DEEP LEARNING WITH VERY FEW AND NO LABELS

A Dissertation presented to

the Faculty of the Graduate School

at the University of Missouri

In Partial Fulfillment

of the Requirements for the Degree

Doctor of Philosophy

---

by

YANG LI

Dr. Zhihai He, Dissertation Supervisor

Dec 2021

The undersigned, appointed by the Dean of the Graduate School, have examined the dissertation entitled:

DEEP LEARNING WITH VERY FEW AND NO LABELS

presented by Yang Li,

a candidate for the degree of Doctor of Philosophy and hereby certify that, in their opinion, it is worthy of acceptance.

Dr. Zhihai He

Dr. Ye Duan

Dr. Filiz Bunyak

Dr. Ming Xin

ACKNOWLEDGMENTS

First and foremost, I would like to sincerely thank my advisor, Dr. Zhihai He, who guided me into the research community and continued to provide me with guidance and support. I am very grateful to Dr. He for his professionalism and encouragement in my graduate research and study. I would also like to express my deep appreciation to Dr. Ye Duan, Dr. Filiz Bunyak, and Dr. Ming Xin, for serving on my thesis committee and sharing their professional advice and feedback.

I would also like to thank all the friends and colleagues in Mizzou, Seattle, and Silicon Valley, for their help and inspiration.

Last but not least, a special thank goes to my dear parents, for their love and support.

TABLE OF CONTENTS

LIST OF TABLES

Table Page

LIST OF FIGURES

Figure Page

ABSTRACT

Deep neural networks have achieved remarkable performance in many computer vision applications such as image classification, object detection, instance segmentation, image retrieval, and person re-identification. However, to achieve the desired performance, deep neural networks often need a tremendously large set of labeled training samples to learn its huge network model. Labeling a large dataset is labor-intensive, time-consuming, and sometimes requiring expert knowledge. In this research, we study the following important question: how to train deep neural networks with very few or even no labeled samples? This leads to our research tasks in the following two major areas: semi-supervised and unsupervised learning.

Specifically, for semi-supervised learning, we developed two major approaches. The first one is the *Snowball approach* which learns a deep neural network from very few samples based on iterative model evolution and confident sample discovery. The second one is the *learned model composition approach* which composes more efficient master networks from student models of past iterations through a network learning process. Critical sample discovery is developed to discover new critical unlabeled samples near the model decision boundary and provide the master model with look-ahead access to these samples to enhance its guidance capability.

For unsupervised learning, we have explored two major ideas. The first idea is *transformed attention consistency* where the network is learned based on self-supervision information across images instead of within one single image. The second one is *spatial assembly networks* for image representation learning. We introduce a new learnable module, called spatial assembly network (SAN), which performs a

learned re-organization and assembly of feature points and improves the network capabilities in handling spatial variations and structural changes of the image scene.

Our experimental results on benchmark datasets demonstrate that our proposed methods have significantly improved the state-of-the-art in semi-supervised and unsupervised learning, outperforming existing methods by large margins.

Chapter 1

Introduction

1.1 Motivation

Lacking of sufficient labeled data limits the performance and application of deep neural networks. Labeling large and frequently changing datasets requires significant human efforts and is costly. Furthermore, providing accurate labels for domain-specific image datasets, such as biological and medical images, requires expert knowledge. Obtaining large-scale labeled training sets for these domain-specific tasks is difficult. Meanwhile, in many applications and real-world problems, unlabeled samples are often massively and easily available, for example, images obtained from the web or social networks. Recently, training an efficient deep neural network using a very small set of labeled samples or even no labeled samples has emerged as an important research topic in deep learning with a wide range of applications in image classification, object detection, instance segmentation, image retrieval, and person re-identification.

Learning from very few labeled data or unlabeled data is very challenging, yet highly desirable in practice. This leads to our research topics in the following two major areas: semi-supervised learning and unsupervised learning.

1.2 Semi-Supervised Learning

Semi-supervised learning aims to train a deep neural network with a small set of labeled samples and a large set of unlabeled samples together [1, 2]. During the past several years, a number of semi-supervised learning algorithms have been developed, including regularization-based methods [1, 2], graph-based methods [3, 4, 5, 6, 7], and Generative Adversarial Networks (GANs)-based methods [8, 9, 10, 11]. Oliver *et al.* [12] provide a comprehensive survey of recent semi-supervised learning methods. In recent approaches for semi-supervised learning aiming to achieve better model generalization for the unseen data, a loss function is often computed on the prediction of unlabeled samples based on the following three principles: (1) *entropy minimization* which encourages the model to output high confident (low entropy) predictions on unlabeled data [13, 14]; (2) *consistency regularization* which encourages the model to produce the same output distribution when its inputs are perturbed [15, 16, 17]; and (3) *generic regularization* which encourages the model to generalize well and avoid over-fitting of the training data [18]. For example, the Mean-Teacher [16] algorithm constructs a teacher model based on the exponential moving average (EMA) of student models obtained from previous training steps to guide the training of the current student model by enforcing the prediction consistency between them on unlabeled samples. MixMatch [19] designs a unified loss function which combines the

techniques of consistency regularization [20, 15, 16] and entropy minimization [13]. The goal of semi-supervised learning is to successfully train the network with fewer and fewer labeled samples.

1.3 Unsupervised Learning

Clustering is one of the earliest methods developed for unsupervised learning. Recently, motivated by the remarkable success of deep learning, researchers have started to develop unsupervised learning methods using deep neural networks [21]. Auto-encoder trains an encoder deep neural network to output feature representations with sufficient information to reconstruct input images by a paired decoder [22]. As we know, during deep neural network training, the network model is updated and learned in an iterative and progressive manner so that the network output can match the target. In other works, deep neural networks need human supervision to provide ground-truth labels. However, in unsupervised learning and recent self-supervised learning, there are no labels available. To address this issue, researchers have exploited the unique characteristics of images and videos to create various self-supervised learning pretext task labels, objective functions, or loss functions, which essentially convert the unsupervised learning into a supervised one so that the deep neural networks can be successfully trained. For example, in DeepCluster [21], clustering is used to generate pseudo labels for images. Various supervised learning methods have been developed to train networks to predict the relative position of two randomly sampled patches [23], solve Jigsaw image puzzles [24], predict pixel values of missing image patches [25], classify image rotations of four discrete angles [26], reconstruct image

transforms [22], etc. Once successfully trained by these pretext tasks, the baseline network should be able to generate discriminative features for subsequent tasks, such as image retrieval, classification, matching, etc [27].

In this dissertation, we mainly use unsupervised deep metric learning to evaluate our proposed methods. Deep metric learning aims to learn discriminative features that can aggregate visually similar images into compact clusters in the high-dimensional feature space while separating images of different classes from each other. In supervised deep metric learning, we assume that the labels for training data are available. In unsupervised deep metric learning task, we consider unsupervised deep metric learning where the image labels are not available. It has many important applications, including image retrieval [28, 29, 30], face recognition [31], and person re-identification [32, 33]. Successful metric learning needs to achieve the following objectives: (1) *Discriminative*. It should be able to aggregate images with the same semantic labels into compact clusters in the high-dimensional feature space while separating images of different classes from each other. (2) *Generalizable*. The learned features should be able to generalize well from the training images to test images of new classes which have not been seen before. Learning directly and automatically from images in an unsupervised manner without human supervision represents a very important yet challenging task in computer vision and machine learning.

1.4 Dissertation Organization

The rest of the paper is organized as follows. In **Chapter 2**, we introduce a joint sample discovery and iterative model evolution method for semi-supervised learning on

small labeled training sets. In **Chapter 3**, we propose to push the performance limit of semi-supervised learning on very small sets of labeled samples by developing a new method called learned model composition with critical sample look-ahead (LMCS). In **Chapter 4**, we propose the transformed attention loss and contrastive clustering loss for unsupervised deep metric learning. In **Chapter 5**, we introduce a learnable spatial assembly network (SAN) for supervised and unsupervised representation learning. In **Chapter 6**, We summarize all the works in the dissertation.

Chapter 2

Snowball: Iterative Model Evolution and Confident Sample Discovery for Semi-Supervised Learning

2.1 Introduction

Most recent semi-supervised learning algorithms[15, 34, 16] which achieve the state of the art performance are based on the principle of consistency regularization, which aims to learn a smooth manifold on the labeled and unlabeled samples [35]. The Mean-Teacher [16] algorithm constructs a teacher model based on the exponential moving average (EMA) of student models obtained from previous training steps to guide the training of the current student model by enforcing the prediction consistency between them on unlabeled samples. In this work, we propose to push the performance limit of semi-supervised learning by developing an efficient learning method on

very small sets of labeled samples. For example, on the same CIFAR-10 dataset, we can achieve successful training with 250 labeled samples while outperforming existing methods on learning with larger sets of labeled samples. On the Street View House Number (SVHN) dataset, we can reduce the training set size from 1000 in existing literature to 100 while maintaining efficient learning performance. Our extensive experimental results demonstrate that our proposed method significantly improves the overall semi-supervised learning performance, outperforming existing state-of-the-art method by a large margin. For example, on the CIFAR-10 dataset, our proposed method has successfully trained a model with 250 labeled samples to achieve an error rate of 11.58%, about 38% lower than the Mean-Teacher (49.91%). We find out that our proposed master-teacher-student framework and sample discovery, once coupled with the augmentation and loss function of the state-of-the-art MixMatch [19], can successfully train the network with only 100 labeled samples and gain significantly improvement.

To achieve this goal, we propose to explore two major ideas: (1) we extend the existing Mean-Teacher method by introducing a master-teacher-student network to provide multi-layer guidance during the model evolution process with multiple iterations and generations. This master network combines the knowledge of the student network and teacher network with additional access to newly discovered samples. Both the master and teacher models are then used to guide the training of the student network by enforcing the prediction consistency between them on unlabeled samples. The student model learns gradually from stable and reliable generated targets from the more powerful master model, which makes significant contributions to the approach of consistency regularization. (2) We develop a confident sample

discovery method and couple it with the master-teacher-student learning to achieve continuous model evolution with more and more samples being discovered. The error rate of sample discovery depends on the performance of the initial model. In self-training [36, 37, 38], the newly selected samples often have high error rates in their label prediction, since the initial model is not accurate, being trained with very few samples. Our method is able to significantly reduce the label prediction error rate in the discovered samples using the master-teacher-student network. For example, on the CIFAR-10 dataset with 500 labeled samples for network training, the error rate of sample discovery reaches below 1%.

Our main contributions can be summarized as follows:

(1) We introduce a new master-teacher-student structure, offers a multi-layer guidance during the model evolution process with multiple iterations and generation.

(2) We develop confident sample discovery method and couple it with the master-teacher-student learning to achieve continuous model evolution with more and more samples being discovered. For this reason, we refer to our method as Snowball.

