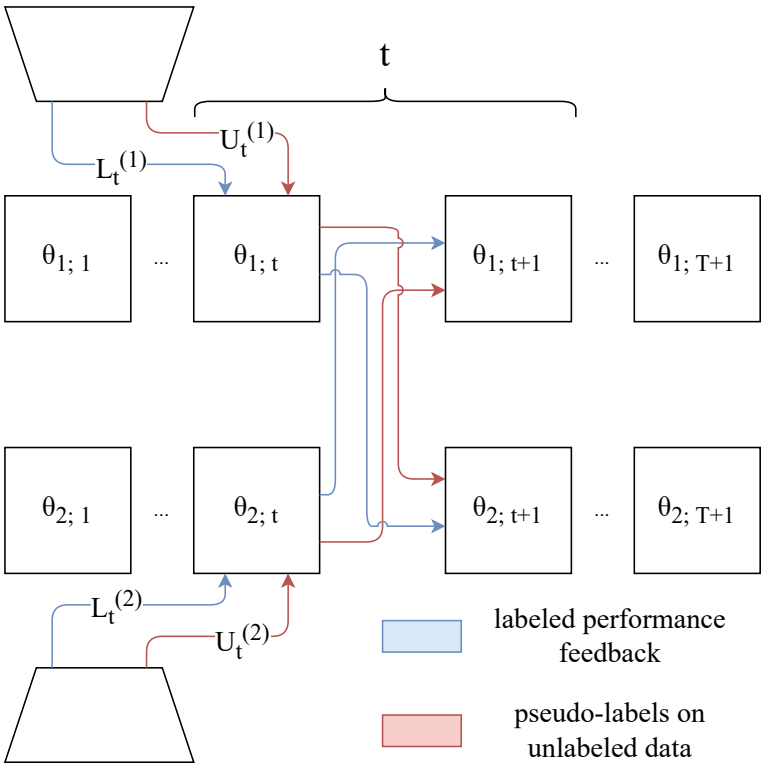


Figure 5.2: At each step $t \in \{1, \ldots, T\}$ of MCT the models that correspond to the so-far learnt parameters $\theta_{1;t}$ and $\theta_{2;t}$ play the role of the student and the teacher simultaneously using batches for their respective views. Pseudo-labeling occurs on complementary views so that the teacher can provide the student with labels on an unlabeled batch. Labeled batches may, or may not, use complementary views as the purpose that they serve is to calculate the risk of the student model on the labeled batch and this result signals the teacher model to update its weights accordingly.

Algorithm Description Algorithm 1 presents pseudocode for the algorithm. The function *SampleBatch*, is sampling a batch from the dataset. The function *getOtherView* returns the complementary view of the first argument; the second argument clarifies which view should be returned ("2" in Line 5). Instances for co-training methods can be viewed as elements of a bipartite graph. Each edge and its two endpoints form an instance. In Line 4 instances are sampled from one side of the bipartite graph and the function *getOtherView* returns the connected elements from the second view.

At each step t of the algorithm each model is first updated based on the pseudo-

Algorithm 1 Meta Co-Training

1: Input: $L^{(1)}, L^{(2)}, U^{(1)}, U^{(2)}, f^{(1)}, f^{(2)}, \theta_1, \theta_2, T, \ell, \eta$
2: **for** step $t \in \{1 \dots T\}$ **do**
3: {Unlabeled views must be complementary}
4: $U_t^{(1)} \leftarrow SampleBatch(U^{(1)})$
5: $U_t^{(2)} \leftarrow getOtherView(U_t^{(1)}, 2)$
6: {Predict soft pseudo-labels}
7: $\hat{y}^{(1)} \leftarrow f_{\theta_1}^{(1)}(U_t^{(1)})$
8: $\hat{y}^{(2)} \leftarrow f_{\theta_2}^{(2)}(U_t^{(2)})$
9: {MCT student update applied to both models}
10: $\theta_1' \leftarrow \theta_1 - \eta \cdot \nabla_{\theta_1} \mathcal{L}_{student}(f_{\theta_1}; (U_t^{(1)}, \hat{y}^{(2)}))$
11: $\theta_2' \leftarrow \theta_2 - \eta \cdot \nabla_{\theta_2} \mathcal{L}_{student}(f_{\theta_2}; (U_t^{(2)}, \hat{y}^{(1)}))$
12: {Sample batches from L for the teacher updates}
13: $X_t^{(1)}, Y_t^{(1)} \leftarrow SampleBatch(L^{(1)})$
14: $X_t^{(2)}, Y_t^{(2)} \leftarrow SampleBatch(L^{(2)})$
15: {MCT teacher update applied to both models}
16: $\hat{y}'^{(1)} \leftarrow f_{\theta_1'}^{(1)}(X_t^{(1)})$
17: $\hat{y}'^{(2)} \leftarrow f_{\theta_2'}^{(2)}(X_t^{(2)})$
18: $h^{(1)} \leftarrow \nabla_{\theta_2'} \mathcal{L}(f_{\theta_2'}; \mathbf{L})^T \cdot \nabla_{\theta_2} \mathcal{L}_{student}(f_{\theta_2}; (U_t^{(2)}, \hat{y}^{(1)}))$
19: $h^{(2)} \leftarrow \nabla_{\theta_1'} \mathcal{L}(f_{\theta_1'}; \mathbf{L})^T \cdot \nabla_{\theta_1} \mathcal{L}_{student}(f_{\theta_1}; (U_t^{(1)}, \hat{y}^{(2)}))$
20: {Weights to be used in the next step}
21: $\theta_1 \leftarrow \theta_1' - \eta \cdot h^{(1)} \cdot \nabla_{\theta_1} \mathcal{L}_{student}(f_{\theta_1}; (U_t^{(2)}, \hat{y}^{(2)}))$
22: $\theta_2 \leftarrow \theta_2' - \eta \cdot h^{(2)} \cdot \nabla_{\theta_2} \mathcal{L}_{student}(f_{\theta_2}; (U_t^{(2)}, \hat{y}^{(2)}))$
23: **end for**

labels the other has provided on a batch of unlabeled data (Lines 13-14). Then each model is updated to provide labels that encourage the other to predict more correctly on the labeled set based on the performance of the other (Lines 24-25). Lines 13-14 correspond to the student updates, while Lines 24-25 correspond to the teacher updates. These updates make tractable the optimization of Equation 5.5 since the student and teacher optimizations are approximated using alternating iterations of stochastic gradient descent. This is following the ideas of MPL and further discussion is available in [105, 48, 117].

Each student model evaluates the performance of their new weights on separate

labeled batches. This performance provides feedback to the teacher model. Geometrically, this can be understood as updating the parameters in the direction of increasing the teacher's confidence in the pseudo-labels, assuming those pseudo-labels helped the student model perform better on the labeled set. Along these lines, the teacher parameters are updated in the opposite direction if it hurt the student's performance on the labeled set.

In general, MCT does not require any preexisting method of view construction to be applied. If a dataset naturally provides two views, then there is no need to construct additional views. One can use any pre-trained representation or none at all and then apply MCT as described above. As an additional remark, for MCT and CT the labeled datasets for each view do *not* need to have the same bipartite correspondence as the unlabeled datasets. This is because the labeled data used to evaluate the change in the performance of the student do not need to have corresponding instances in the teacher's view. In all of the experiments in this work preexisting foundation models were used to construct views for the co-training methods, and thus in all cases there exist two views for each labeled example.

In Algorithm 1 it is illustrative to consider all versions of the parameters within a step and all of their gradients to exist simultaneously in memory. This is undesirable in practice given the large size of common neural networks and their gradients. Instead, we compute $\nabla_{\theta_2}\mathcal{L}_{student}(f_{\theta_2};(U_t^{(2)},\hat{y}^{(2)}))$ and $\nabla_{\theta_1}\mathcal{L}_{student}(f_{\theta_1};(U_t^{(1)},\hat{y}^{(1)}))$ (used in Lines 21-22) first, then we compute (and apply) $\nabla_{\theta_1}\mathcal{L}_{student}(f_{\theta_1};(U_t^{(1)},\hat{y}^{(2)}))$ and $\nabla_{\theta_2}\mathcal{L}_{student}(f_{\theta_2};(U_t^{(2)},\hat{y}^{(1)}))$ (used first in Lines 10-11 and reused in Lines 18-19), then we compute $\nabla_{\theta_1'}\mathcal{L}(f_{\theta_1'};\mathbf{L})^T$ and $\nabla_{\theta_2'}\mathcal{L}(f_{\theta_2'};\mathbf{L})^T$ that we subsequently use to compute $h^{(1)}$ and $h^{(2)}$. We finish the iteration with Lines 24-25 using the saved gradient from the first step. With this approach we only have to compute the forward pass and gradient for the student update once, and we only keep one version of the model weights in

memory at a time.

Computing necessary gradients to compute $h^{(1)}$ and $h^{(2)}$ can be entirely avoided by approximating them as:

$$h^{(1)} \approx \mathcal{L}(f_{\theta_1}; \mathbf{L}) - \mathcal{L}(f_{\theta'_1}; \mathbf{L}), \text{ and}$$
$$h^{(2)} \approx \mathcal{L}(f_{\theta_2}; \mathbf{L}) - \mathcal{L}(f_{\theta'_2}; \mathbf{L}).$$

If the approximation is used, then MCT is no more space-intensive than co-training, though it requires twice as many backward passes and three times as many forward passes. This approximation is suggested by the authors of [105]. I do not use it in my experiments.

Relation to Ensemble Methods As this approach utilizes multiple models that collaborate during prediction, I acknowledge that it bears similarity to the family of ensemble methods. Typically, co-trained models are evaluated against other SSL approaches. In my evaluation I compare my semi-supervised method to supervised deep ensembles to illustrate the impact of MCT versus ensemble methods. I will show that when MCT works well, the performance benefit cannot be attributed to simply utilizing multiple models.

5.4 Experiments

First, I will present an exploratory analysis of views constructed using foundation models. Then, using those views I will compare MCT to baselines across a range of benchmark datasets.

5.4.1 View Construction Experiments

To produce the views used to train the classifiers during CT and MCT the embedding spaces of five representation learning architectures: the Masked Autoencoder (MAE) [59], DINOv2 [100], SwAV [23], EsViT [87], and CLIP [109] were used. The models that produce the views were selected because they have been learned in an unsupervised way, have been made available by the authors of their respective papers for use in PyTorch, and have been shown to produce representations that are appropriate for computer vision classification tasks. As discussed in Section 5.2 under the paragraph "View Construction," CT relies on two assumptions about the sufficiency and independence of the its views. Experiments were conducted to verify these assumptions. Below is given a summary of the results of those experiments.

On the Sufficiency of the Views Sufficiency is fairly easy to verify. If a reasonable sufficiency threshold is chosen for the task (e.g., close or above 75% top-1 accuracy for ImageNet), then simple models can be trained on the views of the dataset that have been constructed. These simple models can serve as examples of functions that demonstrate the satisfaction of the sufficiency property. A single linear layer with softmax output was tested to provide a lower bound on the sufficiency of the views on the standard subsets of the ImageNet labels for semi-supervised classification (Table 5.3). A simple 3-layer multi-layer perceptron (MLP) with 1024 neurons per layer was also evaluated (Table 5.1). These MLPs were the same models as those used for CT and MCT. From these experiments, whose results are shown in Table 5.1 we can see that if we were to pick a threshold for top-1 accuracy around 75%, then CLIP and DINOv2 provide sufficient views for both subsets the ImageNet 1% and 10% subsets.

Table 5.1: MLP performance on individual views. Top-1 accuracy on different subsets of the ImageNet data are shown.

Model	1%	10%
MAE	23.4	48.5
DINOv2	78.4	82.7
SwAV	12.5	32.6
EsViT	71.3	75.8
CLIP	75.2	80.9

Table 5.2: Pairwise translation performance of linear probe on the ImageNet dataset. A linear classifier is trained on the output of an MLP that is trained to predict one view (columns) from another view (rows) by minimizing MSE. The top-1 accuracy (%) of the linear classifier is reported. Both the MLP and the linear classifier have access to the entire embedded ImageNet training set.

	MAE	DINOv2	SwAV	EsViT	CLIP
MAE	-	0.139	0.142	0.137	0.129
DINOv2	0.110	-	0.132	0.135	0.130
SwAV	0.116	0.147	-	0.137	0.126
EsViT	0.112	0.140	0.116	-	0.135
CLIP	0.110	0.146	0.136	0.156	-

On the Independence of the Views To test the independence of the views generated by the representation learning models, an identical MLP architecture was trained to predict each view from each other view; Table 5.2 presents our findings, which are explain below. Given as input the view to predict from (rows of Table 5.2) the MLPs were trained to reduce the mean squared error (MSE) between their output and the view to be predicted (columns of Table 5.2). I linear classifier was then trained on the outputs generated by the MLP given every training set embedding from each view. Had the model faithfully reconstructed the output view given only the input view then we would expect the linear classifier to perform similarly to the last column of Table 5.3. Intuitively this is because if a model is able to reconstruct one view from another exactly then the linear classifier should have the same performance on the

Table 5.3: Linear probe evaluation for views. Top-1 accuracy on different subsets of the ImageNet data are shown.

Model	1%	10%	100%
MAE	1.9	3.2	73.5
DINOv2	78.1	82.9	86.3
SwAV	12.1	41.1	77.9
EsViT	69.1	74.4	81.3
CLIP	74.1	80.9	85.4

reconstructed view as the original view. The linear classifiers never did much better than a random guess on the ImageNet class achieving at most 0.156% accuracy. Thus, while it cannot immediately be concluded that the views are independent, it is clearly not trivial to predict one view given any other. The views are certainly not entirely independent, otherwise we would expect to see half of the values in Table 5.2 below the trivial accuracy of 0.1%. However, all of the accuracies are very close to the trivial accuracy. I believe that this is compelling evidence for a high degree of independence between different representations.

Having constructed at least two strong views of the data, and suspecting that these views are independent there is compelling evidence that MCT and CT will work well on this data.

5.4.2 Co-Training Baseline Experiments

At each iteration of CT the models made predictions for all instances in U. Pseudo-labels were assigned to the 10% of the most confident predictions in U for both models. When confident predictions conflicted on examples, those instances were returned to the unlabeled set, otherwise their complementary views were entered into the labeled set of the other model with the assigned pseudo-label.

For CT, joint predictions yielded better accuracy than the predictions of each

individual model. In the ImageNet experiments, as CT iterations proceeded, top-1 accuracy always decreased. Later it is shown that this was not always the case (see Figure 5.5). The decrease was less pronounced when the views perform at a similar level; see Figure 5.3. In contrast MCT does not exhibit performance degradation after warmup. Figures 5.3 and 5.4 show results on ImageNet 10%. While the CT performance decreases after the supervised warmup, MCT performance increases.

Despite the poor performance of the pseudo-labeling during CT, using the two best performing views CT *performed as well as the previous SoTA method* for SSL classification on the ImageNet-10%; see Co-Training in Table 5.4. Unfortunately, CT was unable to leverage pseudo-labels to improve the accuracy on this task. Given the difficulty of translating between the views, and the fact that the accuracy yielded by the joint prediction is greater than its constituent views, the poor performance of the pseudo-labeling step in CT could be due to imbalanced information content between views rather than view-dependence.

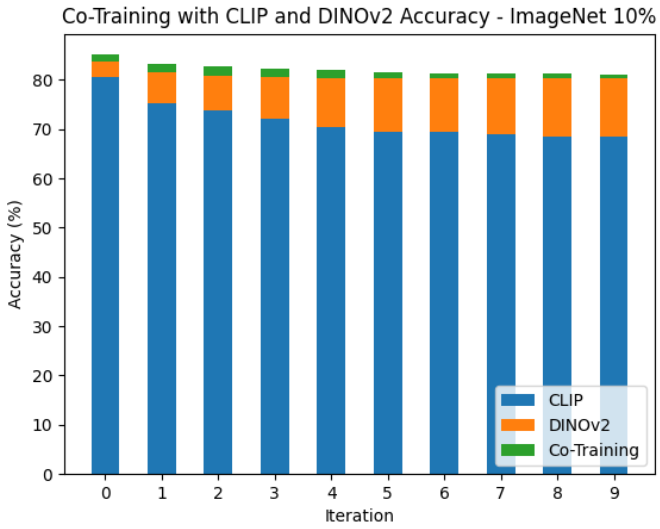


Figure 5.3: Top-1 accuracy of CT iterations on the CLIP and DINOv2 views for a training run on the ImageNet 10% dataset.

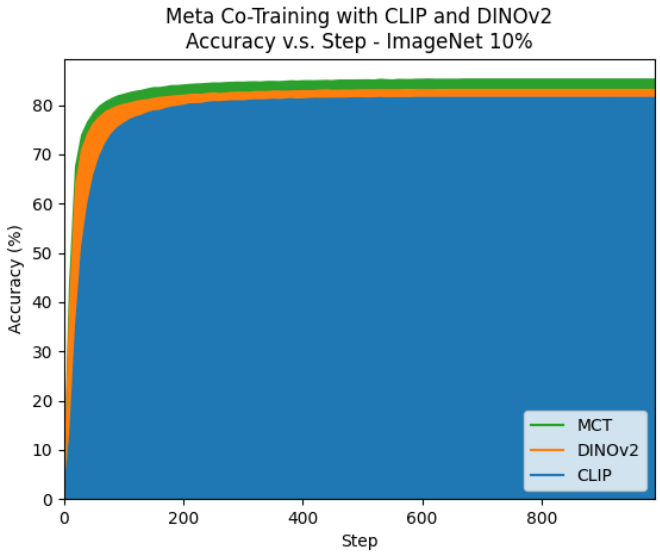


Figure 5.4: Progression of a training run for MCT using the CLIP and DINOv2 views as a function of the training step. Models are trained on 10% of the ImageNet labels.

5.4.3 Meta Co-Training Experiments

To evaluate the performance of the MCT 5-fold cross-validation experiments were performed on the ImageNet [39], Food101 [16], iNaturalist 2021 [63], Flowers102 [96], and FGVCAircraft [92] datasets. One fold was reserved for validation, which is used for model selection, and another was reserved for testing. The remaining three folds are used for the training and unlabeled sets. Accuracy reported in this section is testing accuracy. Cross-validation is used to create a sample of model performances for MCT as well as the CT and Deep Ensemble (DE) [82] baselines. To test the hypothesis that MCT produces more accurate models than the baselines I conduct multiple hypothesis tests for each dataset and label budget. These are one-tailed Welch's t-tests as the variances of the tested populations are not always equal. The null hypothesis is that the mean accuracy of models produced by the baseline being compared to is greater than or equal to the mean accuracy of those produced by MCT. In each case I am

Table 5.4: Performance of different self-supervised and semi-supervised algorithms on ImageNet dataset. An asterisk (*) indicates models that were trained on top of pre-trained frozen backbone models. For all cross-validation experiments the mean testing accuracy as well as the 95% confidence interval are given. The implementation for REACT is not publicly available, so the reported performance is assumed to be the mean. Results that allow rejection of the null hypothesis for both baselines are shown in **bold**. In Chapter 6 I will present PGPL and compare it to MCT as well as the others presented here. PGPL will improve further on the results presented in this chapter.

Reference	Model	Method	ImageNet-1%	ImageNet-10%
[87]	EsViT (Swin-B, W=14)	Linear	69.1	74.4
[59]	MAE (ViT-L)	MLP	23.4	48.5
[109]	CLIP (ViT-L)	Fine-tuned	80.5	84.7
[100]	DINOv2 (ViT-L)	Linear	78.1	82.9
[33]	SimCLRv2 (ResNet152-w2)	Fine-tuned	74.2	79.4
[23]	SwAV (RN50-w4)	Fine-tuned	53.9	70.2
[108]	Deep Co-Training (ResNet-18)	Co-Training	-	53.5
[156]	UDA (ResNet50)	UDA	-	68.78
[133]	FixMatch (ResNet-50)	FixMatch	-	71.46
[105]	MPL (EfficientNet-B6-Wide)	MPL	-	73.9
[21]	Semi-ViT (ViT-L)	Self-trained	77.3	83.3
[88]	REACT (ViT-L)	REACT	81.6	85.1
[14]	Co-Training (MLP)*	Co-Training	82.48 $\pm$ 0.13	86.03 $\pm$ 0.08
[82]	Deep Ensemble *	Deep Ensemble	78.38 $\pm$ 0.33	85.75 $\pm$ 1.20
	Meta Co-Training (MLP)* - ours	Meta Co-Training	**82.66 $\pm$ 0.19**	**86.82 $\pm$ 0.10**

performing multiple comparisons, so I apply the Bonferonni correction to the typical $\alpha = 0.05$ significance threshold. In the cases in which two comparisons are made this results in the threshold $\alpha = 0.025$, in the cases in which three comparisons are being made the threshold becomes $\alpha = 0.01\overline{6}$.

To draw fair comparisons, the model architecture and view set were fixed from the experiments on CT. As in MPL [105] a supervised loss was optimized jointly with a loss on pseudo-labels. I found that beginning with a *warmup* period in which the models for each view were trained in a strictly supervised way expedited training. This warmup period occurred for the same number of updates as the first training phase of CT, so each method started with the same number of supervised updates before pseudo-labeling.

Details on the hyperparameters are given in Rothenberger and Diochnos [2023].

Table 5.5 shows the results for MCT on all of the possible combinations of views used. With one exception, the better the performance of the views, the better MCT performed compared to CT. In nearly all cases in which both views were strong, MCT performed better than CT. These were also the cases with the highest performance overall. Comparing CT accuracy on ImageNet 10% (resp. 1%) shown in Table 5.5a compared to MLP accuracy shown in Table 5.1, CT top-1 accuracy is on average 18.7% (resp. 16.8%) higher than MLP accuracy. The best CT top-1 accuracy was 2.4% (resp. 1.7%) higher compared to the best top-1 MLP accuracy.

Finally, I took the four views that had the greatest performance and constructed two views out of them by concatenating them together. One view was constructed as CLIP | EsViT and the second DINOv2 | SwAV. These were the views used to achieve the results presented in Table 5.4.

MCT and CT are compared on ImageNet 10% (resp., 1%) in Table 5.4. In both datasets the null hypothesis can be rejected for all three baselines (CT, DE, and REACT). The margin of improvement for MCT is small on ImageNet-1%, and although significant is likely not very meaningful. The margin of improvement for MCT on ImageNet-10% is closer to 1%, which represents real progress on this challenging task.

5.4.4 Comparison to Ensemble Methods

Ensemble methods are common in practice to boost the performance of supervised models. To show that MCT provided benefit outside of just adding an additional model, MCT was compared to a DE of five models for each individual view. In order to ensure that a competitive comparison was provided a preliminary investigation into the performance of ensembles in various configurations was conducted. In Table 5.6 is shown the performance of ensembles trained on the individual views. Table 5.6

Table 5.5: CT, MCT, and ensemble top-1 accuracy of view combinations on 10% (1%) of ImageNet labels. As CT, MCT and the MLP ensemble do not depend on the order of the views, only all unique combinations are shown. In the case of the MLP ensemble these views are concatenated in order to form a larger unified view.

