Meta Co-Training: Two Views are Better than One

Jay C. Rothenberger[a,*] **and Dimitrios I. Diochnos**[a]

[a]The University of Oklahoma
ORCID (Jay C. Rothenberger): https://orcid.org/0009-0007-2530-4667, ORCID (Dimitrios I. Diochnos): https://orcid.org/0000-0002-2934-606X

Abstract. In many critical computer vision scenarios unlabeled data is plentiful, but labels are scarce and difficult to obtain. As a result, semi-supervised learning which leverages unlabeled data to boost the performance of supervised classifiers have received significant attention in recent literature. One representative class of semi-supervised algorithms are *co-training* algorithms. Co-training algorithms leverage two different models which have access to different independent and sufficient representations or "views" of the data to jointly make better predictions. Each of these models creates pseudo-labels on unlabeled points which are used to improve the other model. We show that in the common case where independent views are not available, we can construct such views inexpensively using pre-trained models. Co-training on the constructed views yields a performance improvement over any of the individual views we construct and performance comparable with recent approaches in semi-supervised learning. We present *Meta Co-Training*, a novel semi-supervised learning algorithm, which has two advantages over co-training: *(i)* learning is more robust when there is large discrepancy between the information content of the different views, and *(ii)* does not require retraining from scratch on each iteration. Our method achieves new state-of-the-art performance on ImageNet-10% achieving a $\sim 4.7\%$ reduction in error rate over prior work. Our method also outperforms prior semi-supervised work on several other fine-grained image classification datasets.

1 Introduction

In many critical machine learning scenarios we have access to a large amount of unlabeled data, and relatively few labeled data points for a given task. For standard computer vision (CV) tasks there are large well-known and open source datasets with hundreds of millions or even billions of unlabeled images [47, 46]. In contrast, labeled data is usually an order of magnitude more scarce and otherwise expensive to obtain, requiring many human-hours to generate. In this context, *semi-supervised learning (SSL)* methods are useful, as they rely on training more performant models, using small amounts of labeled data and large amounts of unlabeled data.

Not unrelated to and not to be confused with the idea of ***semi**-supervised learning*, is ***self**-supervised learning*.[1] In self-supervised learning a model is trained with an objective that does not require a label that is not evident from the data itself. Self-supervised learning was popularized by the BERT model [18] 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 we might want to solve, 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 [27, 12, 34, 10, 41, 26]. 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 [33, 5, 43, 52, 42, 49, 51, 4] 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 is to enforce that different augmented versions of the same instance are assigned the same label. Entropy minimization is to enforce 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 [51, 4, 49, 3], and entropy minimization is achieved by iterative re-training [33, 5, 43] or sharpening labels [42, 4, 49, 3].

One particular class of SSL algorithms are co-training [5] algorithms in which two different "views" of the data must be obtained or constructed, and then two different models must be trained and retrained iteratively. These two views yield models which 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. We show that constructing views which 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.

We propose a novel SSL method which more effectively leverages pseudo-labels. We call our 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

[*] Corresponding Author. Email: jay.c.rothenberger@ou.edu.

[1] There is also an SSL technique called *self-training*. Recognizing the unfortunate similarity of these three terms, SSL will always mean *semi-supervised learning* in this text and we will not abbreviate the other terms.

and the performance of the other model on a labeled set. Similarly to CT, our approach utilizes multiple views to train models which 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. We show that our approach provides an improvement over both CT and the current state-of-the-art (SoTA) method [36] on the ImageNet-10% dataset, as well as it establishes new SoTA few-shot performance on several other fine-grained image classification tasks.

Summary of Contributions.

1. We propose *Meta Co-Training*, an SSL algorithm which appears to be more robust than co-training and does not require iterative re-training from initialization.
2. We establish SoTA top-1 ImageNet-10% [17] accuracy and other few-shot benchmark classification tasks.
3. We make our implementation publicly available: https://github.com/JayRothenberger/Meta-Co-Training

Outline of the Paper. In Section 2 we establish notation and provide background information. In Section 3 we describe our proposed method, meta co-training. In Section 4 we present our experimental findings on ImageNet [17] and discuss how our method compares to relevant baseline approaches. We perform additional experiments on Flowers102 [40], Food101 [6], FGVCAircraft [38], iNaturalist [28], and iNaturalist 2021 [29] datasets. In Section 5 we discuss related work with emphasis on semi-supervised methods, including co-training methods. In Section 6 we conclude with a summary and directions for future work. The interested reader can find additional information (e.g., details on the training recipe) in https://arxiv.org/abs/2311.18083 [45].