(3) Our experimental results and ablation studies show that the proposed Snowball method outperforms the state-of-the-art semi-supervised learning methods on multiple benchmarks, especially on very small sets of labeled samples.

The rest of the chapter is organized as follows. In Section 2.2, we provide a review of recent work on semi-supervised learning. The proposed Snowball method is presented in Section 2.3. Experimental results are in Section 2.4. Section 2.4 concludes the chapter.

2.2 Related Work

Training semi-supervised models with a small set of labeled samples and a large set of unlabeled samples has become an important research task with significant impact in practice. A number of methods have been developed and achieved remarkable results. These methods can be summarized into the following five major categories.

(1) Self-training methods. The self-training approach [36, 37, 38, 39], also called bootstrapping or self-teaching, first trains a classifier with labeled samples and then uses the pre-trained classifier to classify unlabeled samples, selects the most confident unlabeled samples, predicts their labels, and uses them for the next iteration of training [40]. The bootstrap aggregating method in [41] trains multiple networks independently based on different training subsets, which can make the model more stable than single networks.

(2) Co-training methods. Co-training [42, 43] trains two separate classifiers to learn features on two separate datasets. Each classifier is then used to classify unlabeled samples and retrained using highly confident unlabeled samples predicted by the other classifier. Deep co-training [44] combines the co-training and deep learning. It generates adversarial samples for each model to ensure the view difference. Dual-student [17] designs stabilization constraint which explores stable samples and exchanges reliable information between models.

(3) Graph-based methods. Graph-based semi-supervised algorithms assume that neighboring nodes share similar labels [40]. A graph is constructed to measure the similarity between labeled and unlabeled samples. Knowledge learned from the labeled samples is propagated along the graph to predict the labels of unlabeled samples [45, 4]. The manifold regularization method in [35] applies the Reproducing

Kernel Hilbert Space (RKHS) to a parameterized classifier with squared loss or hinge loss. The label propagation method in [46] compares unlabeled samples with labeled samples by selecting a suitable predefined distance metric. Multi-label propagation [47] extends this method to image annotation tasks. [48] uses the idea of self-training and proposes a graph-based label propagation method. [49] proposed a cross-task network which has two streams to jointly learn two tasks: classification and clustering. Based on the model predictions, a large number of pairwise constraints can be generated from unlabeled images, and are fed to the clustering stream. They used pairwise constraints to encode weak supervision information. Unlabeled images are weighted according to the distances to the clusters discovered, and an improved model is then trained on the classification stream associated with a weighted softmax loss.

(4) GANs-based methods. Another group of methods for semi-supervised learning is based on the generative adversarial networks (GANs) [10, 11]. [8] proposed an auto-encoder generative model for semi-supervised learning. GAN-based methods often require sophisticated tuning of network hyperparameters. [9] demonstrated the effectiveness of random perturbations in semi-supervised learning. [50] used adversarial training to discover the most sensitive perturbation for the labels of input samples. [34] introduced the method of virtual adversarial training (VAT). They proposed a new measure for local smoothness of the conditional label distribution. Unlike adversarial training, their method defines the adversarial direction without label information and is hence applicable to semi-supervised learning.

(5) Entropy minimization-based methods. Entropy minimization for semi-supervised learning [13] is based on a fundamental assumption that the decision boundary should lie in the low-density regions of data distribution [51]. This al-

lows us to utilize unlabeled samples by introducing a loss function based on this assumption. [52] coupled entropy minimization with self-training to label the unlabeled samples. Pseudo-labeling [14] uses the classifier itself to construct hard labels over the training process. It trains the classifier only with the labeled samples and gradually introduces the weighted pseudo-labels with larger class probabilities for unlabeled samples. Unlabeled samples and corresponding pseudo-labels are used in the standard loss function. [53] follows similar principle and uses deep metric embedding to measure the distance between the labeled and unlabeled samples.

(6) Consistency regularization-based methods. Many recent semi-supervised methods are based on consistency regularization which aim to ensure consistent model prediction over different sample perturbations. It often applies a consistency constraint on the teacher-student network to learn knowledge from perturbed data. The Γ-model in Ladder networks [54] trains a clean model as the teacher and a noisy model as the student, and enforces the student to predict the same target provided by teacher. Transform loss [20] and Π-model [15] adopt a similar principle by introducing new loss functions for unlabeled samples which penalize inconsistent predictions. In Π-model, teacher and student share network parameters and each sample is analyzed twice with different random noise. Temporal ensembling [15] improves upon the Π-model method by introducing an exponential moving average of predictions obtained from previous epochs. It predicts unlabeled samples over multiple epochs instead of labeling them with an external model. Deep Coupled Ensemble (DCE) [55] extends temporal ensembing and combines multiple complementary consistency regularizations. It introduces class-wise feature matching and a conditional entropy term to explore and weight unlabeled samples in the training process.

Instead of sharing parameters with different models and averaging of their predictions, the Mean-Teacher algorithm [16] uses an exponential moving average of the student models obtained from previous training steps. The student and teacher models improve each other in an iterative manner. Stochastic Weight Average (SWA) [56] averages weights with the trajectory of SGD to build a more powerful teacher. The Smooth Neighbors on Teacher Graph (SNTG) method [57] constructs a graph-based prediction of the teacher model to make the learned features more discriminative by exploring intra-class similarity and inter-class dissimilarity. Complementary Correction Network (CCN) [58] is constructed on top of two essential networks. It learns complementary knowledge from the output of the one network and the feature from the other network and then tranfers the knowledge via mutual learning. Interpolation Consistency Training (ICT) [59] uses the MixUp approach [18] to produce interpolations from two unlabeled samples and applies consistency regularization between them.

The most recent semi-supervised learning method is Google's MixMatch [19]. It designs a unified loss function which combines the techniques of consistency regularization [20, 15, 16] and entropy minimization [13]. It uses the MixUp approach [18] to mix the labeled samples and unlabeled samples randomly. The MixMatch [19] achieves the state-of-the-art performance in semi-supervised learning. Compared to the current consistency regularization-based methods, our Snowball method has the following unique and important characteristics: (1) The training process is guided by the more powerful master network. It enforces the consistency between the predictions of unlabeled samples by the master network and student network and explores more stable targets for unlabeled samples on very small labeled datasets. (2) Our

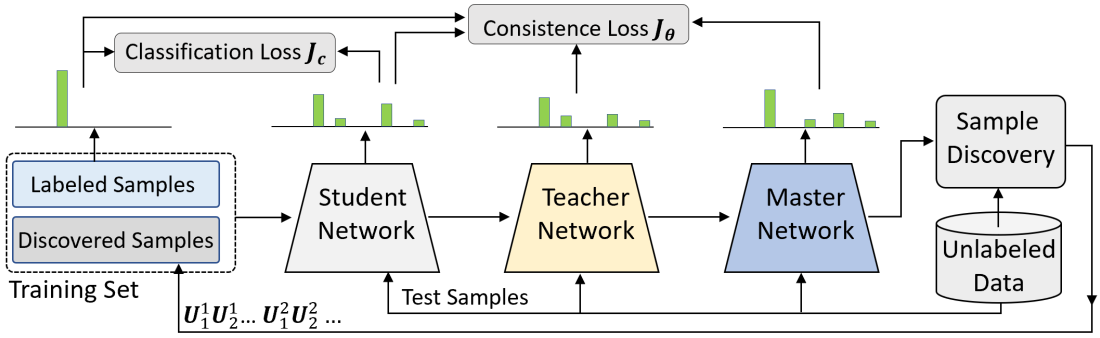


Figure 2.1: Overview of the proposed Snowball method.

sample discovery discovers high confident unlabeled samples and assigns hard labels instead of soft labels or weight for them. In this way, it can reduce the uncertainty of the model for the unlabeled samples.

With semi-supervised learning methods, we are able to train an efficient network using a very small set of labeled samples and a large set of unlabeled samples. In this work, we propose to push the performance limit of semi-supervised learning so that successfully learning on very small training set becomes possible.

In the section, we present our iterative model evolution and confident sample discovery for semi-supervised learning.

2.3 The Proposed Snowball Method

2.3.1 Method Overview

Figure 2.1 provides an overview of the proposed Snowball method. In semi-supervised learning, we have access to a small set of labeled training samples, denoted by Ω_L and a large set of unlabeled samples, denoted by Ω_U. Based on Ω_L, we follow the

Mean-Teacher method in [16] to train the student network $\mathbf{G}_S$ guided by the teacher network $\mathbf{G}_T$. We use this successfully trained student model $\mathbf{G}_S$ to analyze unlabeled images, extract their features, compare them against the labeled images, and discover a subset of confident samples with assigned labels. We denote this subset by $\mathbf{U}$. These discovered samples are combined with the original labeled samples to form an augmented sample set to train a new network and form the master network model $\mathbf{G}_M$. This master network combines the knowledge of the student network $\mathbf{G}_S$ and teacher network $\mathbf{G}_T$, as well as the knowledge of newly discovered samples $\mathbf{U}$. In Snowball, we use the master network to guide the learning of the teacher and student networks and evolve their models over multiple generations with more and more samples being discovered. Our experimental results demonstrate that this tightly coupled model evolution and sample discovery are able to significantly improve the performance of semi-supervised learning, especially on very small sets of labeled training samples.

2.3.2 Master-Teacher-Student Network Model Evolution

In the original Mean-Teacher method [16], a teacher model is constructed by performing an exponential moving average (EMA) of the student network models obtained from past training steps. Each epoch has multiple training steps. This teacher model is then used to guide the training of the student network by enforcing the consistency between the predictions of unlabeled samples by the student and teacher models. This is based on one fundamental semi-supervised learning assumption that neighbor samples in high-density region should have similar outputs [60]. The performance of the teacher and student is limited since their knowledge are based on the original labeled samples. The consistency between the teacher and student is not powerful

enough when the original labeled dataset is very small.