(a) Co-Training

	DINOv2	SwAV	EsViT	CLIP
MAE	81.8 (75.5)	30.5 (11.4)	77.1 (66.2)	78.8 (66.8)
DINOv2		78.5 (74.5)	83.3 (78.9)	**85.1** (**80.1**)
SwAV			70.6 (64.9)	75.5 (67.4)
EsViT				82.3 (76.9)

(b) Meta Co-Training

	DINOv2	SwAV	EsViT	CLIP
MAE	81.2 (73.8)	37.6 (8.1)	73.6 (63.8)	78.6 (63.5)
DINOv2		77.3 (70.7)	83.4 (79.2)	**85.2** (**80.7**)
SwAV			74.4 (67.3)	77.1 (67.4)
EsViT				82.4 (77.5)

(c) MLP Ensemble

	DINOv2	SwAV	EsViT	CLIP
MAE	83.0 (78.9)	38.1 (17.9)	75.0 (72.1)	79.4 (76.1)
DINOv2		83.7 (79.1)	81.2 (78.5)	**84.2** (**80.0**)
SwAV			74.4 (72.5)	81.2 (76.8)
EsViT				78.7 (73.5)

shows the best individual view ensemble across these experiments is produced by the DINOv2 view. A comparison of ensembles trained on the concatenation of each view pair is shown in Table 5.5c. In this second set of experiments it can be observed that view concatenation yielded ensembles that were able to achieve higher accuracy than any individual view. Motivated by the two previous results, in Table 5.7 is shown a comparison of the concatenated-view CT and MCT results with a DE trained on the concatenation of the best four individual views. The results shown in this table use

the same experimental setup and hypothesis testing described above. The experiments presented in this table allow us to reject the null hypothesis that DE produces models whose mean accuracy is greater than or equal to that of MCT on the ImageNet-1% and ImageNet-10% datasets.

Table 5.6: MLP ensemble performance on individual views. Top-1 accuracy on different subsets of the ImageNet data are shown.

Model	ImageNet-1%	ImageNet-10%
MAE	1.7	3.8
DINOv2	79.2	82.8
SwAV	19.7	40.3
EsViT	72.1	75.0
CLIP	76.4	80.8

Table 5.7: Co-training and MCT evaluated on the ImageNet dataset using the CLIP | EsViT and DINOv2 | SwAV views. The Deep Ensemble is evaluated on the concatenation of all four views. All three methods are evaluated using 5-fold cross-validation. For all cross-validation experiments the mean testing accuracy as well as the 95% confidence interval are given. Results that allow rejection of the null hypothesis for both baselines are shown in **bold**.

Method	1%	10%
Deep Ensemble	82.28 ± 0.33	85.75 ± 1.20
CT	82.48 ± 0.13	86.03 ± 0.08
MCT	**82.66 ± 0.19**	**86.82 ± 0.10**

5.4.5 Experiments on Additional Computer Vision Benchmarks

To broaden the evaluation beyond ImageNet MCT was compared against CT and existing SoTA approaches that leverage unlabeled data directly during training (e.g., not just part of a pretext task) on additional benchmark tasks. In these experiments again the hypothesis that MCT produces models with a higher mean accuracy over the folds of cross-validation than the baselines was tested. Again it was assumed for REACT

and Semi-ViT that the reported accuracy was the population mean. Cross-validation results for CT and MCT are presented in Tables 5.8 and 5.9. Figure 5.5 provides an example where CT performed as expected and accuracy increased across CT iterations. In all but two cases both null hypotheses can be rejected. These experiments provide further evidence that the primary cause of the failure of CT is that one or more of the views is sufficient to learn a model that achieves high accuracy. The closer to equivalent in performance individual views were, the less performance suffered. When the two views perform similarly and well, the performance improved. The main takeaway from these additional experiments, MCT was in general more robust to view imbalance than CT and in almost all cases was significantly better. The performance degradation of co-training beyond the initial warmup period was unexpected. In Rothenberger and Diochnos [2023] full details on CT results including charts similar to Figure 5.5 are provided. Those figures instead exhibit the described degradation.

Table 5.8: Additional comparisons to REACT. The authors include experiments for zero-shot performance and 10% of available labels. Results shown are for 10% of labels. For Flowers102 this is 1-shot performance. For all cross-validation experiments the mean testing accuracy as well as the 95% confidence interval are given. Results that allow rejection of the null hypothesis for both baselines are shown in **bold**.

	Dataset		
Method	Flowers102	Food101	FGVCAircraft
REACT (ViT-L)	97.0	85.6	57.1
Co-Training (MLP)	97.2 ± 2.3	91.94 ± 0.29	44.20 ± 1.84
Meta Co-Training (MLP)	**99.11 ± 0.34**	**92.69 ± 0.15**	46.71 ± 1.44

5.4.6 Application of Co-Training Methods to Road Condition Classification

Determining whether or not weather conditions are impacting the road surface is an important part of assessing road safety. With the increasing availability of real-time

Table 5.9: Additional comparisons to Semi-ViT using 10% (1%) of the available labels for training. For the iNaturalist only the 1010 most frequent classes were used. For all cross-validation experiments the mean testing accuracy as well as the 95% confidence interval are given. Results that allow rejection of the null hypothesis for both baselines are shown in **bold**.

	Dataset	
Method	iNaturalist	Food101
Semi-ViT (ViT-B)	67.7 (32.3)	84.5 (60.9)
Co-Training (MLP)	73.21 ± 0.38 (32.49 ± 0.78)	91.94 ± 0.29 (87.38 ± 0.40)
Meta Co-Training (MLP)	73.52 ± 0.34 (**39.57 ± 0.53**)	**92.69 ± 0.15** (**91.60 ± 0.67**)

data for this domain, it is rapidly becoming infeasible for humans to monitor all sources at all times. Having the ability to classify conditions on the surface of roads can help local authorities issue warnings or directives regarding roads that should be avoided, preferred, or if civilians should be advised not to drive at all. Using a mechanism that relies on images that are captured in an automated way at periodic intervals is particularly promising given the success of machine learning methods in *computer vision* (CV). Because of the potential impact of such a solution, prior work has investigated several methods of classifying road surface condition [158, 71, 171, 136, 161, 86, 85, 84, 99, 107, 72, 101, 102, 155, 163, 159, 68, 26, 25, 130, 73]. Most existing approaches rely on training Convolutional Neural Networks (CNNs) or Vision Transformers (ViTs) in a supervised way and require a large amount of labeled data to achieve satisfactory performance. Relying on large amounts of labeled data is problematic because it introduces a large barrier of time and cost to deploying these solutions across many camera locations. Typically models trained with data from one location or camera generalize poorly to new locations or cameras. Real deployment scenarios must allow for the addition of new camera locations and routine replacement of cameras. In this study I investigate the ability of semi-supervised learning to reduce the barriers to deployment by reducing the amount of human effort required for data collection and

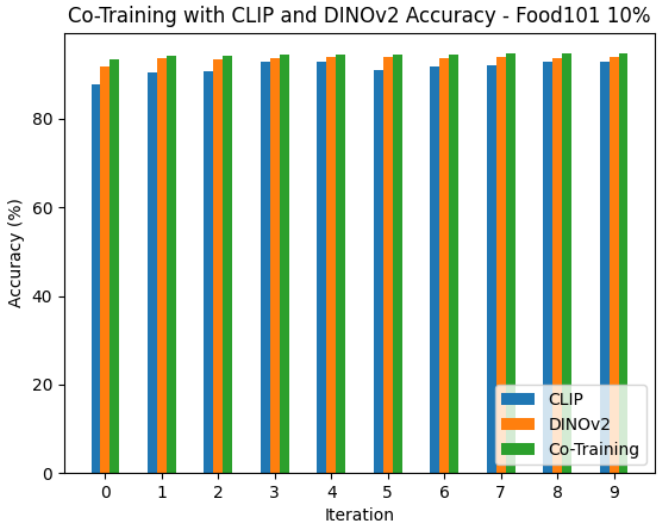


Figure 5.5: The top-1 accuracy of the CT predictions for each iteration of CT. 10% of available Food101 labels were used for training. The method exhibited performance improvement over multiple iterations of pseudo-labeling and retraining.

annotation that would otherwise be needed to label images from numerous locations.

Road Surface Classification Practitioners collect and hand-label data to train and evaluate machine learning models that learn to classify the road surface. These models can be used to inform transportation workers, and potentially the public, in real-time of dangerous conditions on the road's surface. Model generalization is usually poor between cameras due to a diversity of viewing angles, lighting conditions, and other aspects of a camera's view of the road. As a result, this data must be collected and manually labeled by experts for each location at which the road surface condition will be predicted. Reducing the number of labels per site required to train an effective model is valuable because it reduces the effort required to add new camera locations. Fortunately, it is inexpensive to collect unlabeled instances from each camera. Furthermore, cameras are often more or less co-located making this an ideal domain in which to apply multi-view



(a) Dry (b) Snow (c) Wet



(d) Dry (e) Snow (f) Wet

Figure 5.6: Three New York State Department of Transportation (*NYSDOT*) images labeled by road surface condition with both of their corresponding views. The images on the first row are from the camera site whose location in Figure 5.7 is marked with a heart. Each of the classes in the labeled data are represented for each site. The images along the second row are from the closest camera location to that site. All labeled images in the dataset are classified as having a (a, d) *dry*, (b, e) *snowy*, or (c, f) *wet* road condition.

semi-supervised learning.

A large database of images provided by the New York State Department of Transportation (NYSDOT, *https://www.dot.ny.gov/index*) from the state of New York in the United States of America is used. Images from these traffic cameras are publicly available for live viewing at `https://511ny.org/`. Images are captured from the camera sites throughout the day in five minute intervals. For this study the database of images used contains over 100 million images. I automate the task of road surface condition classification with a semi-supervised classifier. The learning problem involves a simple three-class classification over the images. A model seeks to classify these images by the corresponding road surface condition present. As seen in Figure 5.6, the images are classified as having a *Dry*, *Snow*, or *Wet* road surface condition. Through semi-supervised learning unlabeled data are utilized in three ways: (i) unlabeled obser-

vations whose label can be inferred with high probability due to co-location in space and time to labeled observations are leveraged as multiple views, (ii) unlabeled data points whose label can be inferred with learned models are used for pseudo-labeling, and (iii) a similar distribution of instances is leveraged through self-supervised learning to build a useful representation of the data points for learning. Each of the three methods help to build inductive biases that allow for learning performant models with few labels.

To take advantage of the data setting I leverage methods of *co-training* that are designed to take advantage of two representations of the input signal. The two representations used are co-located traffic cameras that view nearby parts of the road surface. I investigate two co-training algorithms for image classification: Co-Training (CT) [14] and Meta Co-Training (MCT) [117]. The former is the original method of co-training proposed by Blum and Mitchell, and the later is the state-of-the-art co-training method for image classification. Benchmark tasks in machine learning rarely present the independent views for learning that co-training algorithms require. Typically co-training algorithms utilize two representations of a single input [108, 117]. My investigation reveals these algorithms, which have been successful on benchmark tasks, perform well on real-world scenarios in which two views are available.

5.4.6.1 Data

The data is comprised of pairs of images taken from NYSDOT cameras that are less than a mile apart in Yonkers, New York. A map of the sites in this area is given in Figure 5.7. Images are captured less than 2.5 minutes apart at 5 minute intervals. From `https://511ny.org/` 192,006 instances have been collected and 4,303 have been labeled by expert human supervision [138]. For the co-located site, I assume that the labels for the instances collected at approximately the same time are the same. Although this may not always be true, I assume that it was true enough for co-training

methods to provide improvement. The remaining 187,703 instances for each view were unlabeled. I split the 4,303 examples into five folds of which one fold was reserved for testing.

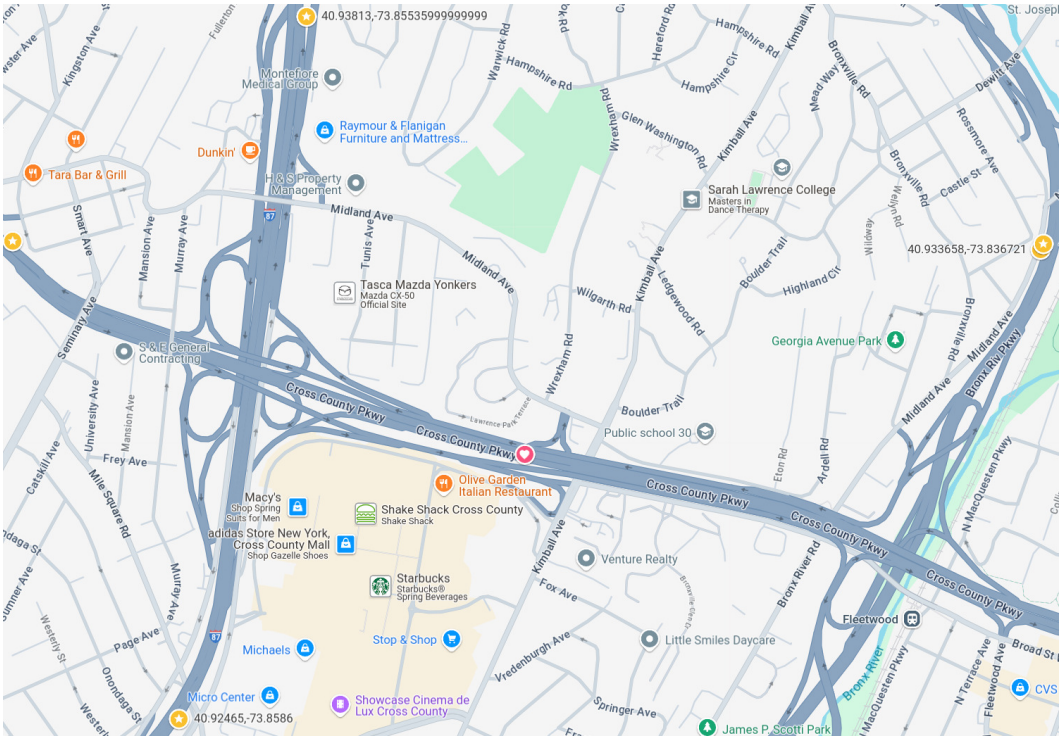


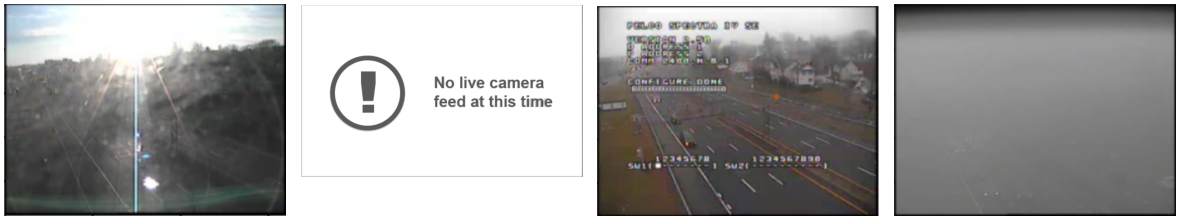
Figure 5.7: The camera site whose images have been labeled by human experts is shown with a heart near the center of the figure. The yellow stars along the perimeter of the figure depict the five closest sites to the camera with expert labeled images. Note that there are two camera sites very close to each other on the right-hand side of the figure. In this particular configuration of camera sites the starred site that is closest to the heart site, is the one that is shown at the top part of the figure. The distance between these two sites is slightly less than 0.6 miles. The ground truth labels for this site are assumed to be similar to those at the heart site since it is unlikely that road conditions due to weather are hyper-local events. Image from Google Maps [53].

The remaining four folds were divided into 20 *shards* of approximately equal size and class distribution. Of each shard 75% of the shard is allocated for validation, and 25% is allocated for training. This split is atypical for normal supervised learning problems, but at very low label counts it was harder to assess which version of the model is the best performing over the course of training. This validation-heavy split is useful because

#Dry	#Wet	#Snow
2378	1431	494

Table 5.10: Class frequencies within the labeled data. Minority classes are over-sampled by the labelers, so this class distribution is not the same as the class distribution for the unlabeled data.

it makes it possible to discriminate between versions of the model during training. From these shards 20 label subsets: $\{\text{subset}_k = \text{shard}_1 \cup \cdots \cup \text{shard}_k : k \in \{1, \ldots, 20\}\}$ were constructed. The size of these label subsets were the label budgets for the experiments that were used to evaluate the performance of each method as a function of the label budget. The validation folds were used for model selection. The test set was only used to compare the different methods after all experiments were performed. When a shard was not part of the label budget for an experiment its instances were treated as unlabeled and included in the unlabeled set for training.



(a) Glare (b) Missing (c) Error (d) Fog

Figure 5.8: Four examples of conditions that are present in the unlabeled data that correspond to input types that are not represented in the labeled training data. Some conditions only present in the unlabeled data can obscure the road surface making it impossible to determine its condition.

Class frequencies for the entire labeled set can be found in Table 5.10. The frequency of each class differs between the labeled and unlabeled data as the *wet* and *snow* classes were over-sampled by the labelers. These conditions occur with far lower frequency than the *dry* class in reality, and they also co-vary more strongly with lighting conditions and time of year. Furthermore, there may be out-of-distribution instances in the unlabeled

data for which the ground truth class cannot be determined at all. In fact, a brief manual inspection of a subset of the data that there revealed at least four types of unlabeled instances that occur in the unlabeled data that do not occur in the labeled data. In Figure 5.8 these four types of instances are shown. This is perhaps the most useful and realistic setting for semi-supervised learning. It is inevitable that some out-of-distribution instances will be present if we cannot afford to inspect and filter all unlabeled examples. If each of these cases were considered as its own class then the type of semi-supervised learning being performed would be *open-world* semi-supervised learning. There is a body of work that attempts to explicitly address the challenges of identifying these unknown classes in the unlabeled data, but such an investigation is left to future work. The semi-supervised methods evaluated in this section have not been benchmarked using unlabeled data collected "in the wild" with unknown classes. In Section 3.1 instances in U were described as being drawn from the same distribution as those in $\mathcal{S}$. The open-world setting relaxes this assumption. One important aspect of this section's evaluation is determining whether or not they remain effective in this more realistic setting.

Data Conditions for Co-Training Blum and Mitchell proved that for co-training to always perform well it is required that the views used are *independent* and *sufficient* [14]. In this case independence is the property that one view of an instance is conditionally independent of the other given the label. Sufficiency is the property that the label can be predicted from either view. In prior work, the independence and sufficiency have been verified when the views are constructed [117], but not when the data collected organically contains two views [14]. In reality it is unlikely that either of these conditions are entirely satisfied. Sufficiency is typically not difficult to verify. If a model or human can be trained to identify the class of an instance from a view then the view must

be sufficient. Independence on the other hand can be tricky to quantify and identify. The authors of MCT designed an experiment in Rothenberger and Diochnos [2023] where they were unable to demonstrate dependence, but were not able to demonstrate independence. Intuitively, we should not expect the views to be entirely independent given the label as it seems like lighting conditions like low-light are clearly correlated with labels like snow and rain. These conditions are sufficient, but not necessary for co-training algorithms to be successful. These conditions are unlikely ever to be entirely satisfied, but prior work has demonstrated co-training can still add value. I will demonstrate that value can be added in the case of road surface classification as well.

5.4.6.2 Experiments

Towards evaluating the performance on co-training algorithms for the task of road surface classification 5-fold cross-validation was performed each algorithm and model type on all 20 label subsets defined in Section 5.4.6.1. Each algorithm and a supervised baseline were evaluated using three types of neural network models: ResNet50, a classification head on top of a pre-trained ViT, and an ensemble of classification heads on top of a pre-trained ViT. ResNet50 was chosen as it is likely the most popular model for image classification. The pre-trained backbone, DINOv2, was trained in a self-supervised way without any labeled data. This model is a leading method in self-supervised representation learning, which is commonly used for SSL. Using a learned representation is a common method of boosting supervised performance in the few-shot learning setting. For the pre-trained transformer the DINOv2 [24] model was chosen because Rothenberger and Diochnos [2023] found it to be most effective. Finally, an ensemble of models learning from a pre-trained representation was selected because co-training methods assume a calibrated uncertainty estimate, which deep ensemble methods can offer. For each label susbet and each model type I test the null hypotheses

# labels	Supervised	+ Self-Supervised Pre-Training	+ Co-Training	+ Meta Co-Training
45	64.2	91.1	93.3	94.6
90	65.0	93.7	93.9	95.8
135	79.0	94.1	94.4	95.8
180	91.4	94.9	94.8	95.8

Table 5.11: Comparison of the best sample mean accuracies for supervised (SL), supervised + self supervised, CT, and MCT on the first four label subsets. Sample mean accuracy increases by applying techniques from self-supervised learning, CT, and MCT.

that the supervised learning and CT baselines produce models that are on average at least as accurate as the models trained with MCT. Figure 5.9 shows a summary of the average gains observed at low label budgets from applying methods from self-supervised learning and semi-supervised learning. Numeric values used in the graphical comparison are given in Table 5.11. Table 5.12 shows the experimental results from comparing all three models and three algorithms for all twenty label subsets and all five folds of cross-validation. Importantly, the results in this table allow for rejection of the both null hypotheses for all three models at the lowest three label budgets for MCT. This gives a strong indication that MCT provides real utility at these low label budgets. MCT also appears to enable learning of significantly more accurate models at some higher label budgets, but the difference between the means is very small and likely not very important.

Co-Training Baseline Experiments In general, the sample mean accuracy of models trained with CT increases with the label budget. For low label budgets CT outperformed the supervised baseline for the same model type (see Table 5.12). At lower label budgets there was complementary benefit of the self-supervised and semi-supervised learning. Additionally, the ensemble models were more effective than the

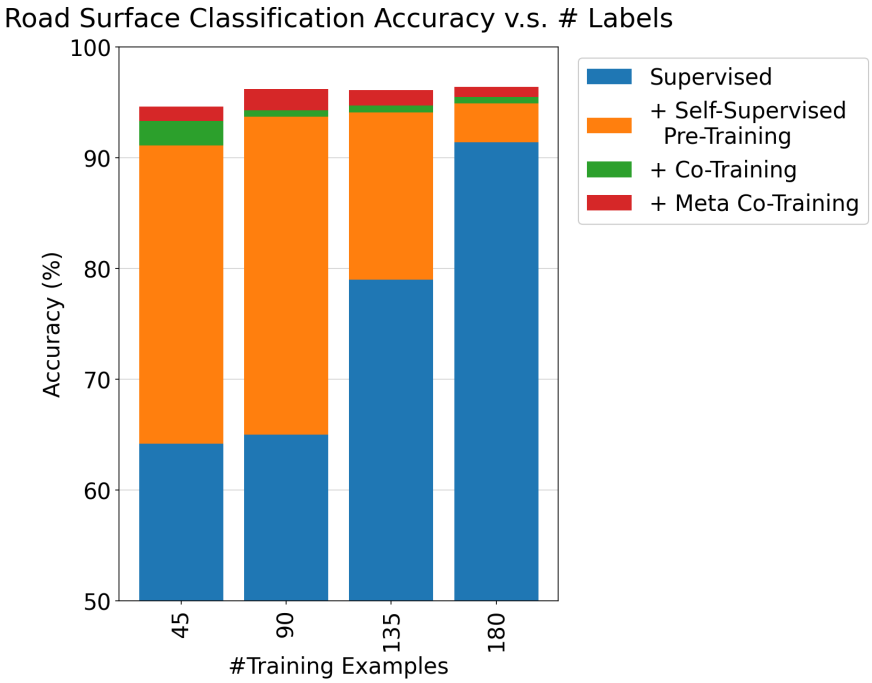


Figure 5.9: The accuracy gained by using methods in combination for the lowest label budgets. At the lowest label budgets the complementary benefits of the self-supervised learning and multi-view semi-supervised learning (CT and MCT) are most pronounced. Table 5.11 has the numeric values.

individual ones indicating that there was also some complementary benefit to having an ensemble of models for each view. I chose to test an ensemble because they are often used to provide more calibrated uncertainty estimates. During the pseudo-labeling step of co-training the instances are chosen based on model confidence. If this confidence is well-calibrated then the pseudo-labels are more likely to be similar to the ground truth. Figure 5.10a shows graphical comparison of the performance of each of the model pairs used for co-training across all label subsets.