2 Background

Notation. The models that we learn are functions of the form $f\colon \mathcal{X} \to \Delta^{|\mathcal{Y}|}$ where $\Delta^{|\mathcal{Y}|}$ is the $|\mathcal{Y}|$-dimensional unit simplex. We use $f(x)|_j$ to denote the j-th value of f in the output. When we want to explicitly refer to a network f parameterized by θ (e.g., in deep neural networks), we write f_θ. For the entirety of this text ℓ will refer to the cross-entropy loss which is defined as $\ell\left(y, f_\theta(x)\right) = -\sum_{\xi\in\mathcal{Y}} \mathbf{1}\left\{\xi = y\right\}\log\left(f_\theta(x)|_\xi\right)$. We use η as the *learning rate* and T as the *maximum number of steps (or updates)* in a gradient-based optimization procedure. Finally, we use $\mathbf{1}\left\{\mathcal{A}\right\}$ as an indicator function of an event $\mathcal{A}$; that is, $\mathbf{1}\left\{\mathcal{A}\right\}$ is equal to 1 when $\mathcal{A}$ holds, otherwise 0.

Semi-Supervised Learning. We are interested in situations where we have a large pool of unlabeled instances U as well as a small portion L of them that is labeled. That is, the learning algorithm has access to a dataset $S = L \cup U$, such that $L = \{(x_i, y_i)\}_{i=1}^m$ and $U = \{x_j\}_{j=1}^u$. For L the instance part is denoted as X_L and the label part is denoted as Y_L. In this setting, *semi-supervised learning (SSL)* methods attempt to first learn an initial model f_{init} using the labeled data L via a supervised learning algorithm, and in sequence, an attempt is being made in order to harness the information that is hidden in the unlabeled set U, so that a better model f can be learnt.

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 [19], automatic feature subsetting [11], random feature subsetting [8], random subspace selection [7], and adversarial examples [43]. We use self-supervised learning to construct views for our 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 our approach is more widely applicable, it is particularly well-suited to computer vision. There are a plethora [23, 12, 10, 34, 44, 41, 27] of competitive learned representations for images. Choosing foundation model representations as views makes our approach lightweight enough to be feasible with limited hardware resources.

3 Proposed Method: Meta Co-Training

We 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 which is inspired by Meta Pseudo Labels (MPL) [42] which we extend to two views to perform co-training [5]. A student is optimized to replicate the labels 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. We say that MCT is a co-training algorithm because it operates on two views of the data. However unlike the original co-training method [5] we do 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 which 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 2). The student parameters θ_S are optimized as a function of the teacher parameters θ_T:

$$\mathcal{L}_u\left(\theta_T, \theta_S\right) = \ell\left(\arg\max_{\xi} f_{\theta_T}(U)|_\xi, f_{\theta_S}(U)\right) \tag{1}$$

$$\theta'_S = \theta_S^{\text{PL}}(\theta_T) \in \arg\min_{\theta_S} \mathcal{L}_u\left(\theta_T, \theta_S\right)\ . \tag{2}$$

The teacher optimization (Equation 4) forms the upper level of the optimization:

$$\mathcal{L}_L\left(\theta'_S\right) = \ell\left(Y_L, f_{\theta'_S}(X_L)\right) \tag{3}$$

$$\theta'_T = \arg\min_{\theta_T} \mathcal{L}_L\left(\theta'_S\right)\ . \tag{4}$$

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

$$\min_{\theta_T} \mathcal{L}_u\left(\theta_T, \theta_S\right) + \mathcal{L}_L\left(\theta'_S\right)\ . \tag{5}$$

It is cumbersome to speak of teachers and students when each model is both. We give the objective of MCT in Equation 6 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.

$$\min_{\theta_1,\theta_2} \mathcal{L}_u\left(\theta_1, \theta_2\right) + \mathcal{L}_L\left(\theta'_2\right) + \mathcal{L}_u\left(\theta_2, \theta_1\right) + \mathcal{L}_L\left(\theta'_1\right) \tag{6}$$

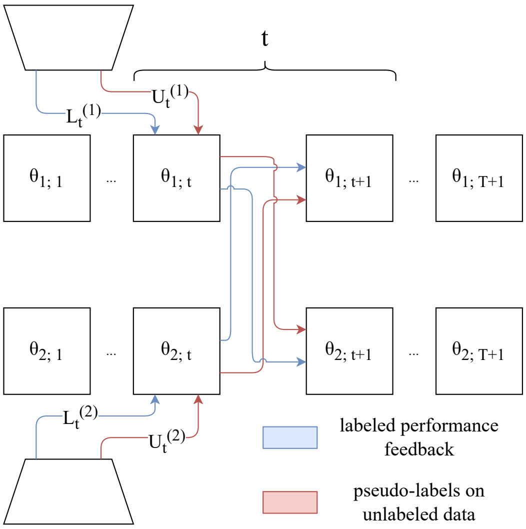

Figure 1: At each step $t \in \{1, \ldots, T\}$ of MCT the models that correspond to the so-far learnt parameters $\theta_{1;t}$ and $\theta_{2;t}$ play the role of the student and the teacher simultaneously using batches for their respective views. Pseudo-labeling occurs on complementary views so that 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.