We define two terms: *iteration* and *generation*. In each iteration indexed by k, our method uses the current network model to find the additional subset of confident samples from the remaining unlabeled samples to refine the models. The first iteration starts with the original labeled samples Ω_L. One generation involves a sequence of iterations. In each generation indexed by m, we use the model obtained from the previous generation and come back to re-discover confident samples and refine models over a new sequence of iterations. Specifically, at generation m and iteration k, let the corresponding student, teacher, and master network models be $\mathbf{G}_S^{m,k}$, $\mathbf{G}_T^{m,k}$, and $\mathbf{G}_M^{m,k}$ respectively. Let $\mathbf{U}_k^m$ be the set of newly discovered samples at iteration k and generation m. The label for each sample in $\mathbf{U}_k^m$ is determined by our algorithm which will be discussed in the next section. Then, the current labeled training set is given by

$$\Omega_L^{m,k} = \Omega_L \cup \mathbf{U}_1^m \cup \mathbf{U}_2^m \cup \cdots \cup \mathbf{U}_k^m. \tag{2.1}$$

The current training set $\Omega^{m,k}$ at generation m and iteration k is defined as $\Omega_L^{m,k} \cup \Omega_U^{m,k}$, where $\Omega_U^{m,k}$ is the corresponding unlabeled dataset. We use $\Omega^{m,k}$ to train and update the student, teacher, and master networks. Specifically, we first train the student network $\mathbf{G}_S^{m,k}$. Let $\mathbf{G}_S^{m,k}[t]$ be the corresponding model obtained at training step t. Each training epoch can be multiple training steps [16]. At each step, the teacher model $\mathbf{G}_T^{m,k}[t]$, according to Mean-Teacher method, is constructed and updated based on the following exponential moving average:

$$\mathbf{G}_T^{m,k}[t] = \alpha \cdot \mathbf{G}_T^{m,k}[t-1] + (1-\alpha) \cdot \mathbf{G}_S^{m,k}[t]. \tag{2.2}$$

where α is the exponential moving average decay parameter. It ramps up from 0.99 to 0.999 with the increase of the training steps. The loss function for training the student network is given by

$$\mathbf{J}_S^{m,k} = \lambda_1 \cdot \mathbf{J}_C^{m,k} + \lambda_2 \cdot \mathbf{J}_\theta^{m,k}. \tag{2.3}$$

Here, $\mathbf{J}_C^{m,k}$ represents the classification loss which is the cross-entropy between the student network prediction (softmax output vector) and the associated label over the current labeled training set $\Omega_L^{m,k}$

$$\mathbf{J}_C^{m,k} = \mathbb{E}_{x_l \in \Omega_L^{m,k}} \Phi\{\mathbf{G}_S^{m,k}[t](x_l), \mathbf{L}(x_l)\}, \tag{2.4}$$

where $\mathbf{L}(x_l)$ represents the label of the input and $\Phi\{\cdot,\cdot\}$ represents the cross-entropy. The consistency loss $\mathbf{J}_\theta^{m,k}$ is the mean squared error (MSE) between the student and teacher predictions

$$\mathbf{J}_\theta^{m,k} = \mathbb{E}_{x \in \Omega^{m,k}} \{\|\mathbf{G}_T^{m,k}[t](x) - \mathbf{G}_S^{m,k}[t](x)\|^2\}. \tag{2.5}$$

To construct the master network, we augment the labeled dataset Ω_L into $\Omega_L^{m,k}$ by newly discovered samples $\mathbf{U}_k^m$. The size of $\Omega_L^{m,k}$ is double the size of $\Omega_L^{m,k-1}$ after each iteration. We use the corresponding training set $\Omega^{m,k}$ to update the student network. The teacher network $\mathbf{G}_T^{m,k}[t]$ is refined by the exponential moving average of student network in Eq.(2.2) and the consistency loss in Eq.(2.5) between the teacher and student network. Then, the master network is obtained using the exponential moving average of the teacher networks

$$\mathbf{G}_M^{m,k}[t] = \beta \cdot \mathbf{G}_M^{m,k}[t-1] + (1-\beta) \cdot \mathbf{G}_T^{m,k}[t]. \tag{2.6}$$

The master network is used to guide the training of the teacher and student networks to achieve better transferability on unseen samples. In our experiments, β is set to 0.999. To this end, we augment the consistency loss in Eq. (2.5) by

$$\begin{aligned} \mathbf{J}_\theta^{m,k} = \mathbb{E}_{x \in \Omega^{m,k}} \{ \| \mathbf{G}_T^{m,k}[t](x) - \mathbf{G}_S^{m,k}[t](x) \|^2 \} \\ + \mathbb{E}_{x_l \in \Omega_L^{m,k}} \{ \| \mathbf{G}_M^{m,k}[t](x_l) - \mathbf{G}_S^{m,k}[t](x_l) \|^2 \}, \end{aligned} \tag{2.7}$$

which has two parts consistency: the prediction of sample x from the current training set $\Omega^{m,k}$ between the teacher and student networks, and the prediction of sample x_l from the current labeled training set $\Omega_L^{m,k}$ between the master and student networks. We use MSE to measure the distance between two vectors. During training, the output vectors from student, teacher, and master network are firstly processed by softmax and then used to calculate the consistency loss. The master-teacher-student provides more stable targets for unlabeled samples and improves the results on very small labeled datasets.

2.3.3 Discovering Confident Samples

In this work, we have found out that discovering confident samples from the unlabeled dataset, once coupled with the above master-teacher-student network evolution, can significantly improve the overall semi-supervised learning performance. Our experimental results will demonstrate that these two are tightly coupled and greatly enhance the performance of each other. To discover confident samples and assign

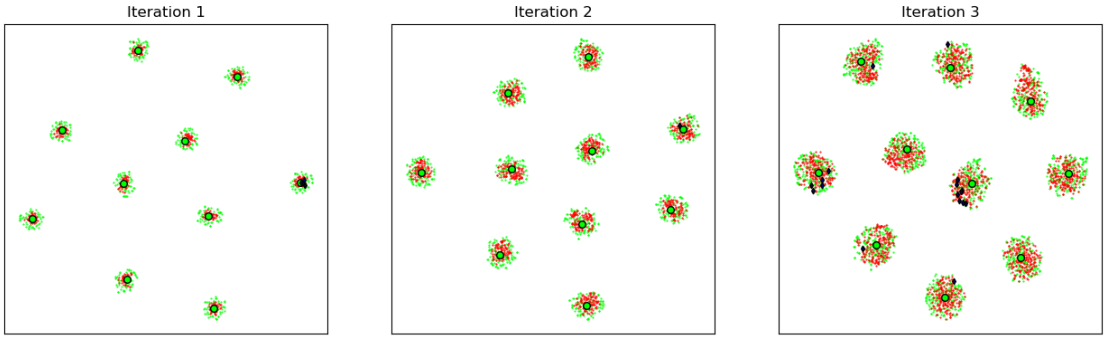


Figure 2.2: Demonstration of the performance of Snowball in each iteration. We select 50 labeled samples from each class as original labeled samples and transfer the high dimension feature to 2-D feature by t-SNE. We use green, red, large green points and black diamond symbols to represent labeled samples, discovered samples, center of labeled samples and error discovered samples respectively.

labels for them, we use the newly trained master network $\mathbf{G}_M^{m,k}$ to extract the feature for each sample x_u in the unlabeled dataset, and denote it by $F(x_u)$. For all samples in the labeled dataset, we also compute their features. We then compute the center for each class

$$\mathbf{C}_n = \frac{1}{T_n} \sum_{x_l \in \Omega_L^{m,k},\ L(x_l)=n} F(x_l), \tag{2.8}$$

where T_n is the total number samples in class n and $L(x_l)$ represents the label of x_l. For the unlabeled image x_u, we find class center C_{n^*} which has the minimum distance to $F(x_u)$, and then assign its label as n^*. Previous works assign soft labels or develop a weighing mechanism for unlabeled samples based on the trained classifier. We assign hard labels for high confident labeled samples directly based on the feature distance between the labeled and unlabeled samples. This will reduce the uncertainty of the model. In our experiments, we find out that samples with smaller distance have higher probability to have correct labels. In iteration (m, k), we select the top $N^{m,k}$ samples

with the smallest feature distance to their centers to form the newly discovered sample set $\mathbf{U}_k^m$. Figure 2.2 visualizes the new sample discovery process on the CIFAR-10 after using t-SNE to reduce the high dimension features into 2D features. We use 500 labeled samples from 10 classes and select the top 500 high confident samples as newly discovered samples in the first iteration. The size of labeled dataset is doubled by introducing newly discovered samples in each iteration. Figure 2.2 shows three iterations with more and more new samples being discovered. The green points are the original labeled samples whose cluster centers are marked with large green points. The newly discovered samples are shown in red. A very small number of discovered samples with wrong predicted labels are marked by a black diamond symbol. We can see that the accuracy of label prediction for newly discovered samples is very high, often in the range of 95-99%.

2.4 Experimental Results

To evaluate the performance of our proposed method, we use three benchmark datasets: SVHN (Street View House Number), CIFAR-10 and CIFAR-100, which have been extensively used for evaluating semi-supervised learning algorithms in the literature [15, 57]. Following the existing evaluation protocol [54, 16], we split the training samples into two parts: labeled and unlabeled samples. To test the performance of our Snowball method on very small sets of labeled samples, we reduce the training set size to from 1000 and 2000 in the literature to 500, 250, and compare with our baseline Mean-Teacher method. In the literature, existing semi-supervised methods often compare their performance on larger sets of labeled samples, for example 1000,

2000, and 4000 images. They did not provide performance results on very small sets. In this case, we choose the Mean-Teacher method for performance comparison, since it has the code publicly available and we have managed to run the code to achieve the same performance as claimed in the original paper. With this, we can generate comparison results of Mean-Teacher on very small sets. We use the source code published on Github by Mean-Teacher [16] and use the same augmentation, training steps, ramp-up and EMA decay rate parameters. We also conduct comparisons on benchmark and very small labeled datasets with the state-of-the-art MixMatch [19] on CIFAR-10 and CIFAR-100 after optimizing our sample discovery method.

The results are averaged over multiple runs with different random seeds. We follow the random sample strategy of Mean-Teacher and ensure that each class has the same number of labeled samples. In both Mean-Teacher and our Snowball methods, two network structures are used: a 13-layer convolutional network (ConvNet-13) and a 26-layer Residual Network [61] with Shake-Shake regularization (Resnet-26) [20]. All results in the following experiments are reported with classification error rates.

2.4.1 Performance Comparison with Existing Methods on the Same Training Sets

(1) Performance Comparison on the SVHN Dataset. The street view house numbers (SVHN) dataset consists of 32×32 pixel RGB images in 10 classes. There are 73257 labeled samples for training and 26032 for testing. It has been used as the benchmark dataset for testing semi-supervised learning and previous state-of-the-art methods have already achieved low error rates which are very close to supervised learning with the full training set (73257 images). All of the labeled and unlabeled

Table 2.1: Error rate percentage of ConvNet-13 and Resnet-26 on the SVHN compared to the state-of-the-art methods.

Methods	250 Labels 73257 Images	73257 Labels 73257 Images
Supervised [16]	27.77 ± 3.18%	2.75 ± 0.10%
Π Model [15]	9.69 ± 0.92%	2.50 ± 0.07%
Temporal Ensembling [15]	12.62 ± 2.91%	2.74 ± 0.06%
SNTG [57]	4.29 ± 0.23%	2.42 ± 0.06%
Mean-Teacher + ConvNet-13 [16]	4.35 ± 0.50%	2.50 ± 0.05%
Mean-Teacher + Resnet-26 [16]	3.53 ± 0.12%	–
Our Method + ConvNet-13	4.07 ± 0.17%	2.50 ± 0.05%
Our Method + Resnet-26	**3.26 ± 0.02%**	–

training datasets are normalized to have zero mean and unit variance. One labeled sample and 99 unlabeled samples are assigned to each mini-batch. Table 2.1 shows our results on the SVHN with 250 labels. We can see that our method outperforms existing state-of-the-art methods, reducing the error rate of the second best (4.29%) further to 3.26%.