Meta Co-Training Experiments To test the hypothesis that MCT produces more accurate models than the baselines on the road surface classification tasks I conduct two hypothesis tests for each model, label subset, and label budget. At each label budget and for each model type the results of MCT training are compared to the corresponding models trained via supervised learning and CT. Two one-tailed Welch's t-tests are conducted. The null hypotheses are that the mean accuracy of models produced by the baselines being compared to are greater than or equal to the mean accuracy of those produced by MCT. In each case I am performing two comparisons, so I apply the Bonferonni correction to the typical $\alpha = 0.05$ significance threshold. This correction results in the significance threshold $\alpha = 0.025$.

or the three lowest label budgets mean accuracies of MCT models were significantly higher than CT for all model types (see Table 5.12). At the two lowest label budgets the MCT models also seemed to benefit from using an ensemble of models for each view. MCT does not rely on confidence-based pseudo-label selection, so the performance benefit from ensembling was likely due to the ensemble of models simply being more effective learners than the individual models. Figure 5.10b shows a graphical comparison of the performance of each of the model pairs used for meta co-training across all label subsets.

The experiments in Table 5.12 showed that the performance of all methods increases with the number of labels. This is expected as with access to sufficient labels SSL would not need to be applied to boost supervised performance. The experiments also showed that MCT remained a high-performer even at higher label budgets. Table 5.12 and Figure 5.10c show that as the label budget increases (for training and validation) the methods begin to perform more similarly. Above 495 training labels (with 1485 validation labels) the difference between the different training methods becomes fairly inconsequential. When access to labels is more restricted, MCT provides the most value. Figure 5.10c shows a graphical comparison of the performance of all models used across all label subsets.

The experiments also showed that stronger individual models for each view produce stronger results for the co-training algorithms. The results indicate that co-training algorithms provide complementary benefits to self-supervised learning. The results also suggest that co-training algorithms are not a replacement for existing self-supervised learning algorithms, but rather can be used in combination with existing algorithms when multiple views are present or can be constructed. Multiple views of nearby road surfaces exist or the application of road surface classification. These views, which have the same road surface condition, can be used to reduce the labeling burden through the application of co-training algorithms.

5.4.6.3 Related Work for Road Surface Classification

Many researchers have applied deep learning to the problem of road surface classification in prior work. Artificial neural networks have been used to predict safety information about the road surface from images taken from phone cameras [101], roadside cameras [130, 1, 98], a combination of the two [102], and even dash cameras [155]. Some approaches combine other data modalities with camera images such as Road Weather

# labels	Supervised Learning ResNet50	Supervised Learning DINOv2	Supervised Learning DINOv2 Ens.	Co-Training ResNet50	Co-Training DINOv2	Co-Training DINOv2 Ens.	Meta Co-Training ResNet50	Meta Co-Training DINOv2	Meta Co-Training DINOv2 Ens.
45	64.2 ± 5.8	90.9 ± 1.3	91.1 ± 1.1	67.9 ± 2.2	92.2 ± 1.2	93.3 ± 1.0	**85.8 ± 2.5**	**93.9 ± 0.8**	**94.6 ± 1.1**
90	65.0 ± 4.3	90.4 ± 1.7	93.7 ± 1.8	74.7 ± 2.0	92.5 ± 1.7	93.9 ± 1.4	**88.8 ± 2.3**	**95.2 ± 0.4**	**95.8 ± 0.5**
135	79.0 ± 4.0	94.1 ± 0.8	93.2 ± 1.2	87.0 ± 1.9	94.4 ± 1.1	94.1 ± 1.2	**90.1 ± 1.1**	**95.8 ± 0.4**	**95.8 ± 0.4**
180	91.4 ± 2.3	94.9 ± 0.6	94.3 ± 1.7	91.2 ± 1.6	94.7 ± 0.5	94.8 ± 0.7	90.5 ± 1.6	95.7 ± 0.4	**95.8 ± 0.2**
225	90.7 ± 1.5	95.6 ± 1.2	94.3 ± 0.8	91.9 ± 1.4	94.9 ± 1.0	94.8 ± 1.0	**93.6 ± 0.9**	**96.4 ± 0.3**	95.3 ± 0.6
270	91.8 ± 1.4	95.2 ± 0.7	94.6 ± 0.9	95.2 ± 1.3	95.0 ± 0.5	96.6 ± 0.6	92.1 ± 0.4	94.9 ± 0.5	**97.1 ± 0.2**
315	93.4 ± 1.9	95.0 ± 0.7	94.9 ± 1.2	94.9 ± 0.5	96.3 ± 0.9	95.6 ± 0.4	94.9 ± 0.4	95.9 ± 0.3	95.8 ± 0.5
360	94.1 ± 1.2	95.2 ± 1.2	92.8 ± 1.5	96.6 ± 0.2	96.2 ± 0.8	96.0 ± 1.0	96.8 ± 0.3	96.0 ± 0.3	**97.1 ± 0.3**
405	94.7 ± 2.1	94.7 ± 1.4	96.7 ± 1.4	97.1 ± 0.5	96.7 ± 0.6	96.5 ± 1.0	94.8 ± 1.1	96.2 ± 0.7	96.8 ± 0.2
450	97.8 ± 1.4	94.8 ± 0.4	96.4 ± 0.6	97.4 ± 1.5	96.6 ± 0.8	97.4 ± 0.8	96.5 ± 0.5	96.5 ± 0.2	97.5 ± 0.2
495	97.4 ± 0.9	95.9 ± 0.7	95.7 ± 0.8	97.8 ± 0.0	95.9 ± 0.4	95.8 ± 0.6	95.6 ± 0.3	**97.4 ± 0.3**	**98.0 ± 0.2**
540	97.9 ± 1.7	95.4 ± 0.5	96.1 ± 0.7	97.7 ± 0.0	97.0 ± 0.3	97.0 ± 0.5	95.6 ± 0.3	**97.6 ± 0.4**	**97.8 ± 0.2**
585	95.6 ± 0.5	96.3 ± 1.0	95.5 ± 0.7	96.1 ± 0.5	97.3 ± 0.4	96.0 ± 0.8	96.1 ± 0.6	**97.8 ± 0.4**	**97.4 ± 0.3**
630	96.2 ± 1.6	95.9 ± 0.9	96.6 ± 0.7	97.3 ± 0.0	95.5 ± 0.6	96.4 ± 0.6	96.6 ± 0.3	**97.4 ± 0.1**	**97.3 ± 0.2**
675	98.0 ± 0.6	96.1 ± 1.1	96.4 ± 0.9	97.1 ± 0.0	97.2 ± 0.7	96.0 ± 0.7	96.7 ± 0.3	97.2 ± 0.3	**97.5 ± 0.2**
716	97.2 ± 0.7	96.2 ± 1.0	96.7 ± 0.8	97.0 ± 0.0	96.9 ± 0.4	97.2 ± 0.8	97.2 ± 0.2	**97.6 ± 0.5**	**98.4 ± 0.1**
761	96.2 ± 1.8	96.6 ± 0.9	96.3 ± 0.8	96.4 ± 0.0	97.5 ± 0.7	97.4 ± 0.5	**97.8 ± 0.4**	97.3 ± 0.3	**98.4 ± 0.1**
804	97.4 ± 0.4	97.2 ± 0.8	96.8 ± 0.7	96.5 ± 0.0	97.3 ± 0.7	97.3 ± 0.6	**98.3 ± 0.3**	97.4 ± 0.1	**98.4 ± 0.1**
850	97.0 ± 0.9	96.2 ± 0.8	97.0 ± 0.6	97.7 ± 0.0	97.7 ± 0.4	97.5 ± 0.3	**98.4 ± 0.6**	98.1 ± 0.2	**97.9 ± 0.2**
861	96.5 ± 0.5	96.7 ± 0.5	97.1 ± 0.4	97.1 ± 0.3	97.9 ± 0.4	97.2 ± 0.3	**97.6 ± 0.3**	97.6 ± 0.2	**98.1 ± 0.3**

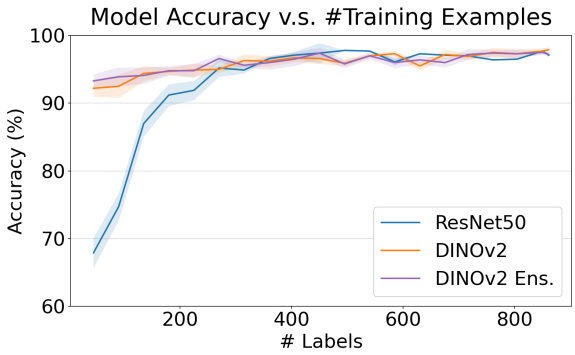
Table 5.12: Comparison of all three models and three algorithms. For all cross-validation experiments the mean testing accuracy as well as the 95% confidence interval are given. Results that allow rejection of the null hypothesis for both baselines are shown in **bold**.

Information System (RWIS) sensor data [68, 26] and weather data [25, 73]. In some prior works researchers leverage transfer learning [101, 102, 98] for better performance.

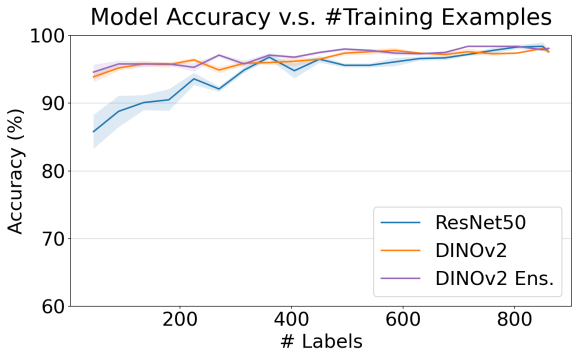
5.5 Conclusion

I proposed MCT, a novel semi-supervised learning algorithm. I showed that MCT significantly out-performs co-training on the ImageNet-1% and ImageNet-10% datasets. I showed that MCT achieved significantly better accuracy than a deep ensemble baseline demonstrating that MCT provides utility beyond what can be attributed to ensembling. I showed that MCT achieved significantly higher accuracy on several fine-grained image classification task than competitive baselines.

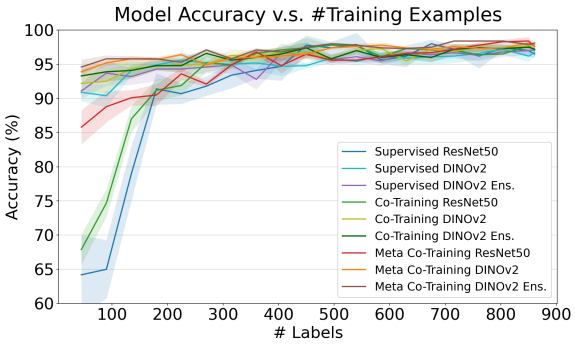
I presented evidence that self-supervised learned representations are good candidates as views for co-training. I showed that the views constructed this way likely contain



(a) Sample mean accuracies of the different models trained using CT as a function of the label budget. All model pairs used across all label subsets for CT are shown.



(b) Sample mean accuracies of the different models trained using MCT as a function of the label budget. All model pairs used across all label subsets for MCT are shown.



(c) Sample mean accuracies of the different models trained using CT, MCT, and supervised learning as a function of the label budget.

Figure 5.10: Comparison of co-training methods' accuracies as a function of their training set sizes. Shaded regions indicate 95% confidence intervals.

independent information and that models trained on these views, or concatenation of these views, can complement each other to produce stronger classifiers. This is encouraging because it shows that co-training algorithms provide improvement on top of self-supervised learning that can be realized for any domain on which multiple self-supervised learning models have been trained.

Even if no self-supervised trained models are available, it may be possible to collect additional views without any more human labeling effort. In Section 5.4.6 was shown a case study in which two views for co-training were identified by leveraging knowledge

of the data domain. For the purpose of this case study I made the assumption that labels assigned by a human labeler to one view would likely correspond to the same ground-truth label for the other view. This assumption is important if one cares about correctly classifying the condition at both views. If only predicting well on the images from a single camera was important, then even cameras that showed different conditions on the road surface but that were sufficient to predict the road surface condition of interest would be useful views.

In Section 5.4 there were cases in which CT achieved higher accuracy than DE with less models and degenerate pseudo-labeling. CT may have out-performed DE because constituent models of CT, which are trained on different views, may be naturally more diverse than those of DE. Investigating the diversity of the ensemble models for DE and the view models for CT was out of scope for this chapter, but is an interesting question for future work. This diversity is unlikely to be constant for CT and MCT over the course of training as the models are learning from each-other and gradually learning to predict more similarly. Investigating the dependence or independence of the individual models for CT and MCT over the course of training is an interesting direction for understanding the training dynamics of these methods.

Chapter 6

Policy Gradient Pseudo-Labeling

Access to high-quality labeled data is often limited or expensive while unlabeled data are plentiful and cheap. Semi-supervised learning algorithms, which improve generalization from few labeled data, have become increasingly popular. Prior work has demonstrated that pseudo-labeling based methods are among the strongest algorithms for semi-supervised learning. Of particular interest are emerging approaches that leverage reinforcement learning to assign useful pseudo-labels. Existing approaches, however, use only batch-level rewards that fail to capture the relative utility of individual pseudo-labels within a batch. I present a novel algorithm for semi-supervised learning that leverages reinforcement learning to assign productive pseudo-labels. My algorithm computes an example-level reward associated with assigning a pseudo-label to an unlabeled instance based on the pseudo-labeled example's contribution to the learning process. This example-level reward attribution allows my models to learn which pseudo-labels it should assign to unlabeled data to improve accuracy on limited labeled data. I experimentally validate my approach on a wide range of semi-supervised benchmark datasets.

6.1 Chapter Introduction

Semi-supervised learning (SSL) has emerged as an important area of machine learning research. The field of semi-supervised learning makes use of unlabeled instances to boost the performance of models that learn from limited supervision, effectively reducing the necessary human effort for labeling. The rise of SSL is motivated by the relative difficulty of gaining access to large amounts of high-quality labeled data. In contrast to the difficulty of acquiring labeled data, unlabeled data are relatively inexpensive

to gather and store. The value of SSL is particularly pronounced in the context of computer vision (CV) when manual human effort is typically required to assign labels.

Many algorithms for SSL have been developed for utilizing available unlabeled data to improve generalization of models trained using limited supervision. There are several different approaches to semi-supervised learning such as pseudo-labeling-based methods [83, 14], self-supervised learning methods [109, 100], and consistency regularization methods [156, 12]. Of particular interest to us in this work are pseudo-labeling based methods. Recent advances in pseudo-labeling algorithms for semi-supervised learning have focused on optimizing pseudo-label assignments through bi-level optimization or reinforcement learning (RL) [105, 117, 60, 61]. These algorithms seek to improve the predictions they make on labeled data by optimizing the labels they assign on unlabeled data.

Although many algorithms for SSL have been developed, achieving strong generalization in the presence of limited labeled data remains a challenging problem. One exciting direction for pseudo-labeling algorithm development has been to utilize RL to improve pseudo-label assignments. In this paper I demonstrate that several existing approaches either are explicitly described as, or can be interpreted as, RL. In all such cases, RL has been employed to improve the assignment of pseudo-labels to unlabeled data. I will show that even though algorithms for RL-based pseudo-labeling have been proposed in prior work, their respective reward functions do not capture the utility of an individual pseudo-labeled example to the learning process. Existing RL algorithms for pseudo-labeling do not model the individual value of each pseudo-labeled example as the marginal contribution of that example to the reduction in loss. To address this gap, I propose *policy gradient pseudo-labeling* (PGPL). PGPL is a novel policy-gradient-based pseudo-labeling method for semi-supervised learning that incorporates example-level rewards. These example-level rewards reflect the marginal reduction in loss associated

with assigning a pseudo-label to an unlabeled instance. I conduct experiments on the ImageNet-1% and ImageNet-10% datasets to evaluate the performance of my approach. The experiments show that my approach has complementary value to existing self-supervised learning work. I achieve new state-of-the-art top-1 accuracy on both benchmark tasks.

Chapter Contributions

- I introduce PGPL, a new algorithm for semi-supervised learning based on reinforcement learning.
- I experimentally evaluate my approach on a wide range of semi-supervised benchmark datasets.
- I release an implementation of my algorithm to aid future algorithm development and research.

6.2 Pseudo-Label Assignment as RL

First, I will show that the *meta pseudo labels* (MPL) [105] method corresponds to a specific formulation of RL for pseudo-labeling. It will follow that *meta co-training* (MCT) [117] also corresponds to a similar RL formulation. Subsequently, I will address the RL formulation of *reinforcement learning-guided semi-supervised learning* (RLGSSL) [61]. Through my discussion I will identify ways in which these RL for SSL formulations can be improved. These potential improvements will motivate my method, which I discuss in the following section.

6.2.1 MPL and MCT as RL

Consider MPL as an example. In MPL a teacher and a student are jointly optimized to produce a student that achieves low loss on some held-out dataset. This is achieved by the teacher providing pseudo-labels to the student, observing the change in performance

of that student, and then learning to make better pseudo-labels using this feedback. At the lower level (Equation 6.2) of the optimization the student parameters θ_S are optimized as a function of the teacher parameters θ_T. Given a pseudo-labeled batch of unlabeled instances, the student is optimized to predict the pseudo-labels well given the unlabeled instances:

$$\mathcal{L}_{student}(f_{\theta_S}; \mathbf{U}) = \frac{1}{m} \sum_{j=n+1}^{n+m} \ell(f_{\theta_S}(x_j), \mathcal{Y} \sim f_{\theta_T}(x_j)) \text{, and} \tag{6.1}$$

$$\theta'_S \in arg\ min_{\theta_S} \mathcal{L}_{student}\left(f_{\theta_S}; (\mathbf{U},)\right) . \tag{6.2}$$

At the upper level (Equation 6.3) the teacher parameters are optimized to improve the student:

$$\theta'_T \in arg\ min_{\theta_T} \mathcal{L}\left(f_{\theta'_S}; \mathbf{L}\right) . \tag{6.3}$$

Rather than computing the expensive argmin in Equation 6.2, Pham et al. [2021] resort to a single update of stochastic gradient descent. The resulting teacher loss at each iteration is

$$\mathcal{L}_{teacher}(f_{\theta_T}; \mathbf{U}) = h \cdot \frac{1}{m} \sum_{j=n+1}^{n+m} CE(f_{\theta_T}(x_j), \mathcal{Y} \sim f_{\theta_T}(x_j)) \tag{6.4}$$

with h a scalar:

$$h = \left\langle \nabla_{\theta'_S} \mathcal{L}(f_{\theta'_S}; \mathbf{L})^T, \nabla_{\theta_S} \mathcal{L}_{student}(f_{\theta_S}; \mathbf{U}) \right\rangle . \tag{6.5}$$

Here the angle brackets $< \cdot,\ \cdot >$ denote the inner product. The bi-level optimization of this objective proceeds as iterations of gradient descent on the student and teacher objectives. The student model is the product of the optimization and the teacher is discarded. The h term is expensive to compute for large student networks, so the authors resort to the first-order Taylor approximation of h, which is the negative difference in

loss achieved by the student step $\hat{h} = \mathcal{L}(f_{\theta_S}; \mathbf{L}) - \mathcal{L}(f_{\theta'_S}; \mathbf{L})$.

Now let us consider an episodic reinforcement learning setting in which an agent parameterized by θ following a policy π_θ must maximize expected reward $E[\sum_{i=1}^{T} r_i]$. Consider episodes with length only $T = 1$; e.g., the agent takes only one action (such as predicting a pseudo-label) in each episode and receives a single reward. Assume that the agent starts in a random starting state $s_1 \in S$ and can take one of a discrete set of actions $a \in A$. By applying the policy gradient theorem to maximize the reward, I follow the gradient

$$\nabla_\theta \mathbb{E}[\sum_{i=1}^{T} r_i] = r_1 \cdot \mathbb{E}[\nabla_\theta log\ \pi_\theta(a|s_1)]. \tag{6.6}$$

Let the result of taking action a be to predict a hard pseudo-label of some class for an unlabeled instance u. Let the environment step result in a gradient update on another model (the student). The environment provides reward r_1, which is the negative difference in loss of the student $\hat{h}$. It follows that in such an RL setting maximizing the reward of such a labeling policy (teacher) would be equivalent to minimizing Equation 6.4;

$$\begin{aligned} -\nabla_\theta \mathbb{E}[\sum_{i=1}^{T} r_i] &= -r_1 \cdot \mathbb{E}[\nabla_\theta log\ \pi_\theta(a|s_1)], \\ &= r_1 \cdot \mathbb{E}[\nabla_\theta CE(\pi_\theta(s_1), a)], \text{ and} \\ &= \hat{h} \cdot \mathbb{E}[\nabla_\theta CE(f_\theta(u), \hat{y}_u)]. \end{aligned}$$

Thus, I have shown that the MPL teacher is equivalent to optimizing a policy in the above reinforcement learning setting. I find that MPL can be understood as using RL to optimize pseudo-labels assigned by the teacher to maximize the reduction in student loss. Because MCT uses the same teacher loss as MPL for both of its constituent models it follows directly that MCT corresponds to a similar RL formulation.

In the RL formulation for MPL the reward is $\hat{h}$, a scalar function of a set of pseudo-labeled examples. $\hat{h}$ represents the reduction in loss associated with a batch of pseudo-labeled examples. The MPL reward function captures the marginal reduction of loss associated with a *batch* but not with an individual *example*.

6.2.2 RLGSSL

RLGSSL is formulated directly as an RL for SSL algorithm. RLGSSL adopts a deterministic policy approach rather than the stochastic one of MPL and MCT. For a state $s = (X, Y \sim \mathbf{L}^{(1)}, U \sim \mathbf{U}^{(1)})$ and action $a = f_\theta(U)$ the reward for RLGSSL is defined as follows:

$$X^m = \mu X + (1 - \mu U), \tag{6.7}$$

$$Y^m = \mu Y + (1 - \mu f_\theta(U)), \text{ and} \tag{6.8}$$

$$R(s, a) = -\frac{1}{C \cdot |X^m|} \sum_{i=1}^{k} \| f_\theta(X^m) - Y^m \|_2^2, \tag{6.9}$$

where C is the number of classes and $\mu \sim Beta(\alpha, \beta)$ with α and β being hyperparameters.

The RLGSSL reward is a scalar function of a set of labeled examples and a set of pseudo-labeled examples. The RLGSSL reward is designed to be a proxy for the generalization of the model providing the pseudo-labels. The heuristic used to measure how well the model generalizes is the model's mean squared error on MixUp [172] examples created from labeled and pseudo-labeled. Because this reward function is computed as an average over the batch it does not capture the marginal contribution of each pseudo-label action. Because it is a heuristic for generalization ability of the model, it also does not exactly capture the value of each pseudo-label to the learning process.