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 1. 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 2 form different views that are used by MCT.

By training two models on two views to collaborate we 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 which results in an improvement. MCT trains models in a way which 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 which are as diverse as those which are trained from independent views.

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 we are sampling 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-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 6 since we approximate the student and teacher optimizations using alternating iterations of stochastic gradient descent. This is following the ideas of MPL and further discussion is available in [42, 21, 45].

Algorithm 1 Meta Co-Training (Section 2 has notation)

```
1:  Input: $L^{(1)}, L^{(2)}, U^{(1)}, U^{(2)}, f^{(1)}, f^{(2)}, \theta_1, \theta_2, T, \ell, \eta$
2:  for step $t \in \{1 \ldots T\}$ 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:      $\hat{y}^{(1)} \leftarrow f^{(1)}_{\theta_1}(U_t^{(1)})$
8:      $\hat{y}^{(2)} \leftarrow f^{(2)}_{\theta_2}(U_t^{(2)})$
9:      // Sample the pseudo-labels from the discrete distribution over
        the classes
10:     $PL^{(1)} \sim \hat{y}^{(1)}$
11:     $PL^{(2)} \sim \hat{y}^{(2)}$
12:     // MCT student update applied to both models
13:     $\theta'_1 \leftarrow \theta_1 - \eta \cdot \nabla_{\theta_1}\ell(PL^{(2)}, \hat{y}^{(1)})$
14:     $\theta'_2 \leftarrow \theta_2 - \eta \cdot \nabla_{\theta_2}\ell(PL^{(1)}, \hat{y}^{(2)})$
15:     // Sample batches from $L$ for the teacher updates
16:     $X_t^{(1)}, Y_t^{(1)} \leftarrow SampleBatch(L^{(1)})$
17:     $X_t^{(2)}, Y_t^{(2)} \leftarrow SampleBatch(L^{(2)})$
18:     // MCT teacher update applied to both models
19:     $\hat{y}'^{(1)} \leftarrow f^{(1)}_{\theta'_1}(X_t^{(1)})$
20:     $\hat{y}'^{(2)} \leftarrow f^{(2)}_{\theta'_2}(X_t^{(2)})$
21:     $h^{(1)} \leftarrow \nabla_{\theta'_2}\ell(Y_t^{(2)}, \hat{y}'^{(2)})^T \cdot \nabla_{\theta_2}\ell(PL^{(1)}, \hat{y}^{(2)})$
22:     $h^{(2)} \leftarrow \nabla_{\theta'_1}\ell(Y_t^{(1)}, \hat{y}'^{(1)})^T \cdot \nabla_{\theta_1}\ell(PL^{(2)}, \hat{y}^{(1)})$
23:     // Weights to be used in the next step
24:     $\theta_1 \leftarrow \theta'_1 - \eta \cdot h^{(1)} \cdot \nabla_{\theta_1}\ell(PL^{(1)}, \hat{y}^{(1)})$
25:     $\theta_2 \leftarrow \theta'_2 - \eta \cdot h^{(2)} \cdot \nabla_{\theta_2}\ell(PL^{(2)}, \hat{y}^{(2)})$
26: end for
```

Each student model evaluates the performance of their new weights on separate labeled batches. This performance provides feedback to the teacher model. Geometrically, we can understand this as updating the parameters in the direction of increasing the teacher's confidence on 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 our experiments in this work we use preexisting foundation models to construct views for our 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 $\nabla_{\theta_2}\ell(PL^{(2)}, \hat{y}^{(2)})$ and $\nabla_{\theta_1}\ell(PL^{(1)}, \hat{y}^{(1)})$ (used in Lines 24-25) first, then we compute (and apply) $\nabla_{\theta_1}\ell(PL^{(2)}, \hat{y}^{(1)})$ and $\nabla_{\theta_2}\ell(PL^{(1)}, \hat{y}^{(2)})$ (used first in Lines 13-14 and reused inLines 21-22), then we compute $\nabla_{\theta'_1}\ell(Y_t^{(1)}, \hat{y}'^{(1)})^T$ and $\nabla_{\theta'_2}\ell(Y_t^{(2)}, \hat{y}'^{(2)})^T$ which 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.