(2) Performance Comparison on the CIFAR-10 Dataset. CIFAR-10 is another benchmark dataset for evaluating semi-supervised learning methods. It consists of 32×32 from 10 classes. There are 50000 labeled training samples and 10000 testing samples. Table 2.2 shows the error rates for 1000, 2000, and all 50000 training samples achieved by our method and existing methods. We can see that our method with Resnet-26 achieves the best performance, with an error rate of 7.82%, much lower than the second best 10.08% by Mean-Teacher with the same network configurations. We also provide results of our method with ConvNet-13 and other methods which use similar network configurations. We include the results for the full training set to demonstrate that all methods are having a similar starting point. In the original paper, the Mean-Teacher did not provide result with Resnet. Our method will be the

Table 2.2: Error rate percentage of ConvNet-13 and Resnet-26 on CIFAR-10 compared to the state-of-the-art.

Methods	1000 Labels 50000 Images	2000 Labels 50000 Images	50000 Labels 50000 Images
Supervised [16]	46.43 ± 1.21%	33.94 ± 0.73%	5.82 ± 0.15%
Π Model [15]	27.36 ± 1.20%	18.02 ± 0.60%	6.06 ± 0.11%
Temporal Ensembling [15]	23.31 ± 1.01	15.64 ± 0.39	5.60 ± 0.10
SNTG [57]	18.41 ± 0.52%	13.64 ± 0.32%	5.20 ± 0.14%
Mean-Teacher + ConvNet-13 [16]	21.55 ± 1.48%	15.73 ± 0.31%	5.94 ± 0.05%
Mean-Teacher + Resnet-26 [16]	10.08 ± 0.41%	8.06 ± 0.14%	–
Our Method + ConvNet-13	17.79 ± 0.11%	14.56 ± 0.38%	5.94 ± 0.05%
Our Method + Resnet-26	**7.82 ± 0.08%**	**7.15 ± 0.17%**	–

Table 2.3: Performance comparison of Resnet-26 with Mean-Teacher on the CIFAR-100.

Methods	5000 Labels 50000 Images	10000 Labels 50000 Images
Mean-Teacher + Resnet-26	37.05 ± 0.06%	28.38 ± 0.23%
Our Method + Resnet-26	**34.00 ± 0.10%**	**27.76 ± 0.01%**

same as the Mean-Teacher method when the full training set is used since the master network will never be activated.

(3) Performance Comparison on the CIFAR-100 Dataset. CIFAR-100 is an extension of CIFAR-10, except it has 100 classes and each class has 500 training samples and 100 testing samples. We run further experiments on CIFAR-100 with Resnet and compare with Mean-Teacher. Table 2.3 shows our results with 5000 and 10000 labeled samples. Our method achieves better performance with 27.76% error rate than the Mean-Teacher with a error rate of 28.38% for 10000 labeled samples. For 5000 labeled samples, the error rate is 3% lower than that of Mean-Teacher.

(4) Performance Comparison with MixMatch. In this experiment, we couple our Snowball method with the state-of-the-art MixMatch [19] and conduct

Table 2.4: Error rate percentage of Snowball with MixMatch and ConvNet-13 on CIFAR-10 and CIFAR-100 compared to the state-of-the-art.

	CIFAR-10	CIFAR-100
Methods	1000 Labels	10000 Labels
Label Propagation [62]	16.93 ± 0.70%	35.92 ± 0.47%
DCE [55]	16.53 ± 0.14%	36.75 ± 0.15%
ICT [59]	15.48 ± 0.78%	−
CCN [58]	12.05 ± 0.42%	35.28 ± 0.23%
Deep Co-Train [44]	−	34.63 ± 0.14%
MT + FSWA [56]	15.58 ± 0.12%	33.62 ± 0.54%
Dual-Student [17]	14.17 ± 0.38%	32.77 ± 0.24%
MixMatch [19]	9.08 ± 0.05%	32.95 ± 0.06%
Our Method + MixMatch	**8.86 ± 0.08**%	**31.86 ± 0.11**%

comparison with the state-of-the-art methods. The augmentation approach and loss function of MixMatch are used to train the student network. We observe that the sample discovery with minimum distance contributes less since the augmentation approach in MixMatch blends the labeled and unlabeled samples. Unlabeled samples which have minimum distance to the labeled class center might be too similar to the original labeled samples and they cannot provide extra knowledge. To address this issue, we define a high confidence set which is ten times of the size of newly discovered samples. We find the samples with maximum distance to the class center in this high confidence set can provide useful knowledge for the model learning. To show the effectiveness of our proposed method, we conduct experiments on CIFAR-10 and CIFAR-100 with ConvNet-13 and Resnet-26. Table 2.4 shows the error rates of our Snowball with MixMatch and performance comparison of ConvNet-13 with other state-of-the-art methods. We can see that our proposed method improves the performance and achieves a new state-of-the-art error rate 8.86% on CIFAR-10 with 1000 labeled samples and improves the state-of-the-art result from 32.95% to 31.86% on

CIFAR-100 with 10000 labeled samples. Table 2.5 shows the performance comparison with FSWA [56] and MixMatch [19] on Resnet. Our proposed method reduces the error rate by 4.78% with 4000 labeled samples on CIFAR-10 and 25.10% with 10000 labeled samples on CIFAR-100.

2.4.2 Performance Comparison on Very Small Training Sets

In the following experiments, we demonstrate the performance of our method on very small training sets and provide comparison with the Mean-Teacher method. Table 2.6 shows the results on the CIFAR-10 dataset with the size of training set reduced from 1000 to 500 and 250. We also copy over the results of 1000 and 2000 from Table 2.2 for the convenience of comparison. We can see that, on very small training sets, our method significantly outperforms Mean-Teacher, reducing the error rate from 49.91% to 11.58% with the Resnet-26 network. This 38% performance improvement is very significant. For the Convnet-13 network, the error rate is reduced from 51.79% to 37.65%. Table 2.7 shows the results on the SVHN dataset. We reduce the original training set size from 250 samples to 100 samples. We can see that our method significantly outperforms the Mean-Teacher method, reducing the error rate 15.29% to 6.04%.

In the following, we evaluate the performance of our Snowball with MixMatch on very small labeled dataset. Table 2.8 shows the results on the CIFAR-10 dataset with the size of training set reduced from 1000 to 250 and 100. For fair comparison, we report our best implementation of MixMatch with 250 labeled samples on ConvNet-13, while the reported error rate in the paper [19] is 14.31%. The results show that our proposed method reduces the state-of-the-art error rate from 13.51% to 12.49%

Table 2.5: Error rate percentage of Snowball with MixMatch and Resnet on CIFAR-10 and CIFAR-100 compared to the state-of-the-art.

Methods	CIFAR-10 4000 Labels	CIFAR-100 10000 Labels
MT + FSWA [56]	5.00%	28.80%
MixMatch [19]	4.95 ± 0.08%	25.88 ± 0.30%
Our Method + MixMatch	**4.78 ± 0.07%**	**25.10 ± 0.15%**

Table 2.6: Performance comparison with Mean-Teacher on very small training sets on CIFAR-10.

Methods	250 Labels 50000 Images	500 Labels 50000 Images	1000 Labels 50000 Images	2000 Labels 50000 Images
Mean-Teacher + ConvNet-13	51.79 ± 2.13%	33.02 ± 1.60%	21.55 ± 1.48%	15.73 ± 0.31%
Mean-Teacher + Resnet-26	49.91 ± 9.38%	15.87 ± 0.10%	10.08 ± 0.41%	8.06 ± 0.14%
Our Method + ConvNet-13	37.65 ± 2.49%	22.30 ± 1.48%	17.79 ± 0.11%	14.56 ± 0.38%
Our Method + Resnet-26	**11.58 ± 0.04%**	**9.15 ± 0.82%**	**7.82 ± 0.08%**	**7.15 ± 0.17%**

with 250 labeled samples. We are able to use only 100 labeled samples to achieve 13.20% error rate, which outperforms the MixMatch with 250 labeled samples.

2.4.3 Ablation Studies and Algorithm Analysis

(1) Convergence Behaviors of Snowball. In this experiment, we demonstrate that the proposed Snowball algorithm converges as more and more confident samples are discovered and the master-teacher-student network evolves over iterations and

Table 2.7: Performance comparison with Mean-Teacher on very small training sets on the SVHN.

Methods	100 Labels 73257 Images	250 Labels 73257 Images
Mean-Teacher + ConvNet-13	46.50 ± 10.12%	4.35 ± 0.50%
Mean-Teacher + Resnet-26	15.29 ± 2.63%	3.53 ± 0.02%
Our Method + ConvNet-13	14.20 ± 0.59%	4.07 ± 0.17%
Our Method + Resnet-26	**6.04 ± 0.43%**	**3.26 ± 0.02%**

Table 2.8: Performance comparison between MixMatch and Snowball with MixMatch and ConvNet-13 on CIFAR-10

Methods	100 Labels 50000 Images	250 Labels 50000 Images
MixMatch	–	13.51 ± 2.04%
Our Method + MixMatch	**13.20 ± 0.20%**	**12.49 ± 0.17%**

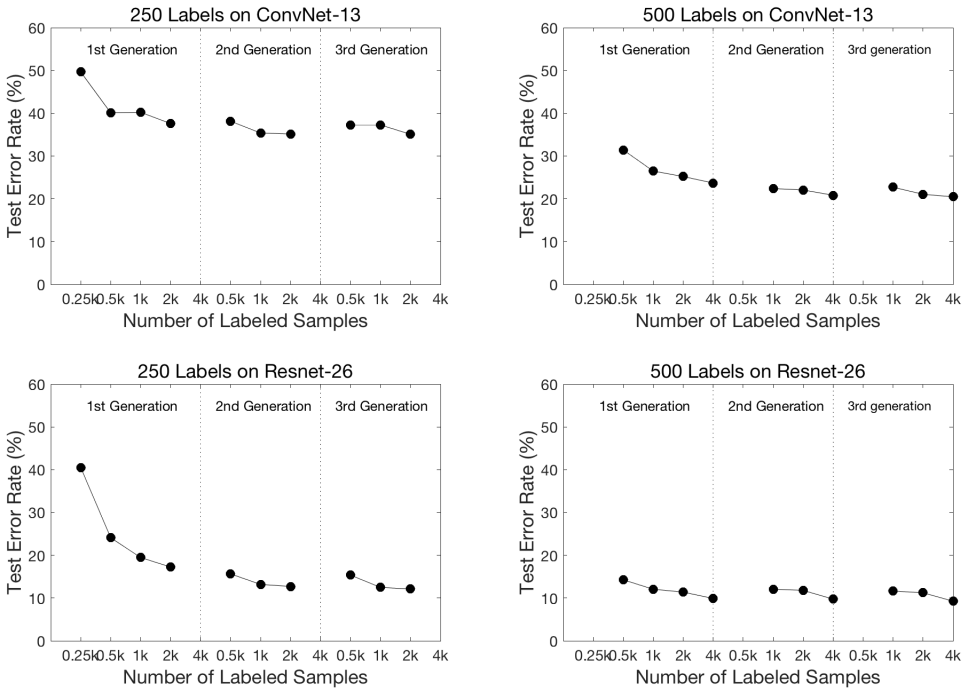


Figure 2.3: Error rate of iterations and generations on CIFAR-10.

generations. In our experiments, We have the termination criterion, which makes the error rate stable between two adjacent iterations, to determine the number of iterations and generations. Typically, the Snowball method uses about 3 generations and each generation has 3-4 iterations. Within each generation, we use the model obtained from the previous generation to re-discover confident samples in each iteration to grow the training samples from 250, to 500, 1000, 2000, and 4000.