6.2.3 Drawbacks of Existing RL for SSL Algorithms

As shown above, existing RL for SSL methods utilize only a scalar, batch-level reward. This batch-level reward does not effectively capture the individual reward that should be associated with each pseudo-label. This batch-level reward is equivalent to assigning every element in the batch the same reward. Importantly, *not all pseudo-labels will contribute equally in every batch.* To see this, one only need consider a case when one pseudo-label in the batch incurs minimal loss because it is identical to the prediction the learner would have made. This pseudo-label can only have zero reward while any other pseudo-label in the batch that is not identical to the learner's prediction must have non-zero reward. The ratio between the rewards of these two pseudo-labels could not have greater magnitude (it is infinite), yet existing algorithms assign both examples the same reward! Furthermore, I cannot resort to a batch size of 1. In this case the scalar reward for each batch would apply to only one action, but this does not capture the dependence of the learner's performance on multiple elements in the batch. In order to provide an effective signal for learning a batch must be large enough to represent the properties of the distribution from which it is drawn. The utility of a particular pseudo-label to learning is dependent on the pseudo-labels assigned to other instances. Batch size and composition matter when determining the reward associated with a pseudo-label.

In summary, the batch-level reward is insufficient to capture the reward signal for individual pseudo-labels. In the latter case, when the batch-level and example-level rewards are the same, I still are not satisfied because the reward is not practically meaningful. Furthermore, RLGSSL does not assign reward based on the reduction in loss associated with assigning a pseudo-label to an unlabeled instance. In light of these drawbacks there is significant room for improvement in RL for SSL algorithms.

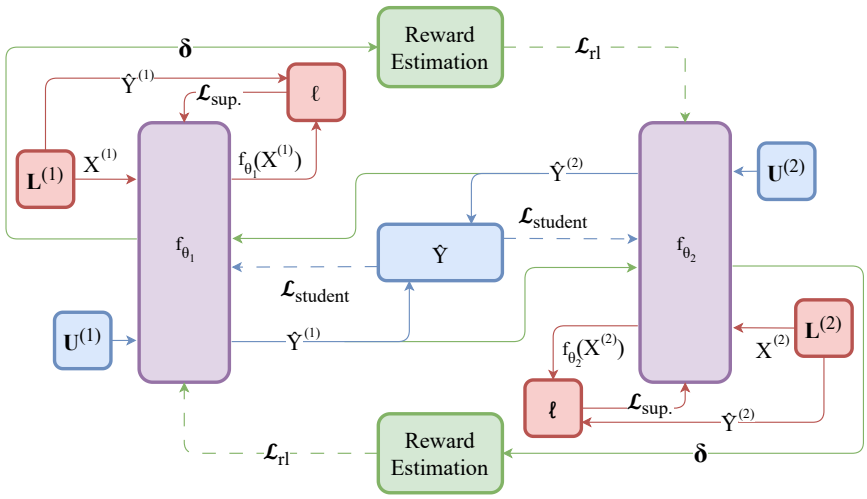


Figure 6.1: An overview of my proposed PGPL method. Both models function as student and teacher. As a student, each model learns from the joint pseudo-label of the two models $\hat{Y}$. As a teacher, each model learns using $\mathcal{L}_{rl}$ to produce better pseudo-labels.

In the next section I discuss my proposed method of RL for SSL that addresses these drawbacks.

6.3 Method

In this section I present my method PGPL. Similar to prior RL for SSL work [105, 117, 61], my algorithm operates within a student-teacher framework. I formulate my algorithm as a stochastic policy gradient optimization. Unlike prior work, I compute example-level rewards that capture the marginal reduction in loss of the student network attributable to the action of the teacher network. A graphical overview is presented in Figure 6.1. The remainder of this section details the elements of my approach.

6.3.1 Student-teacher Framework

Many recent algorithms in SSL employ student-teacher frameworks [105, 117, 61]. Previous work has highlighted the benefits of student-teacher frameworks in terms of

the stability of their optimization and strong generalization [141, 176, 157]. Recent work in semi-supervised learning [117] has highlighted the effectiveness of student-teacher frameworks in which both models act as both student and teacher. Algorithms that utilize student-teacher relationships in this way are referred to as co-training algorithms. Co-training-style algorithms work by taking advantage of differences between models trained on independent input spaces (commonly referred to as *views*). Motivated by the recent success of MCT, I use a student and a teacher each trained on different views rather than a mean teacher. Unlike prior co-training algorithms, I use the average of the student and teacher models predictions as the target for the student. The choice to use to co-prediction as the target for the student is motivated by the insight that the co-prediction was demonstrated to be more accurate than either model's prediction in [117] for MCT and co-training (CT) [14].

My student-teacher framework yields the following student loss terms:

$$\mathcal{L}^{(1)}_{student} = \mathcal{L}\left(f_{\theta_1}; (\mathbf{U}^{(1)}, \frac{f_{\theta_1}(\mathbf{U}^{(1)}) + f_{\theta_2}(\mathbf{U}^{(2)})}{2})\right), \text{ and} \tag{6.10}$$

$$\mathcal{L}^{(2)}_{student} = \mathcal{L}\left(f_{\theta_2}; (\mathbf{U}^{(2)}, \frac{f_{\theta_1}(\mathbf{U}^{(1)}) + f_{\theta_2}(\mathbf{U}^{(2)})}{2})\right). \tag{6.11}$$

The student for each view learns from its teacher replicating its predictions on the unlabeled instances.

6.3.2 Example-level Reward Calculation

The observations in Section 6.2 highlighted the need for a function that assigns a reward to each example in a batch representing the marginal contribution of that example. I define the reward $R(u_j, \hat{y}_j)$ for a pseudo-label $\hat{y}_j$ and unlabeled instance u_j to be the marginal contribution to reduction in loss of a student model over all pseudo-labeled

batches containing $(u_j, \hat{y}_j)$. This yields

$$R(u_j, \hat{y}_j) = \frac{1}{\binom{|\mathbf{U}|-1}{k}} \sum_{\substack{U \subseteq \mathbf{U} \setminus \{u\} \\ |U|=k}} \Big[v(U \cup \{u_j\}, \hat{Y} \cup \{\hat{y}_j\}) - v(U, \hat{Y}) \Big], \tag{6.12}$$

where

$$v(U, \hat{Y}) = \mathcal{L}(f_\theta; \mathbf{L}) - \mathcal{L}(f_{\theta - \eta \cdot \nabla_\theta \mathcal{L}_{student}(f_\theta; \mathbf{U})}; \mathbf{L}) \tag{6.13}$$

is the change in loss $\mathcal{L}(f_\theta, \mathbf{L})$ over a gradient step on an unlabeled batch U and pseudo labels $\hat{Y}$. Computing the reward exactly would require $\binom{|U|}{k}$ gradient updates and is clearly impractical. In this form $R(u_j, \hat{y}_j)$ closely resembles the Shapley value of the pseudo-labeled example towards the reduction of the loss of the student for all batches of size k. Inspired by the effectiveness of kernel SHAP [90] I approximate $R(u_j, \hat{y}_j)$ using a least-squares regression. I present my algorithm for approximating the example-level rewards in Algorithm 2.

In Algorithm 2 my strategy is to estimate the marginal contributions $\mathbf{r}^{(i)}$ using linear least squares. First, N random subsets of U of size m are drawn. A matrix of N rows and $|U^{(i)}|$ columns is constructed such that membership of the jth example in the nth subset of the unlabeled batch $U^{(i)}$ is represented by $M_{n,j} = 1$. A gradient step is performed for each subset along with their pseudo-labels as shown in line 11. The resulting change in loss on the labeled set is recorded for each such subset in the vector δ. Finally, the approximation for $\mathbf{r}^{(i)}$ is simply the minimizer of $||M\mathbf{r}^{(i)} - \delta||$.

6.3.3 Reinforcement Learning Formulation

Because both approaches involve the application of RL, one might naturally assume that my application of RL and that of RLGSSL are similar. Contrary to that intuition, the way in which I leverage RL is actually quite different. In PGPL RL is used to *maximize*

Algorithm 2 Reward Estimation (RE)

Input: $f^{(i)}, \theta_i, U^{(i)}, A^{(i)}, L^{(i)}, N, m < |U^{(i)}|$
Initialize $\delta \leftarrow \mathbf{0}^m$
Initialize membership mask $M \leftarrow \mathbf{0}^{N \times |U^{(i)}|}$
for step $n \in \{1 \dots N\}$ **do**
  Generate a random subset $S \subseteq \{1, 2, \dots, |U^{(i)}|\}$ such that $|S| = m$
  for each $j \in S$ **do**
    Set $M_{n,j} \leftarrow 1$ {Update membership mask}
  end for
  $\overline{U}, \overline{Y} \leftarrow \{U_j^{(i)} | M_{n,j} = 1\}, \{A_j^{(i)} | M_{n,j} = 1\}$
  {Update weights based on select pseudo-labels}
  $\theta_i' \leftarrow \theta_i - \eta \cdot \nabla_{\theta_i} \mathcal{L}(f_{\theta_i}^{(i)}; (\overline{U}, \overline{Y}))$
  {Compute the change in loss from the weight update}
  $\delta_i \leftarrow \mathcal{L}(f_{\theta_i}^{(i)}; \mathbf{L}^{(i)}) - \mathcal{L}(f_{\theta_i'}^{(i)}; \mathbf{L}^{(i)})$
end for
{Estimate example-level rewards}
$\mathbf{r}^{(i)} \leftarrow (M^T M)^+ M^T \delta$
return $\mathbf{r}^{(i)}$

the value of pseudo-labels, but in RLGSSL RL is used to *align model predictions with mixup targets.*

I formulate the SSL problem as a simple episodic RL problem. An agent policy π_θ is trained to maximize an expected reward over trajectories of length 1:

$$\mathbb{E}[\sum_{i=1}^{1} R(s_i, a \sim \pi_\theta(s_0))] = \mathbb{E}[R(s_0, a \sim \pi_\theta(s_0))], \text{ and}$$
$$= \mathbb{E}[R(u_j, \hat{y}_j)].$$

The agent observes only one state, takes one action, and receives one reward. The policy is the teacher model f_θ in my framework. The state corresponds to a single unlabeled instance u. The action is a pseudo-label assigned to u sampled from $\hat{y}_j \sim \pi_\theta(u) = f_\theta(u)$. To compute the reward, the environment estimates the marginal contribution of each pseudo-labeled example to the reduction in the student's loss on the labeled data.

Applying the policy gradient theorem I get the following RL loss term:

$$\mathcal{L}_{rl} = \left\langle \mathbf{r}, log(\pi_\theta(\hat{Y}|U)) \right\rangle, \tag{6.14}$$

Where U is a batch of unlabeled instances, $\hat{Y}$ are the pseudo-labels sampled from $\pi_\theta(U)$, and $\mathbf{r}$ is the vector containing the rewards for each pseudo-labeled example. Because models act as both student and teacher within my student-teacher framework, the above RL loss is applied to both models. Writing the loss for each view's model I get: $\mathcal{L}_{rl}^{(1)} = \left\langle \mathbf{r}^{(1)}, log(\pi_{\theta_1}(\hat{Y}|U^{(1)})) \right\rangle$ and $\mathcal{L}_{rl}^{(2)} = \left\langle \mathbf{r}^{(2)}, log(\pi_{\theta_2}(\hat{Y}|U^{(2)})) \right\rangle$. Each model effectively learns which predictions improve the other model. This objective has additional utility in the context of a co-training algorithm. Co-training algorithms derive their benefit from the independence between the two participant models [14]. Predictions that improve the student must necessarily not be predictions that the student would have made. By learning in this way the teacher is trained to predict more accurately, but also differently from the student. Through RL I are able to extend the usefulness of the co-training framework.

The full algorithm for my method is presented in Algorithm 3. The procedure begins by computing the average of the two soft labels predicted by each model. These are the pseudo labels that are used to compute the student losses and for reward estimation. The next step is the computation of the student loss, which facilitates the transfer of knowledge between models. This computation occurs on lines 8-9. On lines 14-15 the rewards associated with the pseudo-labels are computed in the manner described above. Subsequently the RL loss is computed on lines 16-17. Finally, on lines 22-23 the weights of the network are updated to minimize a combined objective. In addition to the student loss $\mathcal{L}_{student}$ and the RL loss $\mathcal{L}_{rl}$ I include a supervised loss term $\mathcal{L}_{sup.}$ to expedite convergence.

Algorithm 3 Policy Gradient Pseudo-Labeling (PGPL)

Input: $\mathbf{L}^{(1)}$, $\mathbf{L}^{(2)}$, $\mathbf{U}$, $f^{(1)}$, $f^{(2)}$, θ_1, θ_2, T, ℓ, η, N, m, λ_1, λ_2
for step $t \in \{1 \dots T\}$ **do**
  $U^{(1)}, U^{(2)} \sim \mathbf{U}$
  $\hat{Y}^{(1)} \leftarrow f^{(1)}_{\theta_1}(U^{(1)})$
  $\hat{Y}^{(2)} \leftarrow f^{(2)}_{\theta_2}(U^{(2)})$
  $\hat{Y} \leftarrow \frac{\hat{Y}^{(1)}+\hat{Y}^{(2)}}{2}$
  {Compute the student losses}
  $\mathcal{L}^{(1)}_{student} \leftarrow \mathcal{L}(f_{\theta_1}; (\mathbf{U}^{(1)}, \frac{f_{\theta_1}(\mathbf{U}^{(1)})+f_{\theta_2}(\mathbf{U}^{(2)})}{2})$
  $\mathcal{L}^{(2)}_{student} \leftarrow \mathcal{L}(f_{\theta_2}; (\mathbf{U}^{(2)}, \frac{f_{\theta_1}(\mathbf{U}^{(1)})+f_{\theta_2}(\mathbf{U}^{(2)})}{2}))$
  $X^{(1)}, Y^{(1)} \sim \mathbf{L}^{(1)}$
  $X^{(2)}, Y^{(2)} \sim \mathbf{L}^{(2)}$
  {Compute the RL losses}
  $A^{(1)} \sim \hat{Y}, A^{(2)} \sim \hat{Y}$
  $\mathbf{r}^{(1)} \leftarrow RE(f^{(2)}, \theta_2, U^{(2)}, A^{(2)}, (X^{(2)}, Y^{(2)}), N, m)$
  $\mathbf{r}^{(2)} \leftarrow RE(f^{(1)}, \theta_1, U^{(1)}, A^{(1)}, (X^{(1)}, Y^{(1)}), N, m)$
  $\mathcal{L}^{(1)}_{rl} \leftarrow \left\langle \mathbf{r}^{(1)}, log(\pi^{(1)}_{\theta_1}(A^{(1)}|U^{(1)})) \right\rangle$
  $\mathcal{L}^{(2)}_{rl} \leftarrow \left\langle \mathbf{r}^{(2)}, log(\pi^{(2)}_{\theta_2}(A^{(2)}|U^{(2)})) \right\rangle$
  {Compute the supervised losses}
  $\mathcal{L}^{(1)}_{sup.} \leftarrow \mathcal{L}(f^{(1)}_{\theta_1}; (X^{(1)}, Y^{(1)}))$
  $\mathcal{L}^{(2)}_{sup.} \leftarrow \mathcal{L}(f^{(2)}_{\theta_2}; (X^{(2)}, Y^{(2)}))$
  {Update weights using SGD}
  $\theta_1 \leftarrow \theta_1 - \eta \cdot \nabla_{\theta_1}(\mathcal{L}^{(1)}_{sup.} + \lambda_1 \mathcal{L}^{(1)}_{student} + \lambda_2 \mathcal{L}^{(1)}_{rl})$
  $\theta_2 \leftarrow \theta_2 - \eta \cdot \nabla_{\theta_2}(\mathcal{L}^{(2)}_{sup.} + \lambda_1 \mathcal{L}^{(2)}_{student} + \lambda_2 \mathcal{L}^{(2)}_{rl})$
end for

6.4 Experiments

In order to validate the effectiveness of PGPL I conduct 5-fold cross-validation experiments on a range of popular and challenging benchmark datasets for image classification: CIFAR-10 [79], CIFAR-100 [79], STL-10 [34], Food101 [16], iNaturalist 2021 [63], and the ImageNet dataset. These datasets cover a large range of image resolutions and dataset sizes. I reserved one fold for validation, which is used for model selection, and another for testing. The remaining three folds are used for the training and unlabeled sets. Accuracy reported in this section is testing accuracy. Cross-validation is used

Table 6.1: Experimental comparisons on the Food101, and STL-10 datasets. For all cross-validation experiments the mean testing accuracy as well as the 95% confidence interval are given. Results that allow rejection of the null hypothesis for both baselines are shown in **bold**.

	Dataset							
Method	Food101				STL-10			
# labels / class	1	3	5	10	1	3	5	10
CT [14]	77.2	90.5	93.1	94.7	75.4	96.8	98.9	99.6
MixMatch [12]	70.3	87.3	92.2	94.5	83.4	96.9	98.6	99.4
FixMatch [133]	85.3	92.1	94.0	95.0	87.1	99.1	99.4	99.5
FlexMatch [170]	71.2	88.4	92.2	94.4	86.2	97.4	99.1	99.4
MCT [117]	83.31 ± 2.96	93.43 ± 0.52	94.31 ± 0.21	94.43 ± 0.32	72.64 ± 23.40	99.47 ± 0.49	99.04 ± 1.59	99.77 ± 0.15
RLGSSL [61]	62.10 ± 2.91	81.18 ± 0.80	84.23 ± 0.77	87.44 ± 0.20	46.56 ± 4.96	75.07 ± 1.62	82.49 ± 2.44	90.23 ± 1.47
PGPL	84.63 ± 2.74	**93.86 ± 0.49**	94.18 ± 0.23	94.37 ± 0.34	86.86 ± 9.41	99.43 ± 0.68	99.45 ± 0.79	99.73 ± 0.14

Table 6.2: Experimental comparisons on the iNaturalist 2021 dataset. For iNaturalist 2021 only the 1010 most frequent classes were used. For all cross-validation experiments the mean testing accuracy as well as the 95% confidence interval are given. Results that allow rejection of the null hypothesis for both baselines are shown in **bold**.

	iNaturalist			
Examples per class	1	3	5	10
CT	24.2	47.2	59.2	70.5
CT (single)	22.8	43.6	52.4	63.1
MixMatch	24.6	36.6	42.6	50.3
FixMatch	24.0	40.8	48.9	58.6
FlexMatch	23.5	34.1	38.3	43.0
MCT	19.81 ± 0.92	40.61 ± 1.10	53.99 ± 0.96	68.80 ± 0.43
RLGSSL	20.70 ± 1.65	38.74 ± 1.18	47.40 ± 0.22	60.27 ± 0.62
PGPL	**27.64 ± 0.36**	**52.96 ± 0.88**	**64.29 ± 0.32**	**74.67 ± 0.19**

to create a sample of model performances for PGPL as well as MCT and RLGSSL baselines. To test the hypothesis that PGPL produces more accurate models than the MCT and RLGSSL baselines I conduct two hypothesis tests for each dataset and label budget. These are one-tailed Welch's t-tests as the variances of the tested populations are often unequal. The null hypothesis is that the mean accuracy of models produced by the baseline being compared to is greater than or equal to the mean accuracy of those produced by PGPL. In each case two comparisons were performed, so I apply the

Table 6.3: Experimental comparisons on the CIFAR-10 and CIFAR-100 datasets. For all cross-validation experiments the mean testing accuracy as well as the 95% confidence interval are given. Results that allow rejection of the null hypothesis for both baselines are shown in **bold**.

	Dataset							
Method	CIFAR-10				CIFAR-100			
# labels / class	1	3	5	10	1	3	5	10
CT [14]	71.5	94.5	96.9	97.7	64.7	82.2	85.0	87.6
MixMatch [12]	69.8	89.9	93.0	94.5	46.6	64.8	71.4	75.3
FixMatch [133]	84.0	93.4	94.3	94.1	51.6	68.6	74.0	77.7
FlexMatch [170]	82.0	91.9	93.9	94.6	48.6	66.1	71.5	74.5
MCT [117]	78.67 ± 17.65	98.35 ± 1.88	98.97 ± 0.21	99.02 ± 0.12	63.31 ± 6.31	85.94 ± 1.81	88.06 ± 0.62	89.38 ± 0.25
RLGSSL [61]	62.34 ± 4.81	83.15 ± 1.21	89.41 ± 1.53	94.35 ± 0.83	51.99 ± 2.84	74.17 ± 1.35	79.19 ± 0.69	84.66 ± 0.50
PGPL	88.24 ± 9.07	98.47 ± 1.84	99.13 ± 0.05	99.10 ± 0.09	**78.73 ± 2.82**	**87.56 ± 0.83**	**88.85 ± 0.29**	89.57 ± 0.36

Bonferonni correction to the typical 0.05 significance threshold to obtain the threshold $\alpha = 0.025$.

I conducted experiments on each of the first five datasets using 1, 3, 5, and 10 examples per class. As is typical when evaluating semi-supervised learning methods, I sampled an equal number of examples from the training folds for each class to form the labeled set and treated the rest of the data as unlabeled. For STL-10 an unlabeled set is available, so in addition to the training examples that are not selected to be labeled I used the provided unlabeled set as unlabeled data. To highlight the scalability of my method I compared against existing state-of-the-art semi-supervised learning algorithms using the ImageNet-1% and ImageNet-10% splits. In all of my experiments I use the same model, a frozen ViT-L backbone with a trainable MLP head following the example of Rothenberger and Diochnos. The backbone used is either SigLIP 2 or DINOv3 (whichever produced higher validation accuracy as a view for co-training) or both in the case of co-training algorithms. To further ensure a fair comparison all models are trained with the same batch size, for the same number of gradient steps, and limited to at most 48 hours of wall clock time on an RTX 4090 GPU.

For each dataset and labeled budget I include the performance of popular methods for semi-supervised learning from prior work. I will conduct not statistical comparison

Table 6.4: Performance of different methods compared on the ImageNet-1% and ImageNet-10% benchmarks. Self-supervised models used for semi-supervised learning appear on the upper half of the table. Models on the lower half of the table use labeled and unlabeled data during classification training. For all cross-validation experiments the mean testing accuracy as well as the 95% confidence interval are given. Results that allow rejection of the null hypothesis for both baselines are shown in **bold**.

Method	Model	ImageNet-1%	ImageNet-10%
EsViT [87]	Swin-B, W=14	69.1	74.4
MAE [59]	ViT-L	23.4	48.5
CLIP [109]	ViT-L	80.5	84.7
DINOv2 [100]	ViT-L	78.1	82.9
SimCLRv2 [33]	ResNet152-w2d	74.2	79.4
SwAV [23]	ResNet50-w4	53.9	70.2
SigLIP 2 [143]	ViT-L	82.5	85.3
DINOv3 [129]	ViT-L	80.7	84.9
UDA [156]	ResNet50	-	68.78
FixMatch [133]	ResNet50	-	71.46
MPL [105]	EfficientNet-B6-Wide	-	73.9
Semi-ViT + Self-training [21]	ViT-L	77.3	83.3
REACT [88]	ViT-L	81.6	85.1
DHO [70]	ViT-L	84.6	85.9
Co-Training [14]	ViT-L×2	84.1	85.5
Meta Co-Training [117]	ViT-L×2	86.13 ± 0.15	89.05 ± 0.11
RLGSSL [61]	ViT-L	81.94 ± 0.21	86.33 ± 0.05
PGPL - ours	ViT-L×2	**86.61 ± 0.13**	88.95 ± 0.09

to these methods. I only include them to give an informal illustration of how methods not inspired by RL perform in the same setting.