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

4 Experimental Evaluation

Towards evaluating our proposed method we compare to other self-supervised and semi-supervised learning algorithms using the ImageNet [17] classification benchmark, as well as Flowers102 [40], Food101 [6], FGVCAircraft [38], iNaturalist [28], and iNaturalist 2021 [29] datasets. To produce the views used to train the classifiers during CT and MCT we used the embedding spaces of five representation learning architectures: The Masked Autoencoder (MAE) [27], DINOv2 [41], SwAV [10], EsViT [34], and CLIP [44]. We selected the models which produce the views as 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. Hyperparameters used during training can be found in [45] for CT, MCT, and deep ensembles.

Table 1 provides a description of the different datasets used for training and testing, as well as the number of classes for each dataset. Note that in the 10% experiments on the Flowers102 dataset each class has only 1 label. All datasets are approximately class-balanced with the exception of the iNaturalist dataset. In all cases we maintain the original class distribution when sampling subsets. For ImageNet we use the split published with the SimCLRv2 [13] repository. All subsets are created with seeded randomness of a common seed (13) which ensures a fair comparison between methods.

Table 1: Characteristics of datasets used.

Dataset	#train	#test	#classes
FGVCAircraft	3333	3333	100
Flowers102	1020	6149	102
Food101	75,750	25,250	101
iNaturalist	175,489	29,083	1010
ImageNet	1,281,167	50,000	1000

Table 2: 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 3: Pairwise translation performance of linear probe on the ImageNet dataset. A linear classifier is trained on the output of an MLP which 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	-

4.1 Experimental Evaluation on ImageNet

As we discussed in Section 2 under the paragraph "View Construction", CT relies on two assumptions about the sufficiency and independence of the two views in our data. We conducted experiments to verify these assumptions. Below we give a summary of our results.

On the Sufficiency of the Views. Sufficiency is fairly easy to verify. If we choose a reasonable sufficiency threshold for the task, for ImageNet say close or above 75% top-1 accuracy, then we can train simple models on the views of the dataset we have constructed and provide examples of functions which demonstrate the satisfaction of the property. We tested a single linear layer with softmax output to provide a lower bound on the sufficiency of the views on the standard subsets of the ImageNet labels for semi-supervised classification.

A simple 3-layer multi-layer perceptron (MLP) with 1024 neurons per layer was evaluated (Table 2). These were the same models as those used for CT and MCT. From these experiments, whose results are shown in Table 2 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.

On the Independence of the Views. To test the independence of the views generated by the representation learning models, we trained an identical MLP architecture to predict each view from each other view; Table 3 presents our findings, explained below. Given as input the view to predict from (rows of Table 3) the MLPs were trained to reduce the mean squared error (MSE) between their output and the view to be predicted (columns of Table 3). We then trained a linear classifier 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 4. The linear classifiers never did much better than a random guess on the ImageNet class achieving at most 0.156% accuracy. Thus, while we cannot immediately conclude that the views are independent, it is clearly not trivial to predict one view given any other. We believe that this is compelling evidence for the independence of different representations.

Having constructed at least two strong views of the data, and suspecting that these views are independent we hypothesize that MCT and CT will work on this data.

Co-Training Baseline Experiments. At each iteration of CT the models made predictions for all instances in U. We assigned labels

Table 4: 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

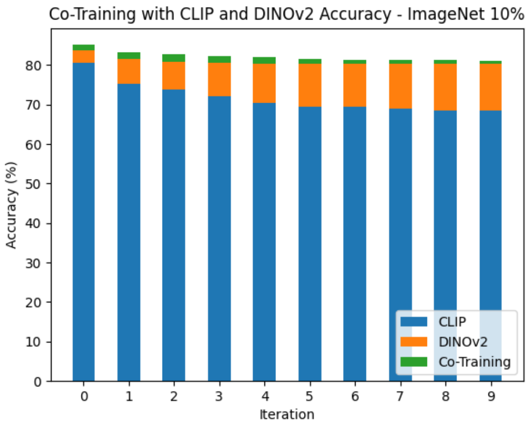

Figure 2: Top-1 accuracy of CT iterations on the CLIP and DINOv2 views for the ImageNet 10% dataset.

to the 10% of the most confident predictions in U for both models. When confident predictions conflicted on examples, we returned those instances 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 our experiments on ImageNet, as CT iterations proceeded, top-1 accuracy decreased. Later we show that this was not always the case (see Figure 4). The decrease was less pronounced when the views perform at a similar level; see Figure 2. In contrast MCT does not exhibit performance degradation after warmup. Figures 2 and 3 show results on ImageNet 10%. We saw that 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. 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, we believe the poor performance of the pseudo-labeling step in CT was due to imbalanced information content between views rather than view-dependence.