Figure 2.3 shows the decreasing error rate of our method on the CIFAR-10 dataset

with two network configurations, ConvNet-13 and Resnet-26, for 250 and 500 labeled samples. The results show that the first generation is the most important part in the Snowball. The error rate is significantly reduced from 40.53% to 17.25% with 250 labeled samples on Resnet-26. Even the error rate of the initial model with 500 labels on Resnet-26 is only 14.27%, the first generation can reduce it to 9.9%. The second generation can clean the error labels from the first generation and reduce the error rate by 2-6%. The third generation is not necessary on larger labeled datasets, since the improvement of it is within 1%. When the number of iterations is over 4, the performance cannot improve too much.

Figure 2.4 shows the detailed sample distribution of 2 selected classes on CIFAR-10 over three sample discovery iterations. Class 1 shows the class without any error labels and class 2 shows the class with error labels. For 2D visualization of these sample images, we applied the principle component analysis (PCA) to reduce the dimension of each image feature extracted from the classification network. The small and large green points represent the labeled samples and class center of labeled samples. The red points show the discovered correct samples. The black diamond symbols show the discovered error samples. The blue points show the discovered samples from previous iteration.

(2) Master-teacher-student model evolution and confident sample discovery. In this work, we recognize that the confident sample discovering and master-teacher-student model evolution are tightly coupled. As discussed in Section 1, discovering confident samples has already been used in self-learning or bootstrap-based semi-supervised learning. These methods suffer from low accuracy because the label prediction error rate in their new sample discovery remains very high. In this

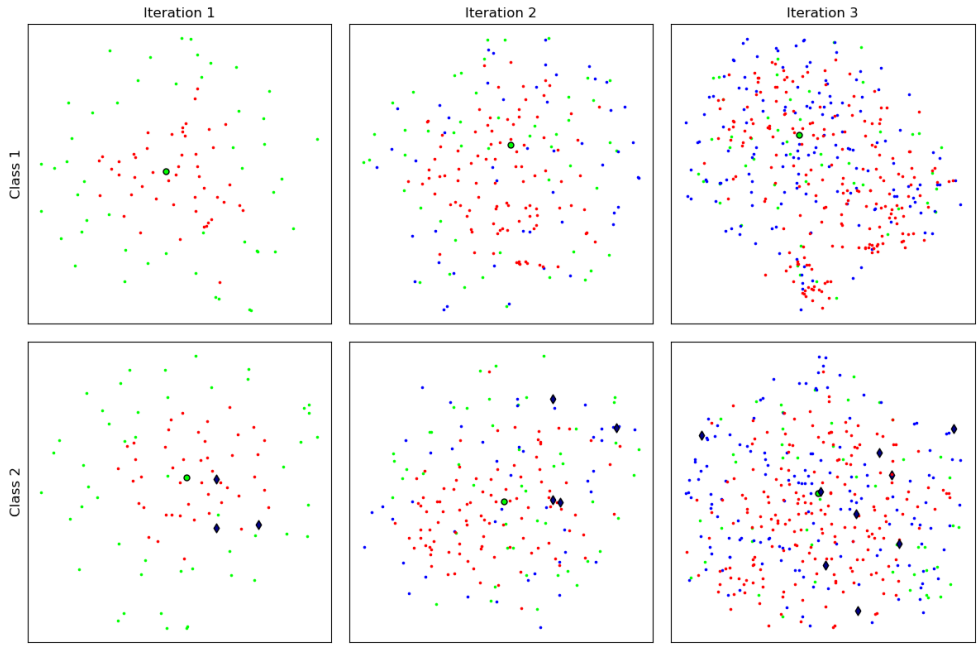


Figure 2.4: The demonstration of sample discovery in each iteration. Each row shows the distribution of clean and discovered samples in 2 different classes. Class 1 shows the class without any error labels. Class 2 shows the class with error labels. The small and large green points show the labeled samples and the center of labeled samples. The red points show the discovered correct samples. The black diamond shows the discovered error samples. The blue points show the discovered samples from previous iteration.

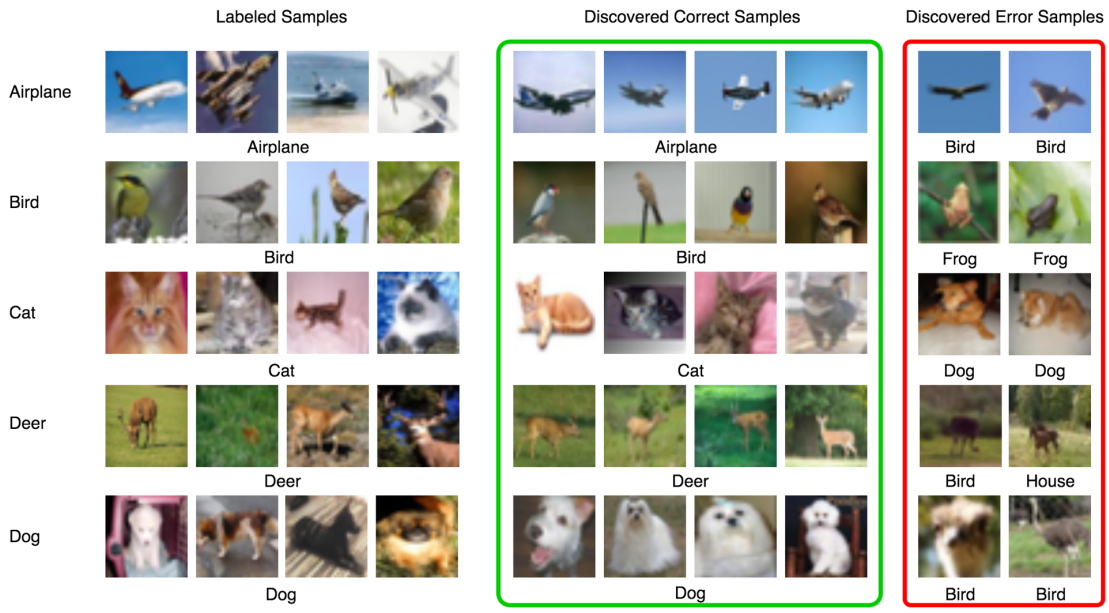


Figure 2.5: Example of labeled samples and discovered samples by sample discovery on CIFAR-10. The discovered samples with error labels are also difficult for human supervision.

case, these new samples would not improve the performance of the original networks. In this work, we find that, once combined with the master-teacher-student model evolution, the new sample discovery can achieve significantly improved performance.

Figure 2.5 shows an example of the new sample discovery process and its errors in label prediction. The first column is the set of labeled samples. The second column is the correct discovered samples. The third column is the discovered error samples. We can see that the discovered error samples are also difficult for human supervision, since these samples have similar color of background or the shape of object to the labeled and discovered correct samples. Figure 2.6(left) shows the comparison of our Snowball method against the self-learning methods with confident sample discovery but without guidance by the master-teacher-student network. We can see that the error rate is dramatically reduced by the Snowball method with master-teacher-

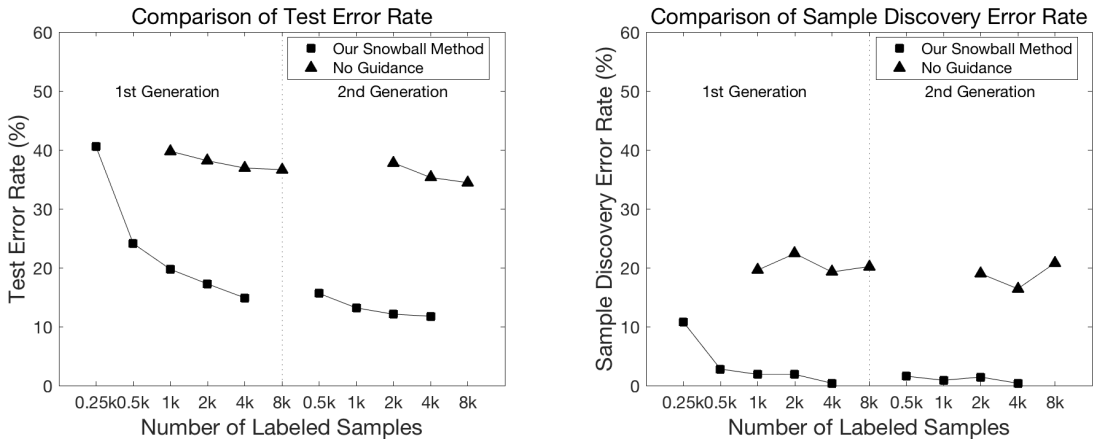


Figure 2.6: Comparison of our Snowball method with self-learning without guidance on the master-teacher-student network. The left figure shows the test error rate on CIFAR-10. The right figure shows the sample discover error rate of each iteration and generation.

student guidance. In Figure 2.6(right), we also show the label prediction error rate of the newly discovered samples by both methods. We can see that the Snowball network is able to predict the labels of new samples much more accurately, which results in significantly improved semi-supervised learning performance.

In our sample discovery method, we use the feature distance to the centroid of labeled samples to discover new samples, while previous self-learning methods use the model prediction to select confident samples. During our experiments, we found that this is much more effective than model prediction by the existing classification network. For example, Figure 2.7 shows the number of samples with incorrect label prediction for each iteration obtained by the feature distance method (blue) and the model prediction method (red) on the CIFAR-10 dataset with 500 labeled training images. We can see that the proposed feature distance method reduces the amount of incorrect label predictions by about a half.

(3) Impact of different selection methods for confident sample discovery.