From the experiments shown in Table 6.2 both null hypotheses can be rejected for all four labeled budgets. The iNaturalist is a particularly difficult dataset, and it has the largest number of classes of all of the datasets tested. It is in this setting that PGPL has the greatest advantage over prior methods. In Table 6.1 and Table 6.3 I observed that some existing popular benchmark datasets had already become saturated with five examples per task. While the sample mean accuracy for PGPL for the one-shot experiments on CIFAR-10 and STL-10 are much higher than the baselines the null

hypothesis cannot be rejected for the MCT comparison in both cases. Beyond the one-shot case for these two datasets it should be noted that the accuracies of the models are very saturated and the assumption that they are normally distributed is unrealistic. This can be seen when some of the confidence intervals include 100% accuracy. Again it is in the more challenging case of CIFAR-100 with only 1, 3, and 5 labels that PGPL produces significantly more accurate models. These experiments highlight the effectiveness of PGPL in the most difficult settings for SSL.

Comparisons on ImageNet Similarly to the experiments performed above, 5-fold cross-validation is used to create a sample of model performances for PGPL as well as the MCT and RLGSSL baselines on ImageNet. To test the hypothesis that PGPL produces more accurate models than the MCT and RLGSSL baselines on ImageNet I conduct two hypothesis tests for each label budget. These are one-tailed Welch's t-tests as the variances of the tested populations are unequal. The null hypothesis is that the mean accuracy of models produced by the baseline being compared to is greater than or equal to the mean accuracy of those produced by PGPL. In each case two comparisons were performed, so I apply the Bonferonni correction to the typical 0.05 significance threshold to obtain the threshold $\alpha = 0.025$.

In Table 6.4 are given a several results from several prior works to illustrate state-of-the-art accuracy for this task. The last three rows of the table show the results from the cross-validation experiments conducted to compare PGPL to MCT and RLGSSL. From the results shown in Table 6.4 it can be observed that the null hypothesis can be rejected for the ImageNet-1% benchmark. On the ImageNet-10% benchmark PGPL produces significantly higher mean accuracy than RLGSSL, but not MCT. PGPL provides the largest benefit over prior methods in the more difficult setting for semi-supervised learning. In this experiment, with expanded access to labeled data its advantage

becomes less pronounced.

6.5 Conclusion

I proposed a novel method for learning pseudo-label assignment using reinforcement learning. PGPL overcomes key drawbacks of existing RL for SSL algorithms. PGPL estimates example-level rewards rather than batch-level rewards allowing it to differentiate between the quality of pseudo-labeled examples within a batch. Furthermore, PGPL estimates the reward of a pseudo-labeled example as its marginal contribution to the reduction in loss of a student model. PGPL highlights the value of pseudo-labels not as their proximity to the ground truth, or the extent to which they represent a heuristic of generalization, but rather as a tool for optimizing a model. What differentiates PGPL from prior SSL work is the explicit use of pseudo-labels outside of their traditional context as weak supervision. Encouragingly, PGPL provides the most advantage over prior work when it had access to the least supervision. The results of my experiments demonstrate that RL for SSL is an exciting avenue of research that has the potential to further reduce the need for large labeled datasets.

6.6 Future Work

In this paper I utilize a bare-bones method of RL, but many algorithms have been developed for RL. One exciting avenue for future investigation is incorporating insights from algorithms such as proximal policy optimization (PPO) [123] to enhance sample-efficiency or speed of convergence. PPO is just one example of the many algorithms for RL, and as the field continues to develop there are sure to be even more. I see integrating SSL and RL as a promising direction for future work.

Chapter 7

Explaining Image Classification with Localized Additive Explanations

7.1 Chapter Introduction

Machine learning has been a huge success story with machine learning algorithms' predictions widely used either directly or to inform human experts for downstream tasks; e.g., predicting recidivism [44], deciding on college admissions [35], and others. In this context it becomes essential to provide 'explanations' for machine learning models' predictions. Hence, an important pillar for 'trustworthy machine learning' is that of 'explainability;' a field that falls under the broader umbrella of *eXplainable Artificial Intelligence (XAI)* [6, 36, 20]. Providing explanations provides benefits including, but not limited to, promoting trust, allowing for accountability of algorithms, and causality that highlights input-output relationships. In this context, unsurprisingly, a lot of effort is devoted to create methods whereby predictions of machine learning models can be explained.

When the learned models are not interpretable by design, providing explanations can be a challenge. For example, linear combinations of features have been used widely, from perceptrons to logistic and linear regression. It is easy to understand such weighted sums and the influence of individual attributes to the prediction. However, when many such modules are combined to form artificial neural networks (ANNs), the input attributes contribute in delicate, difficult to express, ways to the final output. Hence, there is a trade-off between performant models and explainability [111]. LAX seeks to improve on prior work providing explanations to classification tasks made by ANNs.

Chapter Contributions Typical ANNs for image classification leverage convolutional or attention mechanisms to extract features by interleaving these mechanisms with downsampling steps and eventually producing a probability vector. I propose a fundamentally different way of solving the image classification task by forcing the model to produce an output of the same spatial dimension of the image. This output is used to create a pseudo-label whose probabilities are proportional to the area indicated for each class. Furthermore, if the indicated region for a class were to be removed then the pseudo-label will reflect this change by reducing its predicted probability for that class. In this way the proposed method creates explanations that represent a property of the pseudo-label.

A Localized Additive eXplanations (LAX) ANN has three outputs used to assess its performance:

- the probability vector output of a traditional ANN for classification,
- the output when the evidence the model identifies the predicted class of the image has been masked, and
- the output when all the model-identified class-relevant pixels are masked.

The first output is subject to supervision using the class labels. The other two outputs are designed to assess the ability of the model to segment the evidence it used to make its classification. Fundamentally ground-truth labels for the masks are unavailable, so the other two model outputs are used to achieve a good mask in a self-supervised manner. To identify class-relevant pixels the model effectively generates an 'occlusion mask' during its prediction step that serves as an explanation of the prediction. This explanation can be verified for correctness using only model outputs, and requires no external evaluation of the model. By incorporating the explanation in the prediction step, LAX is hundreds of times faster than comparable XAI methods.

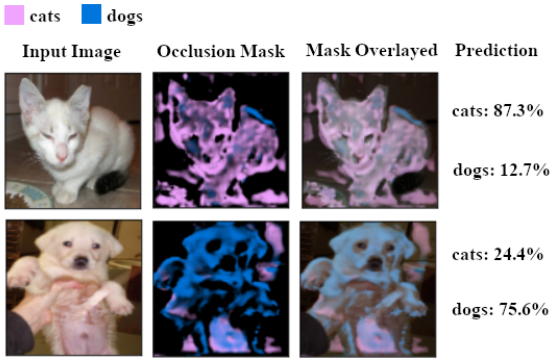


Figure 7.1: Examples from the Cats and Dogs dataset. The LAX explanation is shown in the middle column. The last column shows the explanation overlaid on top of the input image. The probability assigned by the model to each one of the two classes is shown on the right.

LAX provides explanations that are 'localized,' i.e., identifying the position in the image used for predicting a class, and 'additive,' e.g., in the examples shown in Figure 7.1 the area of the prediction mask belonging to a certain color is directly proportional to the probability of that class.

7.2 Related Work

Modern ANNs are used with increasing frequency in various applications due to their predictive accuracy. However, because the prediction rules of ANNs are very complicated and hard to express, proxy methods are needed for their explainability. Such methods can be general, model-agnostic, XAI methods that apply to any model space and thus to ANNs as well, or XAI methods that are tailored to specific architectures of ANNs. LAX falls under this latter spectrum.

Model-Agnostic XAI Methods Local Interpretable Model-Agnostic Explanations *(LIME)* generates a local explanation by training an interpretable model (e.g., LASSO regression or decision tree) using input-output pairs generated by the model in the neighborhood of an instance to be explained [112]. Note that the 'local' in LIME

refers to locality in the input space to the model, not locality within an image example. When classifying images, each image is divided into a set of disjoint contiguous regions, called 'super-pixels'. The LIME explanation is then the most influential super-pixels in the input, and optionally the order of their importance.

One line of work investigates metrics for calculating the influence of inputs (or features) to the predicted outputs [37, 13]. Lundberg and Lee [90] use the game-theoretic notion of the Shapley value, to provide Shapley Additive Explanations *(SHAP)*. Exact calculation of the Shapley value is costly, including several natural cases [38]. In fact, issues arise beyond computational complexity [80]. Shapley values for image classification are approximated by evaluating the effect of removing a square region in the image on the classification probabilities. Methods such as LIME and SHAP can be vulnerable to adversarial inputs [132], while Slack et al. [131] introduce a *Bayesian* version that attaches uncertainty levels to the provided explanations. Additional model agnostic methods include *Anchors* [113], *MAPLE* [106], and *prototypes* [29].

Rate Distortion XAI Methods Rate Distortion Explanations, first proposed in [91], attempt to find the minimal area in an input image responsible for the model's prediction. This is achieved by finding the maximum mask by area that can be applied to an input without altering the predicted class. Several RDE methods have been proposed in prior work such as CartoonX [77], WaveletX [78], and ShearletX [78]. These methods rely on optimizing a mask for each prediction for which an explanation is desired. As a consequence, these methods are interesting but too costly to be practical in scenarios where an explanation is required for every prediction.

ANN-Specific XAI Methods Zeiler and Fergus [168] present the first and possibly simplest method of visualizing the saliency of an input feature using their novel 'de-convnet' approach, where the idea is to identify the contribution of each feature in a

CNN, by following backwards the computation that was performed and allowed the prediction in the first place. The paper also presents another very simple method of evaluation wherein a gray patch is tiled across the image to occlude parts of the image and the change in the prediction is measured. I will refer to this method as 'Occlusion Sensitivity.' Simonyan et al. [128] leverage the gradient of the probability of predicting a class with respect to each input feature, leading to *saliency maps*. Roughly, saliency maps provide a gradient heatmap over the input pixels that justifies which features to which classification is most sensitive locally; see also [76, 145]. Other examples include Layer-wise Relevance Propagation [*LRP*, 7], Class Activation Mapping *(CAM)* and the closely related methods of *GRAD-CAM* and *Guided GRAD-CAM* [125], *interpretable CNNs* [174], *neural additive models (NAMs)* [2], *B-cos Networks* [15], and D-RISE for object detection [104], to name a few. Another direction of exploiting gradients is with the method of *integrated gradients* [137].

Shrikumar et al. [127] in *DeepLIFT*, explain the prediction of a particular input with respect to a reference input, by looking at the difference between these two inputs. Kim et al. [75] explain the internal state of an ANN in terms of human-friendly *concepts*, Dhurandhar et al. [41] propose *contrastive explanations*, while Brendel and Bethge [19] propose the use of *bags of local features*.

7.3 Method

The architecture shown in Figure 7.2 utilizes an image-to-image translation model, that is used to make three predictions. The image-to-image model in the figure is labeled 'U-net' referring to the style of image-to-image model presented by Ronneberger et al. [115], but in theory any sufficiently powerful image-to-image translation model could be used. A neural network that uses 'skip' connections inspired by the 'U-net' but that uses blocks similar to those in 'Focal Modulation Networks' [162] for higher resolutions

and 'Multi-Headed Self-Attention' [144] modules at lower resolution was used as the model for LAX. The input to the image-to-image model is in all cases an image tensor of size $W \times H \times ch$. The output in all cases is a mask tensor of size $W \times H \times (k+1)$. When the model achieves its objective well, slices of size one along the last axis of the output tensor correspond to masks that identify pixels that result in the model being able to predict a certain class. These slices of the output volume will be referred to as 'occlusion masks.' The last output occlusion mask identifies pixels that contribute to no class and are never dropped out. The probability vectors $\boldsymbol{p_0}, \boldsymbol{p_1}$, and $\boldsymbol{p_2}$, shown in Figure 7.2, are explained below.

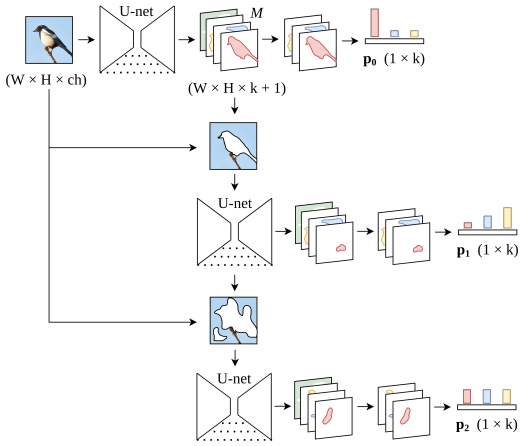


Figure 7.2: Model Architecture. Shown are three prediction steps of the model on a given image. In the first step the model is presented with the image, in the second step it is presented with the image with the area the model indicated for the predicted class masked out, and in the final step the model is presented with the image with the area the model indicated as evidence for all classes removed.

7.3.1 Training

One training step of the model requires three inferences of the image-to-image network. Each of these inferences produces an output that incurs a loss. The LAX training step is described in three parts below.

Part I Initially the model takes as input the image for classification and predicts the parts of the image, were they to be removed, would result in a decrease of the probability of a certain class for all classes. This follows because the same model parameters are used in each step, and in subsequent steps the model will be encouraged by the loss function to predict a region such that if it were removed the model would perform worse. The softmax function is applied across the channel dimension of each pixel in the prediction volume M sums to 1, so for pixels that are unlikely to change the prediction the model assigns weight to the final occlusion mask that corresponds to no class. The average value of each class occlusion mask for each image divided by the sum of the average values for all class occlusion masks for a that image is the predicted probability for a class for that image. This probability vector $\boldsymbol{p_0}$ is the first output of the model and the pseudo-label to which the explanation corresponds.

Part II The second inference of the model takes as input the image for classification with the area the model indicated for the predicted class masked out. The masking replaces pixels indicated as responsible for prediction with the pixel-wise *mean image* μ of the dataset. More precisely, given μ, the occlusion mask $M_{pred} \in \mathbb{R}^{W \times H}$ corresponding to the predicted class, and $x \in \mathbb{R}^{W \times H \times ch}$ the input image, the operation is:

$$x' = (M_{pred} \odot \mu) + ((1 - M_{pred}) \odot x)\,, \tag{7.1}$$

where the Hadamard product $\odot$ is broadcast where applicable. This image x' is input to the same image-to-image model as in the first step and produces $k + 1$ output occlusion masks. These occlusion masks are averaged, and the averages of the class occlusion masks are again re-normalized to produce a probability vector output. This probability vector $\boldsymbol{p_1}$ is the second output of the model.

Part III The third inference of the model takes as input the image for classification combined with the mean of the dataset such that all the pixels in the mask corresponding to any class from the first step are dropped out. Formally, given mask $M \in \mathbb{R}^{W \times H \times k+1}$ and $M_{sum} = \sum_{i=0}^{k} M_i$ we have

$$x'' = (M_{sum} \odot \mu) + ((1 - M_{sum}) \odot x)\,. \tag{7.2}$$

This image x'' is input to the same image-to-image model as in the first step and produces $k+1$ output occlusion masks. These occlusion masks are averaged, and the averages of the class occlusion masks are again re-normalized to produce a probability vector output. This probability vector $\boldsymbol{p_2}$ is the third output of the model.

Each of the outputs from the steps above incurs a loss. The first output $\boldsymbol{p_0}$ incurs cross-entropy loss $\mathcal{L}(f_\theta, \mathbf{L})$ based on its prediction compared to the true label of the instance. The second output $\boldsymbol{p_1}$ incurs a loss $\mathcal{L}_1(f_\theta, \mathbf{L})$ based on the similarity of $\boldsymbol{p_1}$ and $\boldsymbol{p_0}$. I define this loss as

$$\mathcal{L}_1(f_\theta, \mathbf{L}) = \frac{1}{n} \sum_{i=1}^{n} -\ln\left(1 - \langle \boldsymbol{p_0'}, \boldsymbol{p_1'} \rangle\right). \tag{7.3}$$

Where $\boldsymbol{p_0'}$ and $\boldsymbol{p_1'}$ are normalized versions of $\boldsymbol{p_1}$ and $\boldsymbol{p_0}$ that have Euclidean norm of 1. This loss function encourages the LAX model to predict differently in the second pass than it predicted in the first pass. This loss is designed to penalize the model for indicating an area for the predicted class that, when masked out, does not cause its prediction to differ from the pseudo-label. It is this loss that makes LAX PL as it uses the complement of its first pass pseudo-label as a target during training.

The last output $\boldsymbol{p_2}$ incurs a loss

$$\mathcal{L}_2(f_\theta, \mathbf{L}) = -\ln\left(\overline{p_2} / p_2^*\right), \tag{7.4}$$

where $\overline{p_2}$ is the mean of the $p_{2,i}$'s, and p_2^* is the max of the $p_{2,i}$'s. This loss penalizes the model for predicting confidently on the instance that has all of the regions predicted to be relevant for predicting any class masked out. The desired behavior this encourages is that the model should not be able to make a decisive prediction after all of the information it predicted would be relevant has been removed from the input.

All the above yield the following definition.

Definition 7.3.1 (LAX Loss)**.** For a given model f that is equipped with the LAX architecture and an input labeled image $(x, \boldsymbol{y})$, upon prediction, the model suffers loss:

$$\mathcal{L}_{\text{LAX}}\left(f, x, \boldsymbol{y}\right) = \mathcal{L}(f_\theta, \mathbf{L}) + \alpha\mathcal{L}_1(f_\theta, \mathbf{L}) + \beta\mathcal{L}_2(f_\theta, \mathbf{L})\,, \tag{7.5}$$

where α and β are hyperparameters, and x' and x'' are obtained from (7.1) and (7.2) respectively.

Intuition Intuitively, the first term encourages the model to mask the evidence essential to predicting a class, with the occlusion mask corresponding to the class by penalizing the model for adding probability mass to the incorrect classes' occlusion masks. In sequence, the second term, $\mathcal{L}_1$, encourages the model to mask nothing but the evidence essential for the predicted class, as this loss is the lowest when $\boldsymbol{p_1}$ is orthogonal to $\boldsymbol{p_0}$. Finally, $\mathcal{L}_2$ encourages the model to mask all of the evidence essential for any class, as the model suffers no loss for using the final occlusion mask belonging to no class, except when it would otherwise be able to predict any class using those pixels. Perhaps more intuitively, this loss encourages the model to predict uniformly and with maximal uncertainty after masking (e.g. the max and mean of the probability vectors are equal to $\frac{1}{k}$). So, all together, the model's objective can be understood as masking the evidence it is using within the input, all of the evidence essential for any

class, and nothing but the evidence. Note that qualify *evidence* is qualified with with *essential* as the masks need only remove essential signals for the objective function to be optimized. It would be quite fortunate if the model effectively segmented the entire object belonging to a class, and indeed this is one possible optimal solution, but it is not necessary to achieve minimum loss.

Naming The occlusion masks generated by the model identify spatial regions that contain signals essential to classification by design, because when they are removed the same model can no longer predict the correct class, these explanations are thus called "localized." Note that this does not imply that the pixels are logically related to the ground truth, simply that the model is "truthful" in identifying which pixels it is using to predict the predicted class. Furthermore, explanations are "additive" because the class probabilities are directly computed from the occlusion masks by adding up the predicted class probabilities for each pixel and dividing by the sum of class probabilities. Hence, the number of pixels covered by a mask for a class is directly proportional to the probability of that class. In the examples shown, the area of the prediction mask belonging to a certain color is directly proportional to the probability of that class.

7.3.2 Inference

Definition 7.3.2 (Accuracy)**.** Given a sample $S = \{(x_i, \boldsymbol{y_i})\}_{i=1}^{N}$ and a LAX model f predicting $\{(\boldsymbol{p_0^{(i)}}, \boldsymbol{p_1^{(i)}}, \boldsymbol{p_2^{(i)}})\}_{i=1}^{N}$ on S, f has accuracies in the three components $j \in \{0, 1, 2\}$:

$$A_j^{(f)} = 1 - \frac{1}{m} \sum_{i=1}^{m} \mathbb{1} \left\{ \text{argmax}_{\,1 \le q \le k} \left(p_j^{(i)} \right)_q \neq \text{argmax}_{\,1 \le r \le k} (y_i)_r \right\} ,$$

where $\mathbb{1}\{\mathcal{E}\} = 1$ if event $\mathcal{E}$ holds; otherwise, $\mathbb{1}\{\mathcal{E}\} = 0$.

During training a model f is optimized to achieve its objectives well outside of simply prediction accuracy. If this is the case and f generalizes well to unseen data, then we might reasonably be sure that on a given instance it is indeed explaining its prediction satisfactorily. The degree to which f achieves these objectives can be measured by observing the accuracies $A_0^{(f)}, A_1^{(f)}$, and $A_2^{(f)}$, corresponding to the outputs $\boldsymbol{p_0}$, $\boldsymbol{p_1}$, and $\boldsymbol{p_2}$. Ideally, we should have $A_0^{(f)} = 100\%$, $A_1^{(f)} = 0\%$, and $A_2^{(f)} = 1/k$. If we are happy to accept that its predictions, as well as its explanations, are likely reasonable based on test set performance, then we can discard all of the masking steps and simply use the first inference of the image-to-image model. In practice this is desirable as the computational cost of inference with explanation is low. However, we are not at the mercy of aggregate performance statistics when evaluating the quality of a particular explanation. If we retain all three of the prediction steps during inference, we can measure how the predicted label changes between $\boldsymbol{p_0}$, $\boldsymbol{p_1}$, and $\boldsymbol{p_2}$. If the argmax of $\boldsymbol{p_0}$ and $\boldsymbol{p_1}$ differ and $\boldsymbol{p_2}$ has a lower maximum value than $\boldsymbol{p_0}$, then we know that the model has achieved its secondary objectives well on that instance. We revisit this discussion in Section 7.4 but we are now ready to define the *model of best occlusion*.

Definition 7.3.3 (Occlusion Quality)**.** For a LAX model f, with output accuracies $A_0^{(f)}, A_1^{(f)}$, and $A_2^{(f)}$, *occlusion quality* is defined to be $Q_{\text{LAX}}(f) = A_0^{(f)} - \left|A_2^{(f)} - 1/k\right|$.

Definition 7.3.4 (Model of Best Occlusion)**.** Let $\mathcal{F}_{\text{LAX}}$ be a set of different LAX models. The *model of best occlusion* $f^* \in \mathcal{F}_{\text{LAX}}$ is defined as $f^* \in \text{argmax}_{f \in \mathcal{F}_{\text{LAX}}} Q_{\text{LAX}}(f)$.

7.3.3 Trainable Components

The only trainable portion of this architecture is the single image-to-image model labeled 'U-net' in Figure 7.2. Each U-net in the diagram is identical, i.e., they all share the same weights. During training this is advantageous as the model weights can be

stored only once. During the backward pass, the gradient of the weights of the U-net in LAX are propagated through five times. The first time is a consequence of the loss suffered due to $\mathcal{L}$. Then the loss suffered from $\mathcal{L}_1$ and $\mathcal{L}_2$ are propagated through the entire path through the execution graph from output to the input image; thus, twice each through the weights of the U-net. The propagation of the gradient is *not stopped* after the nearest application of the U-net to the output, as this seemed to be a more successful approach than the alternative in preliminary investigations. Further analysis is necessary to determine the effect of that potential stop.

7.4 Experimental Evaluation

7.4.1 Setup for Experiments

The LAX models are evaluated on a three-way split: 70% of the data is used for training, 10% is reserved for validation, and 20% is reserved for testing. Validation data was used to tune both the model's hyperparameters as well as the hyperparameters associated with data augmentation and training. As with many deep learning experiments the choice of value for the batch size, the choice of an optimizer, and the choice of data augmentation strategy are non-trivial. Testing data is evaluated only once hyperparameter selection has completed for each dataset. Only testing and validation set images are displayed in the figures.