Meta Co-Training Experiments. To draw fair comparisons, we fixed the model architecture and view set from our experiments on CT. As in MPL [42] a supervised loss was optimized jointly with a loss on pseudo-labels. We 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 [45].

Table 6 shows the results of MCT on all of the possible combinations of views we 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 6a compared to MLP accuracy shown in Table 2, we observed 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. In Table 6c we showed the performance of an ensemble of models trained on concatenated view pairs. We provide more information on the comparison between MCT and deep ensembles in a separate paragraph below.

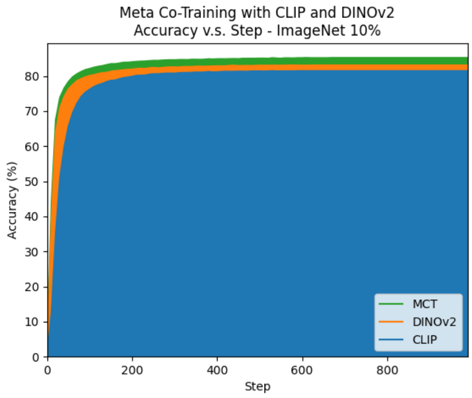

Figure 3: MCT using the CLIP and DINOv2 views as a function of the training step. Models are trained on 10% of the ImageNet labels.

MCT and CT are compared on ImageNet 10% (resp., 1%) in Table 5. We observed that MCT top-1 accuracy is 0.7% (resp., 0.6%) higher overall. An ablation study for the views is given in Table 6. On all view pairs in which MCT or CT provided no benefit over a single view are cases in which one of the views was significantly weaker than the other. This makes sense as one model has nothing to learn from the other. For the experiments in which we had two strong views, MCT always outperformed CT.

Comparison to Ensemble Methods. Ensemble methods are common in practice to boost the performance of supervised models. To show that our method provided benefit outside of just adding an additional model, we compared to a Deep Ensemble [32] of five models for each individual view. In Table 7 we show the performance of ensembles trained on the individual views. We show the performance of an ensemble trained on each view pair concatenated in Table 6c. We show the performance of an ensemble trained on the concatenation of all four views we used to achieve our best performing MCT experiment in Table 8.

In some cases the CT models performed better than the ensemble of models which all have the same input for each sample (the concatenation of the CT views). We believe that CT before pseudo-labeling (which is essentially an ensemble of two different views) out-performed the deep ensembles because the CT models are fewer but they are more diverse. The CT models were more diverse as they were trained on different views, while the ensemble models are trained on the *concatenation* of the views. The concatenation has the same information but may not have produced meaningfully different models in an ensemble. Ensuring diversity of ensemble members is an important problem for ensembling methods, so it seems likely that a less diverse ensemble would perform worse. If this is indeed the case, then it would support our hypothesis that CT failed to im-

Table 5: Performance of different approaches on ImageNet dataset. An asterisk (*) indicates models that were trained on top of pre-trained frozen backbone models. Models on the lower half of the table use unlabeled data during classification training.

Reference	Model	Method	ImageNet-1%	ImageNet-10%
[34]	EsViT (Swin-B, W=14)	Linear	69.1	74.4
[27]	MAE (ViT-L)	MLP	23.4	48.5
[44]	CLIP (ViT-L)	Fine-tuned	80.5	84.7
[41]	DINOv2 (ViT-L)	Linear	78.1	82.9
[13]	SimCLRv2 (ResNet152-w2)	Fine-tuned	74.2	79.4
[10]	SwAV (RN50-w4)	Fine-tuned	53.9	70.2
[43]	Deep Co-Training (ResNet-18)	Co-Training	-	53.5
[51]	UDA (ResNet50)	UDA	-	68.78
[49]	FixMatch (ResNet-50)	FixMatch	-	71.46
[42]	MPL (EfficientNet-B6-Wide)	MPL	-	73.9
[9]	Semi-ViT (ViT-L)	Self-trained	77.3	83.3
[36]	REACT (ViT-L)	REACT	**81.6**	85.1
[5]	Co-Training (MLP)*	Co-Training	80.1	85.1
[32]	Deep Ensemble *	Deep Ensemble	80.0	84.3
	Meta Co-Training (MLP)* - ours	Meta Co-Training	80.7	**85.8**

Table 6: 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, we show only all unique combinations. 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)

Table 7: MLP ensemble performance on individual views. Top-1 accuracy on different subsets of the ImageNet data are shown.

Model	1%	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 8: 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.

Method	1%	10%
Deep Ensemble	80.0	84.3
Co-Training	80.0	84.8
Meta Co-Training	**80.5**	**85.8**

prove after pseudo-labeling due to an imbalance in the sufficiency of views and not a lack of independence. MCT provided performance gains over CT and the deep ensembles, so we can be confident that these gains cannot be attributed solely to the use of multiple models.