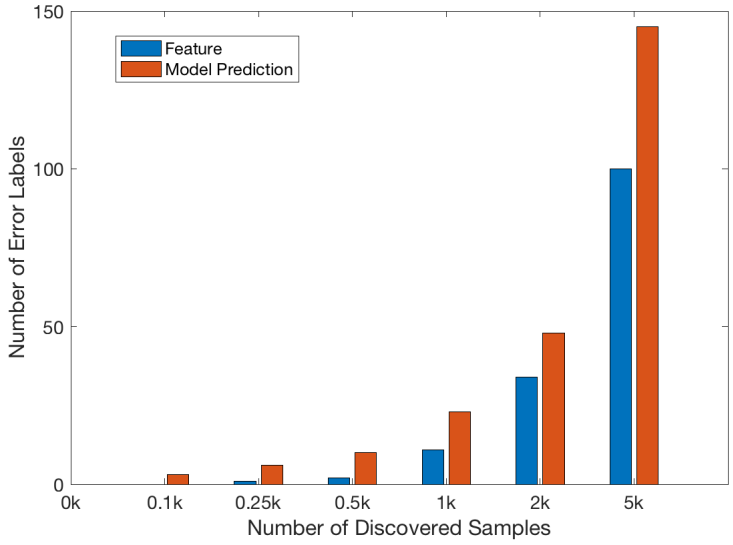


Figure 2.7: Variation of sample discovery error labels.

Table 2.9: Analysis of sample discovery

Methods	With Sample Discovery Label	With Ground-truth Label	Sample Error Rate
500 + 500 (Min)	**11.34%**	10.89%	**0.60%**
500 + 500 (Random)	12.13%	**9.95%**	14.20%
500 + 500 (Max)	16.60%	19.87%	69.20%

Table 2.10: Fusion methods error label percentage of sample discovery.

Fusion Methods	Average Distance	Feature Cascade	Average Sorting Score
Error Labels Percentage	2.9%	2.7%	2.8%

Table 2.11: The Performance of different components from our Snowball method with Resnet-26 on CIFAR-10

Methods	500 Labels 50000 Images	1000 Labels 50000 Images
Mean-Teacher	15.87 ± 0.10%	10.08 ± 0.41%
Mean-Teacher + Sample Discovery	11.30 ± 0.58%	9.04 ± 0.25%
Master-Teacher-Student + Sample Discovery	**9.15 ± 0.82%**	**7.82 ± 0.08%**

In our Snowball new confident sample discovery method, we find the center of labeled samples which has the minimum distance to the current sample and then assign the corresponding label to this unlabeled sample. We choose the top N samples with the smallest distance as the newly discovered samples for the next iteration. On the other hand, we recognize that samples with minimum distance to existing labeled samples might be too similar to existing training samples and the contribution to the model learning and transferability will be degraded. Other possible choices include using the maximum distance criteria, or we randomly select the top N samples. Which method is the most effective? In the following experiment, we set the number of labeled samples to be 500 on the CIFAR-10 dataset. With these 500 labeled samples, the error rate of the trained classifier is 14.44%. The next step is to discover 500 new samples using the above three different methods. We then use these newly discovered samples along with the original labeled samples to refine the classifier. Table 2.9 shows the error rate results on the CIFAR-10 dataset with 500 original labeled training samples and 500 newly discovered samples using different selection methods. The second column in Table 2.9 shows the corresponding model error rates. We can see that the minimum feature distance method achieves the best performance. The third column in Table 2.9 shows the model error rates with 500 new samples selected by three different methods, but using the ground-truth labels. The main difference lies in the error rates in label prediction by these three methods. From the fourth column in Table 2.9, we can see that the minimum feature distance method has an error rate of only 0.60%, while the random method and maximum feature distance method have much higher error rates. If we do not consider the errors in label prediction, for example, if we assume that newly discovered samples all have

correct labels, then the random selection method has the best performance since its samples have the largest diversity. But, in practice, we do not have this ground-truth label. In this case, our minimum distance method achieves the best performance since its percentage of wrong labels in newly discovered samples is much smaller than the other two methods. This is the reason why we choose the minimum distance method in our Snowball method.

(4) Different feature fusion methods for confident sample discovery. In our current method, when we identify new confident samples for automated labeling, we use the student network to extract its feature and evaluate its feature distance to existing labeled samples. In the following experiment, we explore additional options for the feature distance. For example, we can use three network models of the past iterations to extract three separate features. We then fuse these features together to form a joint feature distance metric. The following three fused feature distance metrics are considered. **(1) Average distance** - We use each of these three features to compute the distance and then use the average of them as the distance metric for this unlabeled sample. **(2) Feature cascade** - These three features are cascaded together into one combined feature vector for this unlabeled sample. We then use this cascaded feature to measure the distance to assign labels. **(3) Average sorting score** - We use each of these three features to compute the minimum distance, then sort the samples according to their distance from the smallest to the largest. For each of these three features, we have three separate sorting scores (sorting indices), we then compute their average sorting scores and use this as the distance metric. We use the ratio of the number of error discovered labels to the number of discovered labels to measure the performance. Table 2.10 shows the error label percentage for

these three fusion methods on the CIFAR-10 dataset with 1000 training samples. We can see that the feature cascade method has the best performance. But, the difference between these three are relatively small.

(5) Performance of different algorithm components. Our Snowball method has two major components: confident sample discovery and master-teacher-student network. The confident sample discovery discovers confident samples from the unlabeled dataset and assigns labels to these discovered samples. The master network combines the knowledge of the student network and teacher network, as well as the knowledge of newly discovered samples. These two tightly coupled components are able to significantly improve the performance of semi-supervised learning. In this ablation study, we conduct experiments with 500 and 1000 labeled samples on CIFAR-10 to identify the contribution of each algorithm component. Table 2.11 summarizes the performance results with Resnet-26 using three different configurations: (1) Mean-Teacher, (2) Mean-Teacher with sample discovery, and (3) master-teacher-student with sample discovery. We can see that these two component are both very important for the overall performance.

2.5 Conclusion

In this work, we have successfully developed a joint sample discovery and iterative model evolution method for semi-supervised learning from a very small labeled training set. We have established a master-teacher-student model framework to provide multi-layer guidance during the model evolution process with multiple iterations and generations. Both the master and teacher models are used to guide the training of the

student network by enforcing the consistency between the predictions of unlabeled samples between them and evolve all models when more and more samples are discovered. Our extensive experiments demonstrate that the discovering confident samples from the unlabeled dataset, once coupled with the above master-teacher-student network evolution, can significantly improve the overall semi-supervised learning performance.

Chapter 3

Learned Model Composition with Critical Sample Look-Ahead for Semi-Supervised Learning

3.1 Introduction

In this work, we propose to push the performance limit of semi-supervised learning by developing an efficient learning method on *small* sets of labeled samples. For example, on the same CIFAR-10 dataset, existing methods can achieve successful network training with only 250 labeled samples, approaching the performance of networks being trained on the fully labeled dataset of 50000 samples in a fully supervised manner. Can we successfully train the network with an even smaller set of samples, for example, 100 samples? How about 80 samples? Training efficient deep neural networks on much smaller sets of labeled samples represents an important challenging task in machine learning and computer vision. In the semi-supervised learning process, the

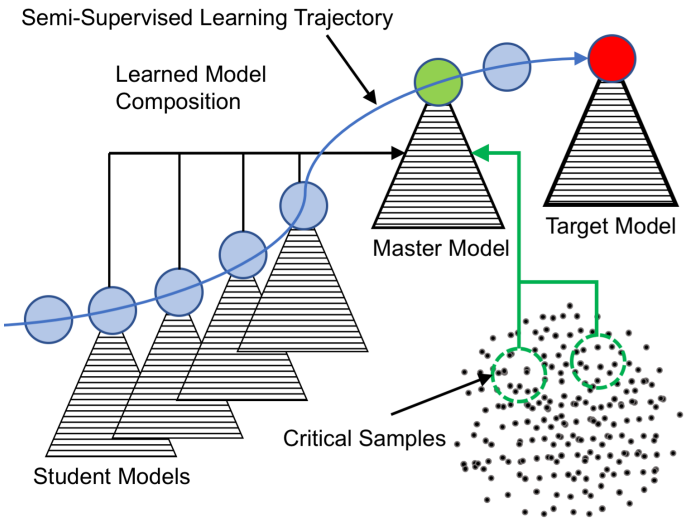


Figure 3.1: Our proposed idea of learned composition of the master model to guide the semi-supervised learning process towards the target model. The master model is composed of student models from past iterations. The target model is trained on the fully labeled dataset. The dashed green circles represent different class of critical samples.

initial network suffers from low accuracy with a limited number of labeled samples. It will lead to an unstable subsequent training. Existing semi-supervised learning methods, including the most recent Mean-Teacher [16], Dual Students [17], and Mix-Match [19] algorithms suffer from significant performance degradation on smaller sets of labeled samples.

To address this issue, we propose to explore a new approach called *learned model composition with critical sample look-ahead* (LMCS). The basic idea is illustrated in Figure 3.1. In semi-supervised learning, the network model, referred to as the student model in this work, is initially trained on the set of labeled samples. Since this set is small, the student model often suffers from low accuracy. The central question of semi-supervised learning is how to guide or enhance the learning process of the

student model so that it can evolve towards the target model which is trained on the fully labeled set. The guidance method should have the capability to identify the correct direction of model evolution so that it can pull the student model towards the target model.

In this work, we introduce a master model to provide this guidance. During this model evolution process, there are two major types of resources to be exploited: the unlabeled samples and the previous versions of the student models obtained from the past training epochs. To achieve the above mentioned capability for the master model, we propose to explore **two major ideas**. First, the master model is composed from the past versions of student models using a learned network so that it can capture the evolution trend of student models. Second, we develop a method called *confined maximum entropy search* to discover new critical samples near the model decision boundary. According to our experiments, these samples are able to provide critical information to refine the master models. We allow the master model to have a look-ahead access to these samples before the student model so that it can identify the direction of evolution for the student model. This master network is used to guide the semi-supervised learning process from a small set of labeled samples to achieve the target performance by enforcing the prediction consistency between student and master models on unlabeled samples. Our extensive experiments demonstrate that the proposed LMCS network outperforms the state-of-the-art semi-supervised learning methods, especially on small labeled training sets.

Our main contributions can be summarized as follows: (1) we introduce a new learned model composition structure, offers a powerful master network that can be composed from student models of past steps through a network learning process. (2)

We develop the confined maximum entropy search to discover new critical samples near the model decision boundary to refine the master network. (3) Our experimental results and ablation studies show that the proposed LMCS method outperforms the state-of-the-art semi-supervised learning methods, especially on small sets of labeled samples. For example, on the CIFAR-10 dataset, with a small set of 80 labeled samples, our proposed method outperforms Google's MixMatch method by 10%.

The rest of the chapter is organized as follows. In Section 3.2, we provide a review of recent work on semi-supervised learning. The proposed LMCS method is presented in Section 3.3. Experimental results and ablation studies are presented in Section 3.4. Section 3.5 concludes the chapter.

3.2 Related Work

Recently, a number of methods have been developed and achieved remarkable performance. These methods which can be summarized in the following categories.