Baselines LAX models were compared to two different models and five total comparable XAI methods for explaining image classification predictions. An Interpretable CNN [174] was trained to provide an ante hoc point of comparison. A standard DenseNet model [64] was also trained to apply post-hoc explanation methods to. LIME, SHAP, Gradient (Saliency Map), and Occlusion Sensitivity methods were applied as 'post hoc' explanation baselines. The 'interpretable' layers of the Interpretable CNN were

added on top of DenseNet121 for the sake of a fair comparison between the ante hoc Interpretable CNN baseline and the post hoc baselines.

7.4.2 Datasets

For evaluating LAX we use the Oxford Pets or "cats and dogs" dataset [103]. While this is not a challenging classification task we believe that it is useful for evaluating explainability methods. Because the source of the ground truth within the image is clear, and humans can easily discriminate between cats and dogs it should also be easy for humans to interpret the quality of explanations. Table 7.1 provides details on the class composition and number of elements in the cats and dogs dataset. Data augmentations were performed "on the fly" and for this reason the mean image was estimated μ using the images that are used in each batch. A small amount of Gaussian noise is also added to each position in μ.

On the Nature of Explanations Cats and dogs have a clear notion of contiguous 'objectness.' Signals that are discriminative for classification are localized in discrete blob-like structures. While it can be tempting to dismiss an explanation because it does not conform to the human understanding of the cat or dog object within the scene, in such cases this is a feature and not a bug. As LAX will reveal, neural networks can pick out extraneous features that may be highly correlated with the class instead of the ground-truth signal. While this may rightfully diminish trust in the model, identifying when the model is focusing on the wrong parts of an image when making its prediction is important. It is critical that LAX can be used to identify when these explanations are truthful, i.e. whether masking out this information alters the model's prediction.

The models achieving the highest accuracy and the highest occlusion quality (Q_{LAX}) were both found with $\alpha = \beta = 2^{-10}$. Table 7.2 shows accuracy results on the Cats

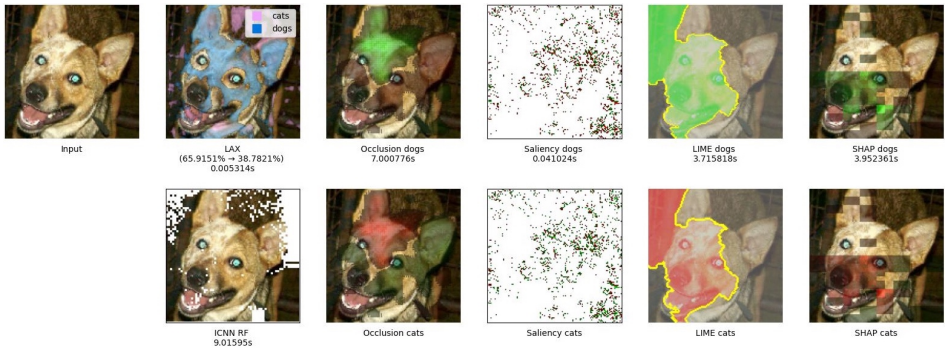
Table 7.1: Class frequencies for Cats and Dogs.

class	train instances (%)	validation instances (%)	testing instances (%)
cats	8195 (50.3%)	1183 (50.8%)	2280 (49.0%)
dogs	8088 (49.7%)	1144 (49.2%)	2372 (51.0%)

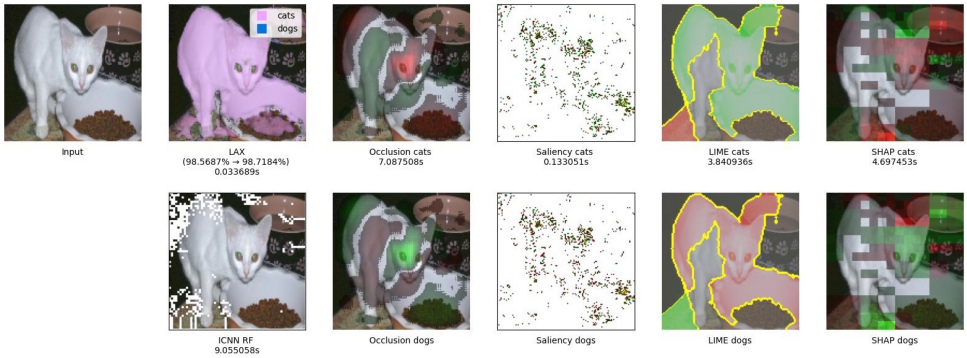
Table 7.2: Evaluation results. The testing accuracy of the LAX model is compared to its accuracy after the pixels identified as relevant have been masked. Recall that the best accuracy LAX model is the one that has the highest accuracy using $\boldsymbol{p_0}$ for prediction, while the best occluded LAX model is the one that maximizes Q_{LAX} (see Definition 7.3.4) and still uses $\boldsymbol{p_0}$ for prediction. Hence, the accuracy after masking the relevant evidence is equivalent to the accuracy if $\boldsymbol{p_2}$ were used for prediction by either model. For $\boldsymbol{p_2}$ accuracy, closer to $\frac{1}{k}$ is better where applicable. DenseNet and Interpretable DenseNet models do not have a notion of $\boldsymbol{p_2}$ accuracy, so only their test set accuracy is displayed.

model description	testing accuracy	$\boldsymbol{p_2}$ testing accuracy
LAX (best accuracy)	88.18%	60.73%
LAX (best occluded)	84.89%	56.58%
DenseNet	89.77%	n/a
Interpretable DenseNet	**90.60%**	n/a

and Dogs dataset (best accuracy and best occluded), the DenseNet baseline, and the Interpretable CNN baseline. Figure 7.3a presents an effective occlusion map, where we can see that by masking out the evidence, the probability of selecting a particular class changes enough to impact the classification. This is not true in Figure 7.3b where the occlusion map appears to be a satisfactory explanation, but it does not meaningfully effect the predicted probability. These examples highlight the usefulness of the additional forward passes (i.e., verifying $\text{argmax}(\boldsymbol{p_0}) = \text{argmax}(\boldsymbol{p_2})$) to determine the fitness of an explanation, as one could be persuaded to accept the second explanation, but it turns out not to correspond to an effective occlusion map.



(a) A successful occlusion from the Cats and Dogs task. The model is predicting the correct label and the occlusion changes the predicted class. The numbers below 'LAX' on the figure represents the percent change in the predicted class (dogs) following the occlusion.



(b) An unsuccessful occlusion from the Cats and Dogs task. The model is predicting the correct label, but the occlusion doesn't significantly impact the predicted class probability. The numbers below 'LAX' on the figure represents the percent change in the predicted class (cats) following the occlusion.

Figure 7.3: Successful and unsuccessful occlusions from the Cats and Dogs task. Images are ordered top to bottom by their descending prediction probabilities. The first column contains the input image. The second column contains the output of the LAX model and the 'Receptive Field' of the Interpretable DenseNet. The last four columns are XAI methods (Occlusion Sensitivity, Saliency Maps, LIME, SHAP in that order) applied to DenseNet. Time in seconds to produce the explanation for the given instance is included for each method.

7.5 Discussion

Advantages Methods based on saliency maps suffer from: (I) highly overlapping information with low spatial frequency in the case of GRAD-CAM and Interpretable

Table 7.3: Compute time comparison for all methods at 128 by 128 resolution on a batch of size 32 examples. Times recorded on a machine with one Nvidia Ampere A100 GPU.

method	forward passes per example	runtime (minutes)
Saliency Maps	**1**	0.028
LAX (ours)	3	**0.002**
LIME	2048	1.94
SHAP	2048	2.04
Occlusion Sensitivity	4096	3.36
Interpretable CNN	4096	4.70

CNNs, or (II) very high spatial frequency information that highlights individual pixels in what frequently appears in a random/noisy way [74], or (III) thousands of inferences are needed in order to generate an explanation. Case I is undesirable because it is hard to see which features in the image are being used as discriminative evidence for a specific class, and it is hard to compare the explanations for different classes. Along similar lines, Rudin [121] argues that 'saliency does not explain anything except where the network is looking' providing a picture of a dog and saliency maps that look almost identical when predicting a dog or a musical instrument. While this can be a very valid point in prior work, in LAX such pathological cases cannot arise as a unique saliency map where pixels are color-coded accordingly for the various classes is provided. Case II is also undesirable because of the potentially fragmented explanations that do not take into account neighboring pixels. While LAX, in principle, allows such situations to arise, nevertheless, no such case was observed in these experiments. Case III corresponds to many available model-agnostic methods (SHAP, LIME, Interpretable CNNs, Occlusion Sensitivity) that require thousands of inferences to generate their explanation. These methods are computationally expensive to approximate, and none has an existing implementation for image classification that achieves high granularity and low latency.

LAX is an order of magnitude faster than any of the baselines, and several orders of magnitude faster than some. Table 7.3 presents a runtime comparison of the explanation methods.[1] LAX is agnostic to the independence, location, or contiguity of essential signals (unlike SHAP, LIME, and Occlusion Sensitivity), and LAX only relies on at most three different forward passes and provides the full and final explanation.

Limitations LAX is architecture-dependent. Similar to Interpretable CNNs, the output of the ANN used for classification must have a special structure. The underlying image-to-image model can vary. Not unexpectedly LAX suffers a mild performance detriment in terms of accuracy over the baselines when the occlusion quality is high. It is unclear if LAX is applicable outside of image classification tasks. Finally, methods like GRAD-CAM and saliency maps provide an exact computation of an approximate explanation; e.g., they can exactly compute some property that approximately explains a prediction. LAX provides exact computation of a *probable* explanation.

7.6 Conclusion

I proposed a novel approach, *localized additive eXplanations* (LAX), for explaining image classification predictions. The approach relies on the application of three different losses that encourage the formation of occlusion masks that cover important features responsible for classification. Experiments indicated that the proposed architecture performs well regardless of whether the explanations are expected to be contiguous segments of images or not. Moreover, the trained architecture provides explanation much faster compared to baselines, which could be critical for real-time applications, while at the same time providing qualitative results that are probably better than both

[1] the official SHAP and LIME implementations in python that are available at `https://shap.readthedocs.io/en/latest/` and `https://lime-ml.readthedocs.io/en/latest/` were used respectively.

methods.

Related to LAX is the idea of *weakly supervised object localization (WSOL)* in which the goal is to localize an object within an image without having a ground truth label for its location. If one could identify the location of the dog in an image labeled 'dog' then that location might be a compelling explanation for the classification. Unfortunately, as we saw, models are not always using ground-truth-relevant signals to make classifications. It is precisely this difference between the goal of LAX and the goal of WSOL that allows LAX to make truthful explanations. The goal of LAX is not to produce a bounding box or segment that captures a particular object within an image. On some level that would require LAX to be making a correct prediction in order to provide a truthful explanation. LAX is designed to identify the regions of an image that are essential to the creation of its pseudo-label. The explanation provided by LAX is a property of the pseudo-label it assigns to an instance, not a property of the instance itself.

Chapter 8

Conclusion

In this dissertation I have presented a sequence of novel PL algorithms together with a review of the broader PL landscape. I use these contributions support my argument that pseudo-labels can be more than just weak supervision. Conventionally, pseudo-labels are treated as worse versions of human annotations. Even within SSL, they have been used in many more varied and interesting ways. The definition and taxonomy given in Chapter 4 laid the groundwork for conceptualizing the use of PL in three new algorithms. From multi-view meta-learning to RL and even XAI I have shown how pseudo-labels can be a powerful tool for enabling new optimization pathways for machine learning models.

Under my definition, the supervisory role of pseudo-labels is incidental rather than essential. What is important is that a model made the prediction, and thus can be optimized such that the prediction has different properties. These properties can be as related to weak supervision as simply being more effective supervision for a student, or they can be as unrelated as the pseudo-labels used in DINO that simply enforce consistency. I hope that this fresh perspective can be used to apply PL to areas outside of SSL and to enable machine learning to continue to solve new and interesting problems.

The novel algorithms in this dissertation embody this broader view of PL. In Chapter 5 I presented MCT, in which PL was used to pass knowledge from a teacher to a student. Because the pseudo-labels were model outputs, we could then optimize the teacher to assign pseudo-labels in a way that would allow the student to continue learning. In Chapter 6 I improved upon this idea by approximating RL rewards at the example-level to improve the quality of the teacher's feedback signal. I was able to connect current and prior works to this RL setting to identify ways in which to more

effectively use pseudo-labels. In Chapter 7 I pushed my definition of PL even further by applying it to an area outside of SSL. LAX uses pseudo-labels to optimize a classifier to produce a truthful explanation. Together these algorithms demonstrate that PL is best understood as a general technique for inducing structure using a model's categorical predictions. When viewed this way, PL is a versatile tool that can be applied creatively inside and outside of SSL. I hope that the formal definition, the taxonomy, and the novel algorithms presented in this dissertation give the reader both the conceptual vocabulary and the concrete examples needed to recognize PL and when it can be useful.

8.1 Future Work

The directions I find most promising follow naturally from the broader view of PL articulated above.

Reinforcement Learning for Pseudo-Labeling PGPL makes use of a very restricted setting for RL. I spent a good deal of Chapter 6 showing that the RL setting identified by prior work reduced to something quite simple. RL is not really needed to express the final objective, even if it is an informative way to understand why some PL methods work. Incorporating multi-step trajectories alongside more sophisticated RL algorithms is a natural next step for this work. I see this as the most exciting open direction for PL in SSL.

Beyond Class-Balanced Benchmarks All of the benchmark tasks that were used for evaluation had class-balanced labeled and unlabeled datasets as well as unlabeled distributions that matched the labeled distributions. In standard benchmark classification dataset scenarios the class distribution of the unlabeled data cannot be known. Further-

more, as was seen in Section 5.4.6 the unlabeled data may contain out-of-distribution instances that correspond to no class represented in the labeled data. Because by nature the unlabeled data are too numerous for humans to inspect, then it cannot be observed by those humans what conditions might be present in those unlabeled data. It may be possible, however, to perform a heuristic balancing or filtering to detect examples that are likely to be out-of-distribution or to downsample unlabeled majority classes. Increasingly researchers have access to very large datasets that have been collected in a completely uncurated way. I see this as a promising direction for applying SSL methods that benchmark well to real and messy data.

Beyond Classification All of the tasks solved in this dissertation were image classification tasks. Image classification is certainly an important task in CV, but it is far from the only one that requires categorical output. PL is usually applied to this area, but object detection and segmentation are areas in which labels are even more scarce and take even more time to create. If PL methods for SSL could improve performance of these models under label scarcity that would be a very practical improvement. Novel class discovery and anomaly detection are also areas for which our fuzzy-partition definition seems naturally suited. Cases in which new instances are encountered that do not belong to any known class seem like a natural next area to apply PL. The problems faced by algorithms for novel class discovery and anomaly detection seem highly related to the problems that SSL algorithms encounter when working with real unlabeled data as we saw in Section 5.4.6.

It is my intention that this dissertation should serve as an example of what things are possible when applying PL. I hope that the most interesting applications of PL have not yet been discovered and that there are many interesting directions for future work that I have not yet considered.

Reference List

[1] Abdelraouf, A., M. Abdel-Aty, and Y. Wu, 2022: Using vision transformers for spatial-context-aware rain and road surface condition detection on freeways. *IEEE transactions on intelligent transportation systems (Print)*, https://doi.org/10.1109/tits.2022.3150715.

[2] Agarwal, R., L. Melnick, N. Frosst, X. Zhang, B. J. Lengerich, R. Caruana, and G. E. Hinton, 2021: Neural Additive Models: Interpretable Machine Learning with Neural Nets. *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, M. Ranzato, A. Beygelzimer, Y. N. Dauphin, P. Liang, and J. W. Vaughan, Eds., 4699–4711, URL https://proceedings.neurips.cc/paper/2021/hash/251bd0442dfcc53b5a761e050f8022b8-Abstract.html.

[3] Angluin, D., and P. D. Laird, 1987: Learning From Noisy Examples. *Mach. Learn.*, **2 (4)**, 343–370, https://doi.org/10.1007/BF00116829, URL https://doi.org/10.1007/BF00116829.

[4] Arazo, E., D. Ortego, P. Albert, N. E. O'Connor, and K. McGuinness, 2020: Pseudo-Labeling and Confirmation Bias in Deep Semi-Supervised Learning. *IJCNN*, IEEE, 1–8.

[5] Arazo, E., D. Ortego, P. Albert, N. E. O'Connor, and K. McGuinness, 2020: Pseudo-Labeling and Confirmation Bias in Deep Semi-Supervised Learning. *2020 International Joint Conference on Neural Networks, IJCNN 2020, Glasgow, United Kingdom, July 19-24, 2020*, IEEE, , 1–8, https://doi.org/10.1109/IJCNN48605.2020.9207304, URL https://doi.org/10.1109/IJCNN48605.2020.9207304.

[6] Arrieta, A. B., and Coauthors, 2020: Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI. *Inf. Fusion*, **58**, 82–115, https://doi.org/10.1016/j.inffus.2019.12.012, URL https://doi.org/10.1016/j.inffus.2019.12.012.

[7] Bach, S., A. Binder, G. Montavon, F. Klauschen, K.-R. Müller, and W. Samek, 2015: On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PloS one*, **10 (7)**, e0130 140.

[8] Balestriero, R., and Coauthors, 2023: A Cookbook of Self-Supervised Learning. *CoRR*, **abs/2304.12210**, 71, https://doi.org/10.48550/ARXIV.2304.12210, URL https://doi.org/10.48550/arXiv.2304.12210, 2304.12210.

[9] Barcina-Blanco, M., J. L. Lobo, P. Garcia-Bringas, and J. D. Ser, 2024: Managing the unknown: a survey on open set recognition and tangential areas. URL

https://arxiv.org/abs/2312.08785, 2312.08785.

[10] Bengio, Y., J. Louradour, R. Collobert, and J. Weston, 2009: Curriculum learning. *Proceedings of the 26th Annual International Conference on Machine Learning, ICML 2009, Montreal, Quebec, Canada, June 14-18, 2009*, A. P. Danyluk, L. Bottou, and M. L. Littman, Eds., ACM, , ACM International Conference Proceeding Series, Vol. 382, 41–48, https://doi.org/10.1145/1553374.1553380, URL https://doi.org/10.1145/1553374.1553380.

[11] Berthelot, D., N. Carlini, E. D. Cubuk, A. Kurakin, K. Sohn, H. Zhang, and C. Raffel, 2019: ReMixMatch: Semi-Supervised Learning with Distribution Alignment and Augmentation Anchoring. *CoRR*, **abs/1911.09785**, URL http://arxiv.org/abs/1911.09785, 1911.09785.

[12] Berthelot, D., N. Carlini, I. J. Goodfellow, N. Papernot, A. Oliver, and C. Raffel, 2019: MixMatch: A Holistic Approach to Semi-Supervised Learning. *NeurIPS*, 5050–5060, URL https://proceedings.neurips.cc/paper/2019/hash/1cd138d0499a68f4bb72bee04bbec2d7-Abstract.html.

[13] Bhatt, U., A. Weller, and J. M. F. Moura, 2020: Evaluating and Aggregating Feature-based Model Explanations. *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI 2020*, C. Bessiere, Ed., ijcai.org, 3016–3022, https://doi.org/10.24963/ijcai.2020/417, URL https://doi.org/10.24963/ijcai.2020/417.

[14] Blum, A., and T. M. Mitchell, 1998: Combining Labeled and Unlabeled Data with Co-Training. *Proceedings of the Eleventh Annual Conference on Computational Learning Theory, COLT 1998, Madison, Wisconsin, USA, July 24-26, 1998*, P. L. Bartlett, and Y. Mansour, Eds., ACM, 92–100, https://doi.org/10.1145/279943.279962, URL https://doi.org/10.1145/279943.279962.

[15] Böhle, M., M. Fritz, and B. Schiele, 2022: B-cos Networks: Alignment is All We Need for Interpretability. *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022*, IEEE, 10 319–10 328, https://doi.org/10.1109/CVPR52688.2022.01008, URL https://doi.org/10.1109/CVPR52688.2022.01008.

[16] Bossard, L., M. Guillaumin, and L. V. Gool, 2014: Food-101 - mining discriminative components with random forests. *Computer Vision - ECCV 2014 - 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part VI*, D. J. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, Eds., Springer, Lecture Notes in Computer Science, Vol. 8694, 446–461, https://doi.org/10.1007/978-3-319-10599-4_29, URL https://doi.org/10.1007/978-3-319-10599-4_29.

[17] Brefeld, U., C. Büscher, and T. Scheffer, 2005: Multi-view Discriminative Sequential Learning. *Machine Learning: ECML 2005, 16th European Conference on Machine Learning, Porto, Portugal, October 3-7, 2005, Proceedings*, J. Gama, R. Camacho, P. Brazdil, A. Jorge, and L. Torgo, Eds., Springer, Lecture Notes in Computer Science, Vol. 3720, 60–71, https://doi.org/10.1007/11564096_11, URL https://doi.org/10.1007/11564096_11.

[18] Brefeld, U., and T. Scheffer, 2004: Co-EM support vector learning. *Machine Learning, Proceedings of the Twenty-first International Conference (ICML 2004), Banff, Alberta, Canada, July 4-8, 2004*, C. E. Brodley, Ed., ACM, ACM International Conference Proceeding Series, Vol. 69, https://doi.org/10.1145/1015330.1015350, URL https://doi.org/10.1145/1015330.1015350.

[19] Brendel, W., and M. Bethge, 2019: Approximating CNNs with Bag-of-local-Features models works surprisingly well on ImageNet. *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*, OpenReview.net, URL https://openreview.net/forum?id=SkfMWhAqYQ.

[20] Burkart, N., and M. F. Huber, 2021: A survey on the explainability of supervised machine learning. *J. Artif. Intell. Res.*, **70**, 245–317, https://doi.org/10.1613/jair.1.12228, URL https://doi.org/10.1613/jair.1.12228.

[21] Cai, Z., and et al., 2022: Semi-supervised Vision Transformers at Scale. *NeurIPS*.

[22] Caron, M., P. Bojanowski, A. Joulin, and M. Douze, 2018: Deep Clustering for Unsupervised Learning of Visual Features. *Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part XIV*, V. Ferrari, M. Hebert, C. Sminchisescu, and Y. Weiss, Eds., Springer, , Lecture Notes in Computer Science, Vol. 11218, 139–156, https://doi.org/10.1007/978-3-030-01264-9_9, URL https://doi.org/10.1007/978-3-030-01264-9_9.

[23] Caron, M., I. Misra, J. Mairal, P. Goyal, P. Bojanowski, and A. Joulin, 2020: Unsupervised Learning of Visual Features by Contrasting Cluster Assignments. *NeurIPS*.

[24] Caron, M., H. Touvron, I. Misra, H. Jégou, J. Mairal, P. Bojanowski, and A. Joulin, 2021: Emerging Properties in Self-Supervised Vision Transformers. *2021 IEEE/CVF International Conference on Computer Vision, ICCV 2021, Montreal, QC, Canada, October 10-17, 2021*, IEEE, , 9630–9640, https://doi.org/10.1109/ICCV48922.2021.00951, URL https://doi.org/10.1109/ICCV48922.2021.00951.

[25] Carrillo, J., and M. Crowley, 2020: Integration of roadside camera images and

weather data for monitoring winter road surface conditions. 2009.12165.

[26] Carrillo, J., M. Crowley, G. Pan, and L. Fu, 2020: Design of efficient deep learning models for determining road surface condition from roadside camera images and weather data. 2009.10282.

[27] Cascante-Bonilla, P., F. Tan, Y. Qi, and V. Ordonez, 2020: Curriculum Labeling: Self-paced Pseudo-Labeling for Semi-Supervised Learning. *CoRR*, **abs/2001.06001**, 13, URL https://arxiv.org/abs/2001.06001, 2001.06001.