4.2 Stronger Views by Concatenation

Finally, we took the four views which had the greatest performance and constructed two views out of them by concatenating them together. We constructed one view CLIP | EsViT and the second DINOv2 | SwAV and measured the performance of CT and MCT on these views in Table 8. Interestingly, CT did not benefit from these larger views; see Table 8.

The additional information may have allowed the individual models to overfit their training data quickly. During MCT we observed that after one model reached 100% training accuracy, its validation accuracy still improved by learning to label for the other model. Possibly due to the rapid overfitting due to the increased view size (and consequently increased parameter count) the 1% split did not show improvement for either model.

4.3 Experiments on Additional Datasets

We compared MCT and CT to existing SoTA approaches that leverage unlabeled data directly during training (e.g., not just part of a pretext task). These experiments supported our hypothesis 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. Tables 9 and 10 provide more information on these additional datasets. Furthermore, Figure 4 provides an example where CT performed as expected and accuracy increased across CT iterations. In all but one case MCT outperforms the SoTA top-1 accuracy. The main takeaway from these additional experiments, apart from establishing new SoTA results in datasets beyond Ima-

Table 9: 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.

	Dataset		
Method	Flowers102	Food101	FGVCAircraft
REACT (ViT-L)	97.0	85.6	**57.1**
Co-Training (MLP)	99.2	94.7	36.4
Meta Co-Training (MLP)	**99.6**	**94.8**	40.1

Table 10: Additional comparisons to Semi-ViT using 10% (1%) of the available labels for training. For the iNaturalist task we used only the 1010 most frequent classes.

	Dataset	
Method	iNaturalist	Food101
Semi-ViT (ViT-B)	67.7 (32.3)	84.5 (60.9)
Co-Training (MLP)	59.7 (29.5)	94.7 (83.9)
Meta Co-Training (MLP)	**76.0** (**58.1**)	**94.8** (**91.7**)

geNet, is that MCT was in general more robust to view imbalance and provided better results than CT. The performance degradation of co-training beyond the initial warmup period was not something we expected. In [45] we provide full details on CT results including charts similar to Figure 4 which instead exhibit that degradation.

5 Related Work

While SSL has become very relevant in recent years with the influx of the data outpacing the availability of human labels, the paradigm and its usefulness have been studied at least since the 1960's; e.g., [25, 16, 48, 22]. Among the main approaches for SSL that are not directly related to our work, one popular direction encourages entropy minimization and consistency regularization using data augmentation such as MixMatch [4], UDA [51], ReMixMatch [3], FixMatch [49], and FlexMatch [53].

Self-Training and Student-Teacher Frameworks. One of the earliest ideas for SSL is that of *self-training* [48, 22]. This idea is still very popular with different *student-teacher* frameworks [55, 52, 42, 35, 24, 9]. This technique is also frequently referred to as *pseudo-labeling* [33] as the student tries to improve its performance by making predictions on the unlabeled data, then integrating the most confident predictions to the labeled set, and subsequently retraining using the broader labeled set. In addition, one can combine ideas; e.g., self-training with clustering [30], or create ensembles via self-training [31], potentially using graph-based information [37]. An issue with these approaches is that of *confirmation bias* [1], where the student cannot improve a lot due to incorrect labeling that occurs either because of self-training, or due to a fallible teacher.

Co-training Algorithms. Co-training was introduced by Blum and Mitchell [5] and their paradigmatic example was classifying web pages using two different views: *(i)* the *bags of words* based on the content that the webpages had, and *(ii)* the bag of words formed by the words used in hyperlinks pointing to these webpages. In our context the two different views are obtained via different *embeddings* that we obtain when we use images as inputs to different foundation models and observe the resulting embeddings.

Co-training has been very successful with wide applicability [20] and can outperform learning algorithms that use only single-view data [39]. Furthermore, the generalization ability of co-training is well-motivated theoretically [5, 15, 2, 14]. There are several natural extensions of the base idea, such as methods for constructing different views [50, 8, 11, 7], including connections to active learning [19], methods that are specific to deep learning [43], and other extensions that exploit more than two views on data; e.g., [54].