(1) Self-training methods. As one of the earlier methods for semi-supervised learning, the self-training approach [37, 38, 39], also called bootstrapping or self-teaching, first trains the initial classifier with labeled samples and then uses the pre-trained classifier to classify unlabeled samples, selects the most confident unlabeled samples, predicts their labels, and uses them for the next iteration of training [40]. The procedure is repeated until it reaches the stop criteria. It is also commonly used in many semi-supervised learning tasks, such as image classification [63], natural language processing [36, 64] and object detection [38]. Traditional self-training is designed by automatically labeling unlabeled samples and increasing the size of labeled

datasets. This often introduces incorrectly labelled samples which might degrade the model performance.

(2) Graph-based methods. Graph-based methods [3, 4, 5, 6, 7] aim to measure the similarity between labeled and unlabeled samples by constructing a graph. Knowledge learned from the labeled samples is propagated along the graph to predict the labels of unlabeled samples [45, 4]. This is based on the assumption that similar samples share the similar predicted labels [40]. Qiu *et al.* [7] proposed a method to exploit the adjacency matrix between a small number of labeled samples and all data samples to construct the graph. They use a regression residue term and a manifold smoothness term jointly on the hard linear constraint problem. Zhang *et al.* [65] proposed a semi-supervised deep hashing method on large scale image retrieval task. It exploits the structures of unlabeled samples and builds an online graph construction method based on the mini-batch during the training.

(3) Label propagation. The label propagation method [46] compares unlabeled samples with labeled samples by selecting a suitable predefined distance metric. Previous label propagation trains and fixes the network in advance. Recently, Wu *et al.* [49] proposed a cross-task network which jointly processes two tasks: classification and clustering. Based on the model predictions, a large number of pairwise constraints can be generated from unlabeled samples, and are fed to the clustering task. They used pairwise constraints to encode weak supervision information. Unlabeled samples are weighted according to the distances to the clusters discovered, and an improved model is then trained on the classification task associated with a weighted softmax loss. Deep co-space [66] learns a feature transformation matrix and discovers reliable unlabeled samples by measuring the category variations of the feature trans-

formations from their two neighbors. Iscen *et al.* [62] perform label propagation while training the network. The model is firstly trained with labeled and unlabeled samples, then the model from previous iteration is used to construct the nearest neighbor graph, which is used to infer pseudo-labels of unlabeled samples for the next iteration.

(4) GANs-based methods. Another group of methods for semi-supervised learning is based on the generative adversarial networks (GANs) [8, 9, 10, 11]. Szegedy *et al.* [50] used adversarial training to discover the most sensitive perturbation for the labels of input samples. Kingma *et al.* [8] proposed an auto-encoder generative model for semi-supervised learning. Pseudo ensemble agreement reported in [9] trains a neural network that the output of each layer does not change much by different perturbed inputs. It demonstrated the effectiveness of random perturbations in semi-supervised learning. Miyato *et al.* [34] introduced a new measure for local smoothness of the conditional label distribution. They proposed the method of virtual adversarial training (VAT), which can discover the maximum adversarial perturbation of a input sample based on the difference between the input and output.

(5) Entropy minimization-based methods. Entropy minimization assumes that the decision boundary of the learned network should lie in the low-density regions of samples [51, 13]. This allows us to define a loss function during network training which is able to utilize unlabeled samples. Triguero *et al.* [52] couple entropy minimization with self-training to propagate the labels to the unlabeled samples. Pseudo-labeling [14] trains the classifier with the labeled samples and gradually introduces the weighted pseudo-labels with larger class probabilities for unlabeled samples. Unlabeled samples and corresponding pseudo-labels are used in the standard loss function. Hoffer *et al.* [53] follow a similar principle and uses deep metric

embedding to measure the distance between the labeled and unlabeled samples.

(6) Consistency regularization-based methods. Consistency regularization ensures consistent model prediction over different sample perturbations. It often applies a consistency constraint on the teacher-student network to learn knowledge from perturbed data. For example, **Π-model** [15] constructs a implicit teacher model which shares parameters with the student model. It forwards a sample with different perturbation twice and introduces new loss functions for unlabeled samples which penalize inconsistent predictions. **Temporal ensembling** [15] improves upon the Π-model method by proposing an exponential moving average of predictions obtained from previous epochs, rather than evaluating the input twice. The **Mean-Teacher** algorithm [16] takes the concept of temporal emsembing and uses an exponential moving average of the parameter of student models obtained from previous training steps. The student and teacher models improve each other in an iterative manner. Athiwaratkun *et al.* [56] combine consistency regularization with stochastic weight averaging (**SWA**) to obtain a more powerful ensemble teacher. The Smooth Neighbors on Teacher Graph (**SNTG**) method [57] constructs a graph-based prediction of the teacher model to make the learned features more discriminative by exploring intra-class similarity and inter-class dissimilarity. **Dual student** [17] replaces the teacher with another independent student model. It has two independent students with different initial states and optimization path.

The most recent semi-supervised learning method, **MixMatch** from Google [19] combines the ideas of consistency regularization [15] and entropy minimization [13]. It uses the model prediction to produce a guessed label for each unlabeled sample based on principle of entropy minimization. It averages the prediction of perturbed

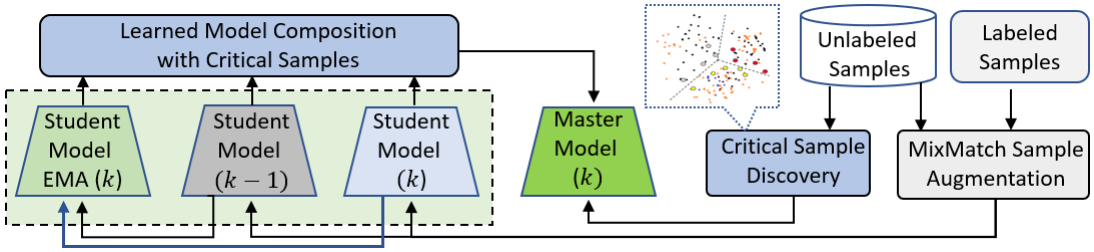


Figure 3.2: Overview of the proposed LMCS method. The labeled and unlabeled samples with the MixMatch sample augmentation are used to train student models. Our master model consists of the student model, exponential moving average (EMA) student model and student model from previous training step with the same architecture. The critical sample discovery uses the confined maximum entropy search and assigns labels to unlabeled critical samples which have large ambiguity.

unlabeled samples and uses the MixUp approach [18] as a regularizer to blend labeled samples and unlabeled samples randomly. This method currently achieves the state-of-the-art performance in semi-supervised learning [19].

3.3 The Proposed LMCS Method

In this section, we present the proposed LMCS method for semi-supervised learning.

3.3.1 Method Overview

Figure 3.2 provides an overview of the proposed LMCS method. In semi-supervised learning, we have access to a small set of labeled training samples, denoted by Ω_L and a large set of unlabeled samples, denoted by Ω_U. The network is trained from scratch. We pair the unlabeled samples with labeled samples to generate soft samples with soft labels using the MixMatch sample augmentation scheme [19]. Specifically, we randomly sample an equal number of labeled and unlabeled samples as the mini-

batch samples for each iteration. let $(x, \mathbf{p})$ be the image sample x and its label probability vector $\mathbf{p}$ from Ω_L. For each image sample x' from the mini-batch in the unlabeled set Ω_U, it uses the current classifier to predict its label probability vector $\mathbf{p}'$. The augmented sample $\tilde{x}$ and its corresponding label probability $\tilde{\mathbf{p}}$ are generated by

$$\lambda = max(\lambda', 1-\lambda'), \tag{3.1}$$

$$\tilde{x} = \lambda x + (1-\lambda)x', \tag{3.2}$$

$$\tilde{\mathbf{p}} = \lambda \mathbf{p} + (1-\lambda)\mathbf{p}', \tag{3.3}$$

where λ' is a random number following a Beta distribution [19]. The augmented mini-batch is then used to train the student model $\mathbf{\Theta}_s^i$ in the next iteration, where i represents the training iteration index. The student model EMA $\bar{\mathbf{\Theta}}_s^i$ which is defined by the following equation:

$$\bar{\mathbf{\Theta}}_s^i = \alpha \cdot \bar{\mathbf{\Theta}}_s^{i-1} + (1-\alpha) \cdot \mathbf{\Theta}_s^i, \tag{3.4}$$

where α is the exponential moving average parameter. In our experiments, we set α to 0.999. It represents the exponential moving average (EMA) of student models obtained from the past training iterations. We define a term: step. In the training process, each step indexed by k can be a number of epochs and each epoch can be a number of iterations. Using the student models of the current step $\mathbf{\Theta}_s^k$ and the previous step $\mathbf{\Theta}_s^{k-1}$, as well as the student model EMA $\bar{\mathbf{\Theta}}_s^k$, we construct the master model $\mathbf{\Theta}_m^k$ using a learned model composition approach. The central issue to be addressed here is: given a set of trained student models of the same network, how

could we construct a new master model which outperforms each student model and is able to effectively guide the learning of the network? The Mean-Teacher method used exponential moving average of student models $\mathbf{\Theta}_s^i$ to construct the ensemble model.

In this work, we formulate this problem into a learning process using a learned model composition (LMC) network. Besides having access to all past student models, the LMC network also discovers critical samples from the unlabeled samples. These critical samples should satisfy two important requirements: (1) the predicted labels for these discovered samples should have high accuracy. Otherwise, it will degrade the learning efficiency. (2) These new samples should be helpful in improving the discriminative power of the master model. In this chapter, we introduce a new approach called *confined maximum entropy search* for critical sample discovery. It first builds a confined unlabeled set by the feature distance. Each unlabeled sample in the confined set is assigned a soft label based on the ranked similarity between it and labeled samples. The critical samples are selected and assigned hard labels by a maximum entropy criteria.

Our extensive experimental results demonstrate that the proposed LMCS method is able to construct powerful and reliable master models to guide the learning process of the student network and significantly improve the semi-supervised learning performance. Coupled with critical sample discovery, the network is able to efficiently learn from a small set of labeled samples in semi-supervised learning. In the following sections, we will explain the learned model composition and critical sample discovery using confined maximum entropy in more details.

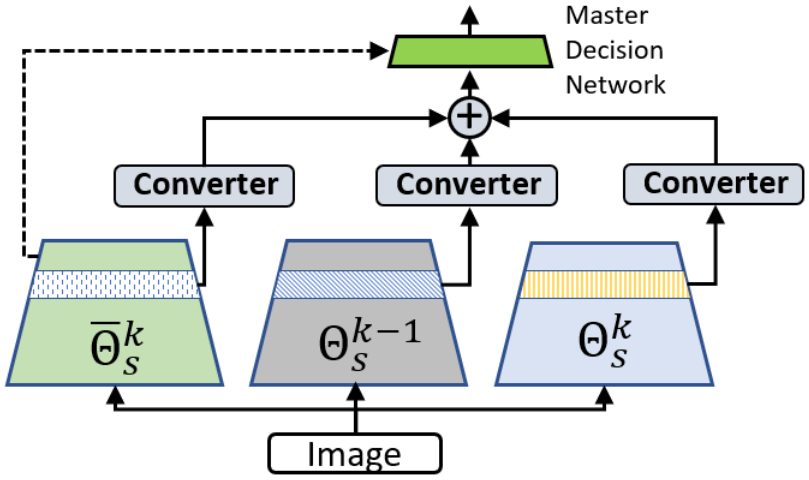


Figure 3.3: Learned model composition for constructing master models. The converter serves as a weighting mechanism for feature map from different student network. The input of the master decision network is the fused feature map.