[28] Chapelle, O., B. Schölkopf, and A. Zien, 2006: *Semi-Supervised Learning*. MIT Press.

[29] Chen, C., O. Li, D. Tao, A. Barnett, C. Rudin, and J. Su, 2019: This Looks Like That: Deep Learning for Interpretable Image Recognition. *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, H. M. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. B. Fox, and R. Garnett, Eds., 8928–8939, URL https://proceedings.neurips.cc/paper/2019/hash/adf7ee2dcf142b0e11888e72b43fcb75-Abstract.html.

[30] Chen, M., Y. Du, Y. Zhang, S. Qian, and C. Wang, 2022: Semi-supervised Learning with Multi-Head Co-Training. *Thirty-Sixth AAAI Conference on Artificial Intelligence, AAAI 2022, Thirty-Fourth Conference on Innovative Applications of Artificial Intelligence, IAAI 2022, The Twelveth Symposium on Educational Advances in Artificial Intelligence, EAAI 2022 Virtual Event, February 22 - March 1, 2022*, AAAI Press, , 6278–6286, https://doi.org/10.1609/AAAI.V36I6.20577, URL https://doi.org/10.1609/aaai.v36i6.20577.

[31] Chen, M., K. Q. Weinberger, and Y. Chen, 2011: Automatic Feature Decomposition for Single View Co-training. *ICML*, Omnipress, 953–960.

[32] Chen, T., S. Kornblith, M. Norouzi, and G. E. Hinton, 2020: A Simple Framework for Contrastive Learning of Visual Representations. *ICML*, PMLR, 1597–1607.

[33] Chen, T., S. Kornblith, K. Swersky, M. Norouzi, and G. E. Hinton, 2020: Big Self-Supervised Models are Strong Semi-Supervised Learners. *NeurIPS*.

[34] Coates, A., A. Y. Ng, and H. Lee, 2011: An analysis of single-layer networks in unsupervised feature learning. *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics, AISTATS 2011, Fort Lauderdale, USA, April 11-13, 2011*, G. J. Gordon, D. B. Dunson, and M. Dudík, Eds., JMLR.org, JMLR Proceedings, Vol. 15, 215–223, URL

http://proceedings.mlr.press/v15/coates11a/coates11a.pdf.

[35] D'Amour, A., H. Srinivasan, J. Atwood, P. Baljekar, D. Sculley, and Y. Halpern, 2020: Fairness Is Not Static: Deeper Understanding of Long Term Fairness via Simulation Studies. *FAT* '20: Conference on Fairness, Accountability, and Transparency, Barcelona, Spain, January 27-30, 2020*, M. Hildebrandt, C. Castillo, L. E. Celis, S. Ruggieri, L. Taylor, and G. Zanfir-Fortuna, Eds., ACM, 525–534, https://doi.org/10.1145/3351095.3372878, URL https://doi.org/10.1145/3351095.3372878.

[36] Das, A., and P. Rad, 2020: Opportunities and Challenges in Explainable Artificial Intelligence (XAI): A Survey. *CoRR*, **abs/2006.11371**, URL https://arxiv.org/abs/2006.11371, 2006.11371.

[37] Datta, A., S. Sen, and Y. Zick, 2016: Algorithmic Transparency via Quantitative Input Influence: Theory and Experiments with Learning Systems. *IEEE Symposium on Security and Privacy, SP 2016, San Jose, CA, USA, May 22-26, 2016*, IEEE Computer Society, 598–617, https://doi.org/10.1109/SP.2016.42, URL https://doi.org/10.1109/SP.2016.42.

[38] den Broeck, G. V., A. Lykov, M. Schleich, and D. Suciu, 2022: On the Tractability of SHAP Explanations. *J. Artif. Intell. Res.*, **74**, 851–886, https://doi.org/10.1613/jair.1.13283, URL https://doi.org/10.1613/jair.1.13283.

[39] Deng, J., W. Dong, R. Socher, L. Li, K. Li, and L. Fei-Fei, 2009: ImageNet: A large-scale hierarchical image database. *CVPR*, IEEE, 248–255.

[40] Devlin, J., M. Chang, K. Lee, and K. Toutanova, 2019: BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *NAACL-HLT*, ACL, 4171–4186.

[41] Dhurandhar, A., P. Chen, R. Luss, C. Tu, P. Ting, K. Shanmugam, and P. Das, 2018: Explanations based on the Missing: Towards Contrastive Explanations with Pertinent Negatives. *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, S. Bengio, H. M. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, Eds., 590–601, URL https://proceedings.neurips.cc/paper/2018/hash/c5ff2543b53f4cc0ad3819a36752467b-Abstract.html.

[42] Di, W., and M. M. Crawford, 2012: View Generation for Multiview Maximum Disagreement Based Active Learning for Hyperspectral Image Classification. *IEEE Trans. Geosci. Remote. Sens.*, **50 (5-2)**, 1942–1954.

[43] Doersch, C., A. Gupta, and A. A. Efros, 2015: Unsupervised visual representation learning by context prediction. *CoRR*, **abs/1505.05192**, URL http://arxiv.org/abs/1505.05192, 1505.05192.

[44] Dressel, J., and H. Farid, 2018: The accuracy, fairness, and limits of predicting recidivism. *Science advances*, **4 (1)**, eaao5580.

[45] Duval, Q., I. Misra, and N. Ballas, 2023: A Simple Recipe for Competitive Low-compute Self supervised Vision Models. *CoRR*, **abs/2301.09451**, 15, https://doi.org/10.48550/ARXIV.2301.09451, URL https://doi.org/10.48550/arXiv.2301.09451, 2301.09451.

[46] Evans, D. A., K. J. Sulia, N. P. Bassill, C. D. Thorncroft, J. C. Rothenberger, and L. C. Gaudet, 2025: Predicting forecast error for the hrrr using lstm neural networks: A comparative study using new york and oklahoma state mesonets. URL https://arxiv.org/abs/2512.14898, 2512.14898.

[47] Ferrera, V. A., J. C. Rothenberger, M. Wilson Reyes, C. Sutter, A. H. Fagg, and D. Diochnos, 2023: Classifying Road Surface Conditions with Self-Trained Artificial Intelligence. *103rd Annual AMS Meeting 2023*, American Meteorological Society Meeting Abstracts, Vol. 103, 15B.6.

[48] Finn, C., P. Abbeel, and S. Levine, 2017: Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks. *ICML*, 1126–1135.

[49] Frénay, B., and M. Verleysen, 2014: Classification in the Presence of Label Noise: A Survey. *IEEE Trans. Neural Networks Learn. Syst.*, **25 (5)**, 845–869, https://doi.org/10.1109/TNNLS.2013.2292894, URL https://doi.org/10.1109/TNNLS.2013.2292894.

[50] Gao, B.-B., C. Xing, C.-W. Xie, J. Wu, and X. Geng, 2017: Deep Label Distribution Learning with Label Ambiguity. *IEEE Transactions on Image Processing*, **26 (6)**, 2825–2838, https://doi.org/10.1109/TIP.2017.2689998, 1611.01731.

[51] Gidaris, S., P. Singh, and N. Komodakis, 2018: Unsupervised representation learning by predicting image rotations. *CoRR*, **abs/1803.07728**, URL http://arxiv.org/abs/1803.07728, 1803.07728.

[52] Goldman, S. A., and Y. Zhou, 2000: Enhancing Supervised Learning with Unlabeled Data. *Proceedings of the Seventeenth International Conference on Machine Learning (ICML 2000), Stanford University, Stanford, CA, USA, June 29 - July 2, 2000*, P. Langley, Ed., Morgan Kaufmann, , 327–334.

[53] Google, n.d.: [google maps driving directions from augusta, ga to atlanta, ga]. URL https://goo.gl/maps/X7nRGbrVm7fsv5Yd8, retrieved April 29, 2026.

[54] Gou, J., B. Yu, S. J. Maybank, and D. Tao, 2021: Knowledge Distillation: A Survey. *Int. J. Comput. Vis.*, **129 (6)**, 1789–1819, https://doi.org/10.1007/S11263-021-01453-Z, URL https://doi.org/10.1007/s11263-021-01453-z.

[55] Grill, J., and et al., 2020: Bootstrap Your Own Latent - A New Approach to Self-Supervised Learning. *NeurIPS*.

[56] Haase-Schütz, C., R. Stal, H. Hertlein, and B. Sick, 2020: Iterative Label Improvement: Robust Training by Confidence Based Filtering and Dataset Partitioning. *25th International Conference on Pattern Recognition, ICPR 2020, Virtual Event / Milan, Italy, January 10-15, 2021*, IEEE, , 9483–9490, https://doi.org/10.1109/ICPR48806.2021.9411918, URL https://doi.org/10.1109/ICPR48806.2021.9411918.

[57] Hady, M. F. A., and F. Schwenker, 2008: Co-training by Committee: A New Semi-supervised Learning Framework. *Workshops Proceedings of the 8th IEEE International Conference on Data Mining (ICDM 2008), December 15-19, 2008, Pisa, Italy*, IEEE Computer Society, , 563–572, https://doi.org/10.1109/ICDMW.2008.27, URL https://doi.org/10.1109/ICDMW.2008.27.

[58] Han, B., Q. Yao, T. Liu, G. Niu, I. W. Tsang, J. T. Kwok, and M. Sugiyama, 2020: A Survey of Label-noise Representation Learning: Past, Present and Future. *CoRR*, **abs/2011.04406**, 24, URL https://arxiv.org/abs/2011.04406, 2011.04406.

[59] He, K., X. Chen, S. Xie, Y. Li, P. Dollár, and R. B. Girshick, 2022: Masked Autoencoders Are Scalable Vision Learners. *CVPR*, IEEE, 15 979–15 988.

[60] Heidari, M., and Y. Guo, 2025: Bi-level optimization for pseudo-labeling based semi-supervised learning. URL https://openreview.net/forum?id=AEi2wyAMyb.

[61] Heidari, M., H. Zhang, and Y. Guo, 2024: Reinforcement learning-guided semi-supervised learning. URL https://arxiv.org/abs/2405.01760, 2405.01760.

[62] Hinton, G. E., O. Vinyals, and J. Dean, 2015: Distilling the Knowledge in a Neural Network. *CoRR*, **abs/1503.02531**, 9, URL http://arxiv.org/abs/1503.02531, 1503.02531.

[63] Horn, G. V., E. Cole, S. Beery, K. Wilber, S. J. Belongie, and O. M. Aodha, 2021: Benchmarking representation learning for natural world image collections. *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2021,*

virtual, June 19-25, 2021, Computer Vision Foundation / IEEE, 12 884–12 893, https://doi.org/10.1109/CVPR46437.2021.01269, URL https://openaccess.thecvf.com/content/CVPR2021/html/Van_Horn_Benchmarking_Representation_Learning_for_Natural_World_Image_Collections_CVPR_2021_paper.html.

[64] Huang, G., Z. Liu, L. van der Maaten, and K. Q. Weinberger, 2017: Densely Connected Convolutional Networks. *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, IEEE Computer Society, 2261–2269, https://doi.org/10.1109/CVPR.2017.243, URL https://doi.org/10.1109/CVPR.2017.243.

[65] Iscen, A., G. Tolias, Y. Avrithis, and O. Chum, 2019: Label Propagation for Deep Semi-Supervised Learning. *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, Computer Vision Foundation / IEEE, , 5070–5079, https://doi.org/10.1109/CVPR.2019.00521, URL http://openaccess.thecvf.com/content_CVPR_2019/html/Iscen_Label_Propagation_for_Deep_Semi-Supervised_Learning_CVPR_2019_paper.html.

[66] Jia, C., and Coauthors, 2021: Scaling Up Visual and Vision-Language Representation Learning With Noisy Text Supervision. *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, M. Meila, and T. Zhang, Eds., PMLR, , Proceedings of Machine Learning Research, Vol. 139, 4904–4916, URL http://proceedings.mlr.press/v139/jia21b.html.

[67] Jing, L., and Y. Tian, 2021: Self-Supervised Visual Feature Learning With Deep Neural Networks: A Survey. *IEEE Trans. Pattern Anal. Mach. Intell.*, **43 (11)**, 4037–4058, https://doi.org/10.1109/TPAMI.2020.2992393, URL https://doi.org/10.1109/TPAMI.2020.2992393.

[68] Jonsson, P., 2011: Classification of road conditions: From camera images and weather data. *2011 IEEE International Conference on Computational Intelligence for Measurement Systems and Applications (CIMSA) Proceedings*, 1–6, https://doi.org/10.1109/CIMSA.2011.6059917.

[69] Kage, P., and P. Andreadis, 2021: Class Introspection: A Novel Technique for Detecting Unlabeled Subclasses by Leveraging Classifier Explainability Methods. *CoRR*, **abs/2107.01657**, 10, URL https://arxiv.org/abs/2107.01657, 2107.01657.

[70] Kang, S., D. B. Lee, H. Jang, and S. J. Hwang, 2025: Simple yet effective semi-supervised knowledge distillation from vision-language models via dual-head optimization. URL https://arxiv.org/abs/2505.07675, 2505.07675.

[71] Kawai, S., K. Takeuchi, K. Shibata, and Y. Horita, 2014: A smart method to

distinguish road surface conditions at night-time using a car-mounted camera. *IEEJ Transactions on Electronics, Information and Systems*, **134**, 878–884, https://doi.org/10.1541/ieejeiss.134.878.

[72] Kawarabuki, H., and K. Onoguchi, 2014: Snowfall detection under bad visibility scene. *17th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, 3058–3063, https://doi.org/10.1109/ITSC.2014.6958181.

[73] Khan, M. N., and M. M. Ahmed, 2022: Weather and surface condition detection based on road-side webcams: Application of pre-trained convolutional neural network. *International Journal of Transportation Science and Technology*, **11 (3)**, 468–483, https://doi.org/https://doi.org/10.1016/j.ijtst.2021.06.003, URL https://www.sciencedirect.com/science/article/pii/S2046043021000526.

[74] Kim, B., J. Seo, S. Jeon, J. Koo, J. Choe, and T. Jeon, 2019: Why are Saliency Maps Noisy? Cause of and Solution to Noisy Saliency Maps. *2019 IEEE/CVF International Conference on Computer Vision Workshops, ICCV Workshops 2019, Seoul, Korea (South), October 27-28, 2019*, IEEE, 4149–4157, https://doi.org/10.1109/ICCVW.2019.00510, URL https://doi.org/10.1109/ICCVW.2019.00510.

[75] Kim, B., M. Wattenberg, J. Gilmer, C. J. Cai, J. Wexler, F. B. Viégas, and R. Sayres, 2018: Interpretability Beyond Feature Attribution: Quantitative Testing with Concept Activation Vectors (TCAV). *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, J. G. Dy, and A. Krause, Eds., PMLR, Proceedings of Machine Learning Research, Vol. 80, 2673–2682, URL http://proceedings.mlr.press/v80/kim18d.html.

[76] Kim, J. S., G. Plumb, and A. Talwalkar, 2022: Sanity Simulations for Saliency Methods. *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, K. Chaudhuri, S. Jegelka, L. Song, C. Szepesvári, G. Niu, and S. Sabato, Eds., PMLR, Proceedings of Machine Learning Research, Vol. 162, 11 173–11 200, URL https://proceedings.mlr.press/v162/kim22h.html.

[77] Kolek, S., D. A. Nguyen, R. Levie, J. Bruna, and G. Kutyniok, 2022: Cartoon explanations of image classifiers. *Computer Vision - ECCV 2022 - 17th European Conference, Tel Aviv, Israel, October 23-27, 2022, Proceedings, Part XII*, S. Avidan, G. J. Brostow, M. Cissé, G. M. Farinella, and T. Hassner, Eds., Springer, Lecture Notes in Computer Science, Vol. 13672, 443–458, https://doi.org/10.1007/978-3-031-19775-8_26, URL https://doi.org/10.1007/978-3-031-19775-8_26.

[78] Kolek, S., R. Windesheim, H. Andrade-Loarca, G. Kutyniok, and R. Levie, 2023: Explaining image classifiers with multiscale directional image representation.

IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2023, Vancouver, BC, Canada, June 17-24, 2023, IEEE, 18 600–18 609, https://doi.org/10.1109/CVPR52729.2023.01784, URL https://doi.org/10.1109/CVPR52729.2023.01784.

[79] Krizhevsky, A., 2009: Learning multiple layers of features from tiny images. URL http://www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf.

[80] Kumar, I. E., S. Venkatasubramanian, C. Scheidegger, and S. A. Friedler, 2020: Problems with Shapley-value-based explanations as feature importance measures. *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, PMLR, Proceedings of Machine Learning Research, Vol. 119, 5491–5500, URL http://proceedings.mlr.press/v119/kumar20e.html.

[81] Laine, S., and T. Aila, 2017: Temporal Ensembling for Semi-Supervised Learning. *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*, OpenReview.net, , 13, URL https://openreview.net/forum?id=BJ6oOfqge.

[82] Lakshminarayanan, B., A. Pritzel, and C. Blundell, 2017: Simple and Scalable Predictive Uncertainty Estimation using Deep Ensembles. *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, I. Guyon, U. von Luxburg, S. Bengio, H. M. Wallach, R. Fergus, S. V. N. Vishwanathan, and R. Garnett, Eds., , , 6402–6413, URL https://proceedings.neurips.cc/paper/2017/hash/9ef2ed4b7fd2c810847ffa5fa85bce38-Abstract.html.

[83] Lee, D.-H., 2013: Pseudo-Label : The Simple and Efficient Semi-Supervised Learning Method for Deep Neural Networks. *Workshop on Challenges in Representation Learning, ICML*, **3 (2)**, 896.

[84] Lee, H., M. Kang, J. Song, and K. Hwang, 2020: The detection of black ice accidents for preventative automated vehicles using convolutional neural networks. *Electronics*, **9 (12)**, https://doi.org/10.3390/electronics9122178, URL https://www.mdpi.com/2079-9292/9/12/2178.

[85] Lee, J., B. Hong, S. Jung, and V. Chang, 2018: Clustering learning model of cctv image pattern for producing road hazard meteorological information. *Future Generation Computer Systems*, **86**, 1338–1350, https://doi.org/https://doi.org/10.1016/j.future.2018.03.022, URL https://www.sciencedirect.com/science/article/pii/S0167739X17318253.

[86] Lee, J., B. Hong, Y. Shin, and Y.-J. Jang, 2016: Extraction of weather informa-

tion on road using cctv video. *2016 International Conference on Big Data and Smart Computing (BigComp)*, 529–531, https://doi.org/10.1109/BIGCOMP.2016.7425986.

[87] Li, C., and et al., 2022: Efficient Self-supervised Vision Transformers for Representation Learning. *ICLR*.

[88] Liu, H., K. Son, J. Yang, C. Liu, J. Gao, Y. J. Lee, and C. Li, 2023: Learning Customized Visual Models with Retrieval-Augmented Knowledge. *CVPR*, IEEE, 15 148–15 158.

[89] Lopes, R. G., S. Fenu, and T. Starner, 2017: Data-Free Knowledge Distillation for Deep Neural Networks. *CoRR*, **abs/1710.07535**, 8, URL http://arxiv.org/abs/1710.07535, 1710.07535.

[90] Lundberg, S. M., and S. Lee, 2017: A Unified Approach to Interpreting Model Predictions. *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, I. Guyon, U. von Luxburg, S. Bengio, H. M. Wallach, R. Fergus, S. V. N. Vishwanathan, and R. Garnett, Eds., 4765–4774, URL https://proceedings.neurips.cc/paper/2017/hash/8a20a8621978632d76c43dfd28b67767-Abstract.html.

[91] MacDonald, J., S. Wäldchen, S. Hauch, and G. Kutyniok, 2019: A rate-distortion framework for explaining neural network decisions. *CoRR*, **abs/1905.11092**, URL http://arxiv.org/abs/1905.11092, 1905.11092.

[92] Maji, S., J. Kannala, E. Rahtu, M. Blaschko, and A. Vedaldi, 2013: Fine-Grained Visual Classification of Aircraft. Tech. rep., VGG, University of Oxford. 1306.5151.

[93] Minderer, M., A. A. Gritsenko, and N. Houlsby, 2023: Scaling Open-Vocabulary Object Detection. *CoRR*, **abs/2306.09683**, 22, https://doi.org/10.48550/ARXIV.2306.09683, URL https://doi.org/10.48550/arXiv.2306.09683, 2306.09683.

[94] Miyato, T., S. Maeda, M. Koyama, and S. Ishii, 2019: Virtual Adversarial Training: A Regularization Method for Supervised and Semi-Supervised Learning. *IEEE Trans. Pattern Anal. Mach. Intell.*, **41 (8)**, 1979–1993, https://doi.org/10.1109/TPAMI.2018.2858821, URL https://doi.org/10.1109/TPAMI.2018.2858821.

[95] Nigam, K., A. K. Mccallum, S. Thrun, and T. Mitchell, 2000: Text Classification from Labeled and Unlabeled Documents using EM. *Machine Learning*, **39 (2)**, 103–134, https://doi.org/10.1023/A:1007692713085.

[96] Nilsback, M., and A. Zisserman, 2008: Automated Flower Classification over a Large Number of Classes. *ICVGIP*, 722–729.

[97] Noroozi, M., and P. Favaro, 2016: Unsupervised learning of visual representations by solving jigsaw puzzles. *CoRR*, **abs/1603.09246**, URL http://arxiv.org/abs/1603.09246, 1603.09246.

[98] Ojala, R., and E. Alamikkotervo, 2024: Road surface friction estimation for winter conditions utilising general visual features. *arXiv.org*, https://doi.org/10.48550/arxiv.2404.16578.

[99] Omer, R., and L. Fu, 2010: An automatic image recognition system for winter road surface condition classification. *13th International IEEE Conference on Intelligent Transportation Systems*, 1375–1379, https://doi.org/10.1109/ITSC.2010.5625290.

[100] Oquab, M., and et al., 2024: DINOv2: Learning Robust Visual Features without Supervision. *Trans. Mach. Learn. Res.*, **2024**.

[101] Pan, G., L. Fu, R. Yu, and M. Muresan, 2018: Winter road surface condition recognition using a pretrained deep convolutional network. 1812.06858.

[102] Pan, G., L. Fu, R. Yu, and M. Muresan, 2019: Evaluation of alternative pre-trained convolutional neural networks for winter road surface condition monitoring. *2019 5th International Conference on Transportation Information and Safety (ICTIS)*, 614–620, https://doi.org/10.1109/ICTIS.2019.8883540.

[103] Parkhi, O. M., A. Vedaldi, A. Zisserman, and C. V. Jawahar, 2012: Cats and dogs. *2012 IEEE Conference on Computer Vision and Pattern Recognition, Providence, RI, USA, June 16-21, 2012*, IEEE Computer Society, 3498–3505, https://doi.org/10.1109/CVPR.2012.6248092, URL https://doi.org/10.1109/CVPR.2012.6248092.

[104] Petsiuk, V., R. Jain, V. Manjunatha, V. I. Morariu, A. Mehra, V. Ordonez, and K. Saenko, 2021: Black-Box Explanation of Object Detectors via Saliency Maps. *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2021, virtual, June 19-25, 2021*, Computer Vision Foundation / IEEE, 11 443–11 452, https://doi.org/10.1109/CVPR46437.2021.01128, URL https://openaccess.thecvf.com/content/CVPR2021/html/Petsiuk_Black-Box_Explanation_of_Object_Detectors_via_Saliency_Maps_CVPR_2021_paper.html.