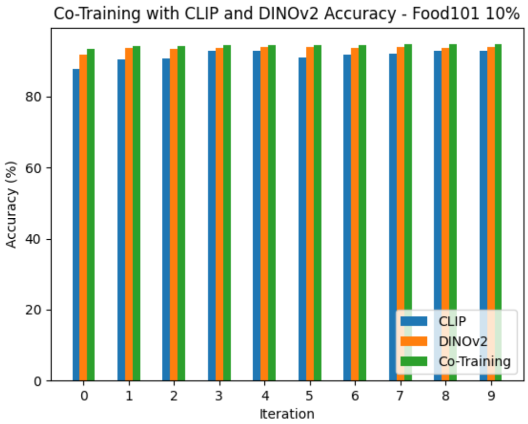

Figure 4: The top-1 accuracy of the CT predictions for each iteration of CT. 10% of available Food101 labels were used for training. The method exhibited performance improvement over multiple iterations of pseudo-labeling and retraining.

6 Conclusion and Future Work

We proposed MCT, a novel semi-supervised learning algorithm. We showed that our algorithm outperforms co-training in general and establishes a new SoTA top-1 classification accuracy on the ImageNet 10%. We also tested the performance of deep ensembles on ImageNet 10% and 1%. In both cases MCT achieved better results, demonstrating that MCT provides utility beyond what can be attributed to ensembling. In addition, using MCT, we were able to achieve new SoTA few-shot top-1 classification accuracy on other popular CV benchmark datasets, including one-shot top-1 classification accuracy for the Flowers102 dataset. Furthermore, we presented evidence that self-supervised learned representations are good candidates for views. We showed that the views constructed this way likely contain independent information and that models trained on these views, or concatenation of these views, can complement each other to produce stronger classifiers.

Our experiments indicate that our method performs best when it has access to enough labeled data to produce strong views for pseudo-labeling. For some datasets this may preclude its application in one-shot settings. Further, if no sufficient and independent views are available and none can be constructed then we do not expect our model to perform meaningfully better than MPL.

For future work we are interested in fine-tuning the frozen backbones that we used for CT and MCT. Furthermore, we did not make use of the large available unlabeled datasets for CV in this study. We are also interested in evaluating CT and MCT using unlabeled data collected "in the wild" which may not follow the class distribution of the labeled dataset.

Acknowledgements

This material is based upon work supported by the U.S. National Science Foundation under Grant No. RISE-2019758. This work is part of the NSF AI Institute for Research on Trustworthy AI in Weather, Climate, and Coastal Oceanography (NSF AI2ES).

Some of the computing for this project was performed at the OU Supercomputing Center for Education & Research (OSCER) at the University of Oklahoma (OU).

References

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

[2] M. Balcan, A. Blum, and K. Yang. Co-Training and Expansion: Towards Bridging Theory and Practice. In *NeurIPS*, pages 89–96, 2004.

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

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

[5] A. Blum and T. M. Mitchell. Combining Labeled and Unlabeled Data with Co-Training. In *COLT*, pages 92–100. ACM, 1998.

[6] L. Bossard, M. Guillaumin, and L. V. Gool. Food-101 - Mining Discriminative Components with Random Forests. In *ECCV*, pages 446–461, 2014.

[7] U. Brefeld and T. Scheffer. Co-EM support vector learning. In *ICML*. ACM, 2004.

[8] U. Brefeld, C. Büscher, and T. Scheffer. Multi-view Discriminative Sequential Learning. In *ECML*, pages 60–71. Springer, 2005.

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

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

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

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

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

[14] M. Darnstädt, H. U. Simon, and B. Szörényi. Supervised learning and Co-training. *Theor. Comput. Sci.*, 519:68–87, 2014.

[15] S. Dasgupta, M. L. Littman, and D. A. McAllester. PAC Generalization Bounds for Co-training. In *NeurIPS*, pages 375–382. MIT Press, 2001.

[16] N. E. Day. Estimating the components of a mixture of normal distributions. *Biometrika*, 56(3):463–474, 1969.

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

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

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

[20] J. Du, C. X. Ling, and Z.-H. Zhou. When Does Cotraining Work in Real Data? *IEEE Trans. on Knowledge and Data Engineering*, 23(5): 788–799, 2011.

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

[22] S. Fralick. Learning to recognize patterns without a teacher. *IEEE Trans. on Inf. Th.*, 13(1):57–64, 1967.

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

[24] X. Gu. A self-training hierarchical prototype-based approach for semi-supervised classification. *Inf. Sci.*, 535:204–224, 2020.

[25] H. O. Hartley and J. N. Rao. Classification and estimation in analysis of variance problems. *Revue de l'Inst. Intl. de Stat.*, pages 141–147, 1968.

[26] K. He, H. Fan, Y. Wu, S. Xie, and R. B. Girshick. Momentum Contrast for Unsupervised Visual Representation Learning. In *CVPR*, pages 9726–9735. Computer Vision Foundation / IEEE, 2020.

[27] K. He, X. Chen, S. Xie, Y. Li, P. Dollár, and R. B. Girshick. Masked Autoencoders Are Scalable Vision Learners. In *CVPR*, pages 15979–15988. IEEE, 2022.