[105] Pham, H., Z. Dai, Q. Xie, and Q. V. Le, 2021: Meta Pseudo Labels. *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2021, virtual, June 19-25, 2021*, Computer Vision Foundation / IEEE, , 11 557–11 568, https://doi.org/10.1109/CVPR46437.2021.01139, URL https://openaccess.thecvf.com/content/

CVPR2021/html/Pham_Meta_Pseudo_Labels_CVPR_2021_paper.html.

[106] Plumb, G., D. Molitor, and A. Talwalkar, 2018: Model Agnostic Supervised Local Explanations. *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, S. Bengio, H. M. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, Eds., 2520–2529, URL https://proceedings.neurips.cc/paper/2018/hash/b495ce63ede0f4efc9eec62cb947c162-Abstract.html.

[107] Qian, Y., E. J. Almazan, and J. H. Elder, 2016: Evaluating features and classifiers for road weather condition analysis. *2016 IEEE International Conference on Image Processing (ICIP)*, 4403–4407, https://doi.org/10.1109/ICIP.2016.7533192.

[108] Qiao, S., W. Shen, Z. Zhang, B. Wang, and A. L. Yuille, 2018: Deep Co-Training for Semi-Supervised Image Recognition. *Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part XV*, V. Ferrari, M. Hebert, C. Sminchisescu, and Y. Weiss, Eds., Springer, Lecture Notes in Computer Science, Vol. 11219, 142–159, https://doi.org/10.1007/978-3-030-01267-0_9, URL https://doi.org/10.1007/978-3-030-01267-0_9.

[109] Radford, A., and et al., 2021: Learning Transferable Visual Models From Natural Language Supervision. *ICML*, 8748–8763.

[110] Radford, A., and Coauthors, 2021: Learning Transferable Visual Models From Natural Language Supervision. *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, M. Meila, and T. Zhang, Eds., PMLR, Proceedings of Machine Learning Research, Vol. 139, 8748–8763, URL http://proceedings.mlr.press/v139/radford21a.html.

[111] Rai, A., 2020: Explainable AI: From black box to glass box. *Journal of the Academy of Marketing Science*, **48 (1)**, 137–141.

[112] Ribeiro, M. T., S. Singh, and C. Guestrin, 2016: "Why Should I Trust You?": Explaining the Predictions of Any Classifier. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016*, B. Krishnapuram, M. Shah, A. J. Smola, C. C. Aggarwal, D. Shen, and R. Rastogi, Eds., ACM, 1135–1144, https://doi.org/10.1145/2939672.2939778, URL https://doi.org/10.1145/2939672.2939778.

[113] Ribeiro, M. T., S. Singh, and C. Guestrin, 2018: Anchors: High-Precision Model-Agnostic Explanations. *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial*

Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018, S. A. McIlraith, and K. Q. Weinberger, Eds., AAAI Press, 1527–1535, URL https://www.aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/16982.

[114] Rizve, M. N., K. Duarte, Y. S. Rawat, and M. Shah, 2021: In Defense of Pseudo-Labeling: An Uncertainty-Aware Pseudo-label Selection Framework for Semi-Supervised Learning. *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*, OpenReview.net, , 20, URL https://openreview.net/forum?id=-ODN6SbiUU.

[115] Ronneberger, O., P. Fischer, and T. Brox, 2015: U-net: Convolutional networks for biomedical image segmentation. *International Conference on Medical image computing and computer-assisted intervention*, Springer, 234–241.

[116] Rothenberger, J., S. Crowell, W. Keely, D. Cusworth, and K. Howell, 2024: The Promise of Foundation Models for Improving Automated Detection of Methane Sources in Hyperspectral Imagers: A Case Study for the EMIT Instrument. *AGU Fall Meeting Abstracts*, AGU Fall Meeting Abstracts, Vol. 2024, H01–08.

[117] Rothenberger, J. C., and D. I. Diochnos, 2023: Meta Co-Training: Two Views are Better than One. URL https://doi.org/10.48550/arXiv.2311.18083, https://doi.org/10.48550/ARXIV.2311.18083, 2311.18083.

[118] Rothenberger, J. C., E. P. Grimit, M. J. Murphy, and R. Wallace, 2024: Explaining the Role of Lightning Data in Hail Nowcasting. *104th Annual AMS Meeting 2024*, American Meteorological Society Meeting Abstracts, Vol. 104, 437201.

[119] Rothenberger, J. C., P. Kage, P. Andreadis, and D. I. Diochnos, 2025: A review of pseudo-labeling for computer vision. URL https://arxiv.org/abs/2408.07221, 2408.07221.

[120] Rothenberger, J. C., T. Le, C. Sutter, K. J. Sulia, and D. Diochnos, 2025: Improving Road Surface Classification with Co-Training Algorithms. *105th Annual AMS Meeting 2025*, American Meteorological Society Meeting Abstracts, Vol. 105, 451835.

[121] Rudin, C., 2019: Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nat. Mach. Intell.*, **1 (5)**, 206–215, https://doi.org/10.1038/s42256-019-0048-x, URL https://doi.org/10.1038/s42256-019-0048-x.

[122] Ruspini, E. H., 1969: A new approach to clustering. *Information and Control*,

15 (1), 22–32, https://doi.org/https://doi.org/10.1016/S0019-9958(69)90591-9, URL https://www.sciencedirect.com/science/article/pii/S0019995869905919.

[123] Schulman, J., F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, 2017: Proximal policy optimization algorithms. *CoRR*, **abs/1707.06347**, URL http://arxiv.org/abs/1707.06347, 1707.06347.

[124] Scudder, H., 1965: Probability of error of some adaptive pattern-recognition machines. *IEEE Trans. on Inf. Th.*, **11 (3)**, 363–371.

[125] Selvaraju, R. R., M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, 2020: Grad-CAM: Visual Explanations from Deep Networks via Gradient-Based Localization. *Int. J. Comput. Vis.*, **128 (2)**, 336–359, https://doi.org/10.1007/s11263-019-01228-7, URL https://doi.org/10.1007/s11263-019-01228-7.

[126] Shi, W., Y. Gong, C. Ding, Z. Ma, X. Tao, and N. Zheng, 2018: Transductive Semi-Supervised Deep Learning Using Min-Max Features. *Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part V*, V. Ferrari, M. Hebert, C. Sminchisescu, and Y. Weiss, Eds., Springer, , Lecture Notes in Computer Science, Vol. 11209, 311–327, https://doi.org/10.1007/978-3-030-01228-1_19, URL https://doi.org/10.1007/978-3-030-01228-1_19.

[127] Shrikumar, A., P. Greenside, and A. Kundaje, 2017: Learning Important Features Through Propagating Activation Differences. *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, D. Precup, and Y. W. Teh, Eds., PMLR, Proceedings of Machine Learning Research, Vol. 70, 3145–3153, URL http://proceedings.mlr.press/v70/shrikumar17a.html.

[128] Simonyan, K., A. Vedaldi, and A. Zisserman, 2014: Deep inside convolutional networks: Visualising image classification models and saliency maps. *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Workshop Track Proceedings*, Y. Bengio, and Y. LeCun, Eds., URL http://arxiv.org/abs/1312.6034.

[129] Siméoni, O., and Coauthors, 2025: Dinov3. URL https://arxiv.org/abs/2508.10104, 2508.10104.

[130] Sirirattanapol, C., M. NAGAI, A. Witayangkurn, S. Pravinvongvuth, and M. Ekpanyapong, 2019: Bangkok cctv image through a road environment extraction system using multi-label convolutional neural network classification. *ISPRS International Journal of Geo-Information*, **8 (3)**, https://doi.org/10.3390/ijgi8030128,

URL https://www.mdpi.com/2220-9964/8/3/128.

[131] Slack, D., A. Hilgard, S. Singh, and H. Lakkaraju, 2021: Reliable Post hoc Explanations: Modeling Uncertainty in Explainability. *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, M. Ranzato, A. Beygelzimer, Y. N. Dauphin, P. Liang, and J. W. Vaughan, Eds., 9391–9404, URL https://proceedings.neurips.cc/paper/2021/hash/4e246a381baf2ce038b3b0f82c7d6fb4-Abstract.html.

[132] Slack, D., S. Hilgard, E. Jia, S. Singh, and H. Lakkaraju, 2020: Fooling LIME and SHAP: adversarial attacks on post hoc explanation methods. *AIES '20: AAAI/ACM Conference on AI, Ethics, and Society, New York, NY, USA, February 7-8, 2020*, A. N. Markham, J. Powles, T. Walsh, and A. L. Washington, Eds., ACM, 180–186, https://doi.org/10.1145/3375627.3375830, URL https://doi.org/10.1145/3375627.3375830.

[133] Sohn, K., and Coauthors, 2020: FixMatch: Simplifying Semi-Supervised Learning with Consistency and Confidence. *NeurIPS*, URL https://proceedings.neurips.cc/paper/2020/hash/06964dce9addb1c5cb5d6e3d9838f733-Abstract.html.

[134] Song, H., M. Kim, D. Park, Y. Shin, and J. Lee, 2023: Learning From Noisy Labels With Deep Neural Networks: A Survey. *IEEE Trans. Neural Networks Learn. Syst.*, **34 (11)**, 8135–8153, https://doi.org/10.1109/TNNLS.2022.3152527, URL https://doi.org/10.1109/TNNLS.2022.3152527.

[135] Sun, S., F. Jin, and W. Tu, 2011: View Construction for Multi-view Semi-supervised Learning. *Advances in Neural Networks - ISNN 2011 - 8th International Symposium on Neural Networks, ISNN 2011, Guilin, China, May 29-June 1, 2011, Proceedings, Part I*, D. Liu, H. Zhang, M. M. Polycarpou, C. Alippi, and H. He, Eds., Springer, , Lecture Notes in Computer Science, Vol. 6675, 595–601, https://doi.org/10.1007/978-3-642-21105-8_69, URL https://doi.org/10.1007/978-3-642-21105-8_69.

[136] Sun, Z., and K. Jia, 2013: Road surface condition classification based on color and texture information. *2013 Ninth International Conference on Intelligent Information Hiding and Multimedia Signal Processing*, 137–140, https://doi.org/10.1109/IIH-MSP.2013.43.

[137] Sundararajan, M., A. Taly, and Q. Yan, 2017: Axiomatic Attribution for Deep Networks. *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, D. Precup, and Y. W. Teh, Eds., PMLR, Proceedings of Machine Learning Research, Vol. 70, 3319–3328,

URL http://proceedings.mlr.press/v70/sundararajan17a.html.

[138] Sutter, C., and Coauthors, 2025: Quantitative Content Analysis Data for Hand Labeling Road Surface Conditions in New York State Department of Transportation Camera Images. Zenodo, URL https://doi.org/10.5281/zenodo.15257486, https://doi.org/10.5281/zenodo.15257486.

[139] Sutter, C., and Coauthors, 2026: Machine learning detection of road surface conditions: A generalizable model using traffic cameras and weather data. URL https://arxiv.org/abs/2510.06440, 2510.06440.

[140] Tarvainen, A., and H. Valpola, 2017: Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, I. Guyon, U. von Luxburg, S. Bengio, H. M. Wallach, R. Fergus, S. V. N. Vishwanathan, and R. Garnett, Eds., , , 1195–1204, URL https://proceedings.neurips.cc/paper/2017/hash/68053af2923e00204c3ca7c6a3150cf7-Abstract.html.

[141] Tarvainen, A., and H. Valpola, 2017: Weight-averaged consistency targets improve semi-supervised deep learning results. *CoRR*, **abs/1703.01780**, URL http://arxiv.org/abs/1703.01780, 1703.01780.

[142] Tong Xiao, Tian Xia, Yi Yang, Chang Huang, and Xiaogang Wang, 2015: Learning from Massive Noisy Labeled Data for Image Classification. *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, Boston, MA, USA, 2691–2699, https://doi.org/10.1109/CVPR.2015.7298885.

[143] Tschannen, M., and Coauthors, 2025: Siglip 2: Multilingual vision-language encoders with improved semantic understanding, localization, and dense features. *CoRR*, **abs/2502.14786**, https://doi.org/10.48550/ARXIV.2502.14786, URL https://doi.org/10.48550/arXiv.2502.14786, 2502.14786.

[144] Vaswani, A., N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, 2017: Attention is all you need. *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, I. Guyon, U. von Luxburg, S. Bengio, H. M. Wallach, R. Fergus, S. V. N. Vishwanathan, and R. Garnett, Eds., 5998–6008, URL https://proceedings.neurips.cc/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html.

[145] Wagner, J., J. M. Köhler, T. Gindele, L. Hetzel, J. T. Wiedemer, and S. Behnke, 2019: Interpretable and Fine-Grained Visual Explanations for Convolutional

Neural Networks. *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, Computer Vision Foundation / IEEE, 9097–9107, https://doi.org/10.1109/CVPR.2019.00931, URL http://openaccess.thecvf.com/content_CVPR_2019/html/Wagner_Interpretable_and_Fine-Grained_Visual_Explanations_for_Convolutional_Neural_Networks_CVPR_2019_paper.html.

[146] Wallin, E., L. Svensson, F. Kahl, and L. Hammarstrand, 2023: Improving open-set semi-supervised learning with self-supervision. URL https://arxiv.org/abs/2301.10127, 2301.10127.

[147] Wallin, E., L. Svensson, F. Kahl, and L. Hammarstrand, 2024: Prosub: Probabilistic open-set semi-supervised learning with subspace-based out-of-distribution detection. URL https://arxiv.org/abs/2407.11735, 2407.11735.

[148] Wang, J., S. Luo, and X. Zeng, 2008: A random subspace method for co-training. *Proceedings of the International Joint Conference on Neural Networks, IJCNN 2008, part of the IEEE World Congress on Computational Intelligence, WCCI 2008, Hong Kong, China, June 1-6, 2008*, IEEE, , 195–200, https://doi.org/10.1109/IJCNN.2008.4633789, URL https://doi.org/10.1109/IJCNN.2008.4633789.

[149] Wang, Q., B. Han, T. Liu, G. Niu, J. Yang, and C. Gong, 2021: Tackling Instance-Dependent Label Noise via a Universal Probabilistic Model. *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, AAAI Press, , 10 183–10 191, https://doi.org/10.1609/AAAI.V35I11.17221, URL https://doi.org/10.1609/aaai.v35i11.17221.

[150] Wang, Q., B. Han, Y. Liu, C. Gong, T. Liu, and J. Liu, 2025: W-DOE: Wasserstein Distribution-Agnostic Outlier Exposure . *IEEE Transactions on Pattern Analysis & Machine Intelligence*, **47 (05)**, 3530–3545, https://doi.org/10.1109/TPAMI.2025.3531000, URL https://doi.ieeecomputersociety.org/10.1109/TPAMI.2025.3531000.

[151] Wang, W., and Z. Zhou, 2013: Co-Training with Insufficient Views. *Asian Conference on Machine Learning, ACML 2013, Canberra, ACT, Australia, November 13-15, 2013*, C. S. Ong, and T. B. Ho, Eds., JMLR.org, , JMLR Workshop and Conference Proceedings, Vol. 29, 467–482, URL http://proceedings.mlr.press/v29/Wang13b.html.

[152] Wang, Y., J. Guo, S. Song, and G. Huang, 2020: Meta-semi: A meta-learning approach for semi-supervised learning. *CoRR*, **abs/2007.02394**, URL https:

//arxiv.org/abs/2007.02394, 2007.02394.

[153] Wen, T., S. Lai, and X. Qian, 2021: Preparing lessons: Improve knowledge distillation with better supervision. *Neurocomputing*, **454**, 25–33, https://doi.org/10.1016/J.NEUCOM.2021.04.102, URL https://doi.org/10.1016/j.neucom.2021.04.102.

[154] Williams, R. J., 1992: Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Mach. Learn.*, **8 (3–4)**, 229–256, https://doi.org/10.1007/BF00992696, URL https://doi.org/10.1007/BF00992696.

[155] Wu, M., and T. J. Kwon, 2022: An automatic architecture designing approach of convolutional neural networks for road surface conditions image recognition: Tradeoff between accuracy and efficiency. *Journal of Sensors*, **2022**, 1–11, https://doi.org/10.1155/2022/3325282.

[156] Xie, Q., Z. Dai, E. H. Hovy, T. Luong, and Q. Le, 2020: Unsupervised Data Augmentation for Consistency Training. *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, Eds., , , 20, URL https://proceedings.neurips.cc/paper/2020/hash/44feb0096faa8326192570788b38c1d1-Abstract.html.

[157] Xie, Q., M. Luong, E. H. Hovy, and Q. V. Le, 2020: Self-Training With Noisy Student Improves ImageNet Classification. *CVPR*, 10 684–10 695.

[158] Yamada, M., K. Ueda, I. Horiba, and N. Sugie, 2005: Discrimination of the road condition toward understanding of vehicle driving environments. *IEEE Transactions on Intelligent Transportation Systems*, **2 (1)**, 26–31, https://doi.org/10.1109/6979.911083.

[159] Yan, X., Y. Luo, and X. Zheng, 2009: Weather recognition based on images captured by vision system in vehicle. *International Symposium on Neural Networks*, URL https://api.semanticscholar.org/CorpusID:39282129.

[160] Yan, Y., Z. Xu, I. W. Tsang, G. Long, and Y. Yang, 2016: Robust Semi-Supervised Learning through Label Aggregation. *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, February 12-17, 2016, Phoenix, Arizona, USA*, D. Schuurmans, and M. P. Wellman, Eds., AAAI Press, , 2244–2250, https://doi.org/10.1609/AAAI.V30I1.10276, URL https://doi.org/10.1609/aaai.v30i1.10276.

[161] Yang, H.-J., H. Jang, J.-W. Kang, and D.-S. Jeong, 2014: Classification algorithm

for road surface condition. *International Journal of Computer Science and Network Security (IJCSNS)*, **14 (1)**, 1.

[162] Yang, J., C. Li, and J. Gao, 2022: Focal modulation networks. *arXiv preprint arXiv:2203.11926*.

[163] Yang, S., and C. Lei, 2024: Research on the classification method of complex snow and ice cover on highway pavement based on image-meteorology-temperature fusion. *IEEE Sensors Journal*, **24 (2)**, 1784–1791, https://doi.org/10.1109/JSEN.2023.3336667.

[164] Yarowsky, D., 1995: Unsupervised word sense disambiguation rivaling supervised methods. *33rd Annual Meeting of the Association for Computational Linguistics*, Association for Computational Linguistics, Cambridge, Massachusetts, USA, 189–196, https://doi.org/10.3115/981658.981684, URL https://aclanthology.org/P95-1026/.

[165] Yi, K., G.-H. Wang, and J. Wu, 2022: arXiv:2202.08436. PENCIL: Deep Learning with Noisy Labels. arXiv, https://doi.org/10.48550/arXiv.2202.08436, 2202.08436.

[166] Yuan, B., J. Chen, W. Zhang, H. Tai, and S. McMains, 2018: Iterative Cross Learning on Noisy Labels. *2018 IEEE Winter Conference on Applications of Computer Vision, WACV 2018, Lake Tahoe, NV, USA, March 12-15, 2018*, IEEE Computer Society, , 757–765, https://doi.org/10.1109/WACV.2018.00088, URL https://doi.org/10.1109/WACV.2018.00088.

[167] Zadeh, L., 1965: Fuzzy sets. *Information and Control*, **8 (3)**, 338–353, https://doi.org/https://doi.org/10.1016/S0019-9958(65)90241-X, URL https://www.sciencedirect.com/science/article/pii/S001999586590241X.

[168] Zeiler, M. D., and R. Fergus, 2014: Visualizing and Understanding Convolutional Networks. *Computer Vision - ECCV 2014 - 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part I*, D. J. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, Eds., Springer, Lecture Notes in Computer Science, Vol. 8689, 818–833, https://doi.org/10.1007/978-3-319-10590-1_53, URL https://doi.org/10.1007/978-3-319-10590-1_53.

[169] Zhai, X., B. Mustafa, A. Kolesnikov, and L. Beyer, 2023: Sigmoid Loss for Language Image Pre-Training. *IEEE/CVF International Conference on Computer Vision, ICCV 2023, Paris, France, October 1-6, 2023*, IEEE, , 11 941–11 952, https://doi.org/10.1109/ICCV51070.2023.01100, URL https://doi.org/10.1109/ICCV51070.2023.01100.

[170] Zhang, B., Y. Wang, W. Hou, H. Wu, J. Wang, M. Okumura, and T. Shinozaki, 2021: FlexMatch: Boosting Semi-Supervised Learning with Curriculum Pseudo Labeling. *NeurIPS*, 18 408–18 419, URL https://proceedings.neurips.cc/paper/2021/hash/995693c15f439e3d189b06e89d145dd5-Abstract.html.

[171] Zhang, C., W. Wang, Y. Su, and X. Bai, 2012: Discrimination of highway snow condition with video monitor for safe driving environment. *2012 5th International Congress on Image and Signal Processing*, 1241–1244, https://doi.org/10.1109/CISP.2012.6469850.

[172] Zhang, H., M. Cissé, Y. N. Dauphin, and D. Lopez-Paz, 2018: mixup: Beyond Empirical Risk Minimization. *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*, OpenReview.net, , 13, URL https://openreview.net/forum?id=r1Ddp1-Rb.

[173] Zhang, L., J. Song, A. Gao, J. Chen, C. Bao, and K. Ma, 2019: Be Your Own Teacher: Improve the Performance of Convolutional Neural Networks via Self Distillation. *2019 IEEE/CVF International Conference on Computer Vision, ICCV 2019, Seoul, Korea (South), October 27 - November 2, 2019*, IEEE, , 3712–3721, https://doi.org/10.1109/ICCV.2019.00381, URL https://doi.org/10.1109/ICCV.2019.00381.

[174] Zhang, Q., Y. N. Wu, and S. Zhu, 2018: Interpretable Convolutional Neural Networks. *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, Computer Vision Foundation / IEEE Computer Society, 8827–8836, https://doi.org/10.1109/CVPR.2018.00920, URL http://openaccess.thecvf.com/content_cvpr_2018/html/Zhang_Interpretable_Convolutional_Neural_CVPR_2018_paper.html.

[175] Zhang, R., P. Isola, and A. A. Efros, 2016: Colorful image colorization. URL https://arxiv.org/abs/1603.08511, 1603.08511.

[176] Zhou, T., S. Wang, and J. A. Bilmes, 2020: Time-Consistent Self-Supervision for Semi-Supervised Learning. *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, PMLR, , Proceedings of Machine Learning Research, Vol. 119, 11 523–11 533, URL http://proceedings.mlr.press/v119/zhou20d.html.

[177] Zhou, Y., and S. A. Goldman, 2004: Democratic Co-Learning. *16th IEEE International Conference on Tools with Artificial Intelligence (ICTAI 2004), 15-17 November 2004, Boca Raton, FL, USA*, IEEE Computer Society, , 594–602, https://doi.org/10.1109/ICTAI.2004.48, URL https://doi.org/10.1109/ICTAI.

2004.48.

[178] Zhou, Z., and M. Li, 2005: Tri-Training: Exploiting Unlabeled Data Using Three Classifiers. *IEEE Trans. Knowl. Data Eng.*, **17 (11)**, 1529–1541, https://doi.org/10.1109/TKDE.2005.186, URL https://doi.org/10.1109/TKDE.2005.186.

[179] Zhu, X., and Z. Ghahramani, 2002: Learning from Labeled and Unlabeled Data with Label Propagation. Tech. rep., Carnegie Mellon University, Pittsburgh, PA, USA.