[28] G. V. Horn and et al. The INaturalist Species Classification and Detection Dataset. In *CVPR*, pages 8769–8778, 2018.

[29] G. V. Horn, E. Cole, S. Beery, K. Wilber, S. J. Belongie, and O. M. Aodha. Benchmarking Representation Learning for Natural World Image Collections. In *CVPR*, pages 12884–12893, 2021.

[30] S. A. Koohpayegani, A. Tejankar, and H. Pirsiavash. Mean Shift for Self-Supervised Learning. In *ICCV*, pages 10306–10315. IEEE, 2021.

[31] S. Laine and T. Aila. Temporal Ensembling for Semi-Supervised Learning. In *ICLR*, 2017.

[32] B. Lakshminarayanan, A. Pritzel, and C. Blundell. Simple and Scalable Predictive Uncertainty Estimation using Deep Ensembles. In *NeurIPS*, pages 6402–6413, 2017. URL https://proceedings.neurips.cc/paper/2017/hash/9ef2ed4b7fd2c810847ffa5fa85bce38-Abstract.html.

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

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

[35] X. Li and et al. Learning to Self-Train for Semi-Supervised Few-Shot Classification. In *NeurIPS*, pages 10276–10286, 2019.

[36] H. Liu, K. Son, J. Yang, C. Liu, J. Gao, Y. J. Lee, and C. Li. Learning Customized Visual Models with Retrieval-Augmented Knowledge. In *CVPR*, pages 15148–15158. IEEE, 2023.

[37] Y. Luo, J. Zhu, M. Li, Y. Ren, and B. Zhang. Smooth Neighbors on Teacher Graphs for Semi-Supervised Learning. In *CVPR*, pages 8896–8905, 2018.

[38] S. Maji, J. Kannala, E. Rahtu, M. Blaschko, and A. Vedaldi. Fine-Grained Visual Classification of Aircraft. Technical report, VGG, University of Oxford, 2013.

[39] K. Nigam and R. Ghani. Analyzing the Effectiveness and Applicability of Co-training. In *CIKM*, pages 86–93. ACM, 2000.

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

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

[42] H. Pham, Z. Dai, Q. Xie, and Q. V. Le. Meta Pseudo Labels. In *CVPR*, pages 11557–11568, 2021.

[43] S. Qiao, W. Shen, Z. Zhang, B. Wang, and A. L. Yuille. Deep Co-Training for Semi-Supervised Image Recognition. In *ECCV*, pages 142–159, 2018.

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

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

[46] C. Schuhmann and et al. LAION-400M: Open Dataset of CLIP-Filtered 400 Million Image-Text Pairs. In *Data Centric AI NeurIPS Workshop*, 2021.

[47] C. Schuhmann and et al. LAION-5B: An open large-scale dataset for training next generation image-text models. In *NeurIPS*, 2022.

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

[49] K. Sohn, D. Berthelot, N. Carlini, Z. Zhang, H. Zhang, C. Raffel, E. D. Cubuk, A. Kurakin, and C. Li. FixMatch: Simplifying Semi-Supervised Learning with Consistency and Confidence. In *NeurIPS*, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/06964dce9addb1c5cb5d6e3d9838f733-Abstract.html.

[50] J. Wang, S. Luo, and X. Zeng. A random subspace method for co-training. In *IJCNN*, pages 195–200. IEEE, 2008.

[51] Q. Xie, Z. Dai, E. H. Hovy, T. Luong, and Q. Le. Unsupervised Data Augmentation for Consistency Training. In *NeurIPS*, 2020.

[52] Q. Xie, M. Luong, E. H. Hovy, and Q. V. Le. Self-Training With Noisy Student Improves ImageNet Classification. In *CVPR*, pages 10684–10695, 2020.

[53] B. Zhang, Y. Wang, W. Hou, H. Wu, J. Wang, M. Okumura, and T. Shinozaki. FlexMatch: Boosting Semi-Supervised Learning with Curriculum Pseudo Labeling. In *NeurIPS*, pages 18408–18419, 2021. URL https://proceedings.neurips.cc/paper/2021/hash/995693c15f439e3d189b06e89d145dd5-Abstract.html.

[54] Z. Zhou and M. Li. Tri-Training: Exploiting Unlabeled Data Using Three Classifiers. *IEEE Trans. Knowl. Data Eng.*, 17(11):1529–1541, 2005.

[55] B. Zoph, G. Ghiasi, T. Lin, Y. Cui, H. Liu, E. D. Cubuk, and Q. Le. Rethinking Pre-training and Self-training. In *NeurIPS*, 2020.