3.3.2 Learned Model Composition

In semi-supervised learning, the network is trained over a sequence of steps. For example, in our experiments, we choose one step to be 10 epochs. At step k, the corresponding model is referred to as the student model $\mathbf{\Theta}_s^k$. The task of learned model composition (LMC) is to construct a more powerful master model from the existing student models $\mathbf{\Theta}_s^n$, $n = 1, 2, \cdots, k$. Note that k is variable and growing with the training steps. Instead of using all k student models, in LMC, we choose to use the following three models, the current student model $\mathbf{\Theta}_s^k$, the previous student model $\mathbf{\Theta}_s^{k-1}$, and the current EMA student model $\bar{\mathbf{\Theta}}_s^k$ which is the EMA student model $\bar{\mathbf{\Theta}}_s^i$ from current iteration. As observed in the Mean-Teacher method [16], the EMA student models provide an efficient integrated representation of the past student models. It accumulates the information from each iteration instead of each epoch. Parameters are averaged to output better intermediate representations of each layer.

In this way, the EMA student models improve the prediction accuracy.

Figure 3.3 shows the proposed LMC network to construct the master model from these three student models. It should be noted that they share the same network layer structure and configuration. The only difference is their weight parameters. We choose one layer as the composition layer, which splits each network into two network modules: feature extraction and decision networks which correspond to network layers before and after the composition layers. During composition, the feature maps at the composition layer from each student network are firstly processed by a converter which is a fully connected layer and then added together to produce the fused feature map. The converter serves as a weighting mechanism for each feature map. The master decision network maps the fused feature map into the final classification output. The three converter networks and the master decision network are learned with the whole training dataset, including both labeled and unlabeled samples, using the master loss function $\mathbf{J}_M$ to be defined in the following.

The loss function for training the master model composition network consists of two parts: consistency loss $\mathbf{J}_C$ and discriminative power loss $\mathbf{J}_D$. The consistency loss $\mathbf{J}_C$ is the mean squared error (MSE) between the student model EMA and master model predictions on unlabeled samples

$$\mathbf{J}_C = \mathbb{E}_{x' \in \Omega_U} \Phi[\mathbf{\Theta}_m^k(x'), \bar{\mathbf{\Theta}}_s^k(x')], \tag{3.5}$$

where $\Phi[\cdot, \cdot]$ represents the mean squared error between the predicted label probabilities by these two networks. When training the master model composition network, we also expect that the constructed master model has improved discriminative power on the unlabeled samples. According to the minimum entropy principle, if a network

classifier has a higher discriminative power, then its decision boundary should lie in the low-density regions of samples. In other words, if we use this network to encode the input image into a feature, in the corresponding feature space, the test samples should exhibit better clustering behaviors. Let $\mathbf{f}(x)$ be the feature vector extracted by the master model $\mathbf{\Theta}_m^k$ of an unlabeled sample x in Ω_U. We cluster these features into N clusters. Let $\mathbf{C}_0(x)$ and $\mathbf{C}_1(x)$ be the cluster centers with the smallest and the second smallest feature distance to x. If $\mathbf{C}_0(x)$ and $\mathbf{C}_1(x)$ are very close to each other, then the discriminative power of $\mathbf{\Theta}_m^k$ on sample x is low. Therefore, we can use the following average distance as the discriminative power loss

$$\mathbf{J}_D = \sum_{x \in \Omega_U} \{||\mathbf{C}_0(x) - \mathbf{f}(x)||_2 - ||\mathbf{C}_1(x) - \mathbf{f}(x)||_2\}^2. \tag{3.6}$$

We define the loss function of the master network in Eq. (3.7) by

$$\mathbf{J}_M = \mathbf{J}_C + \eta \cdot \mathbf{J}_D, \tag{3.7}$$

where $\eta > 0$ is a hyperparameter that balances the contributions of the consistency loss and the discriminative power loss. We select the loss function from MixMatch [19] as the student loss $\mathbf{J}_S$ and use the same training procedure, hyperparameters as MixMatch [19] to train the student network.

3.3.3 Critical Sample Discovery Using Confined Maximum Entropy Search

Another important feature of our LMCS method is to discover samples from the unlabeled set Ω_U, assign labels to them, and include them into the training set along

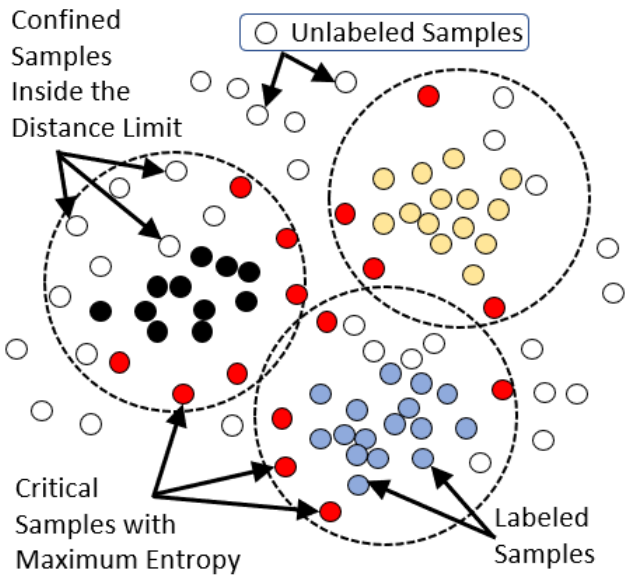


Figure 3.4: Overview of the critical sample discovery. The black, yellow and blue data points represent the labeled samples from different classes. The white and red data points represent the unlabeled and discovered critical samples separately. The dashed circle represents the confined limit.

with the labeled samples to train the master model. We observe that samples from the unlabeled set with large ambiguity or being close to the decision boundary, if assigned with correct labels, will contribute more to the model learning performance. But, the pseudo-label accuracy of these samples is relatively low. The unlabeled samples which are close to the class center of labeled samples have high pseudo-label accuracy, but these samples might be too similar to the initial labeled samples and their contribution to the model training is very limited.

To address this issue and successfully identify the critical samples, we propose the following *confined maximum entropy search* approach. Figure 3.4 shows the overview of this approach. First, we find confined unlabeled samples which are not far away from the samples with known labels so that we can assign labels to them with high confidence. Specifically, we use the current student model $\mathbf{\Theta}_s^k$ to extract feature $\mathbf{g}(x')$

for each sample x' in the unlabeled set Ω_U and feature $\mathbf{g}(x)$ for each labeled sample x in the labeled set Ω_L. For the labeled set, we compute the center for each class

$$\mathbf{C}_n = \frac{1}{T_n} \sum_{\{x \in \Omega_L,\ L(x)=n\}} \mathbf{g}(x), \tag{3.8}$$

where T_n is the total number samples in class n and $L(x)$ represents the label of x. For each unlabeled sample x', we find its minimum distance $d(x')$ to the closest class center. Let

$$\Omega_C = \{x' | d(x') \leq \Delta\}, \tag{3.9}$$

be the confined set of unlabeled samples. Δ is a distance threshold.

Second, we assign a soft label for each unlabeled sample $x' \in \Omega_C$ based on the ranked similarity between it and labeled samples. Figure 3.5 shows the example of critical sample discovery on the confined set. Specifically, let $d(x', x)$ be the feature distance between x' and each labeled sample x in Ω_L. We rank the labeled samples based on distance $d(x', x)$ from the smallest to the highest. Let $\mathbf{R}(x)$ be the rank order index of each labeled sample. We then define the following rank similarity weight

$$\mathbf{W}(x) = \frac{\beta}{\beta + \mathbf{R}(x)/|\Omega_L|}, \tag{3.10}$$

where β is a weight control parameter and $|\Omega_L|$ represents the number of labeled samples in the labeled set. Let $\mathbf{p}(x)$ be the label probability vector of x. Since x is from the labeled set, $\mathbf{p}(x)$ is a binary vector with only a single 1 at the location corresponding to its class. The soft label assigned to the confined sample x' is given

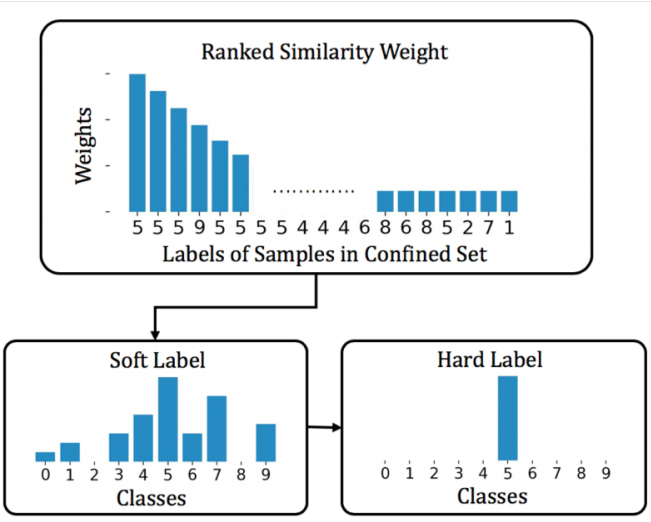


Figure 3.5: Example of the critical sample discovery in confined set. Each unlabeled sample in confined set has a ranked similarity weight. The soft label is voted by the ranked similarity weight. The hard label is assigned to the critical sample.

by

$$\mathbf{p}(x') = \left[\sum_{x \in \Omega_L} \mathbf{W}(x) \cdot \mathbf{p}(x)\right] / \sum_{x \in \Omega_L} \mathbf{W}(x). \tag{3.11}$$

In the third step, we choose those confined samples near the decision boundary based on the maximum entropy criteria. Specifically, if the classifier has higher uncertainty about the sample, the predicted label probability vector will have a large entropy

$$\mathcal{H}(x') = \sum_{l=1}^{N} p'_l \cdot \log_2 \frac{1}{p'_l}, \tag{3.12}$$

with $\mathbf{p}(x') = [p'_1, p'_2, \cdots, p'_N]$. We rank all confined samples in $|\Omega_C|$ based on their predicted label entropy and select the top N^k samples as the critical samples at training step k.

3.3.4 Multi-Generation Master-Student Network Learning

Our student network training, master network model composition, and critical sample discovery operate in a multi step-generation framework to achieve efficient semi-supervised learning, especially from a small set of labeled samples.

We define two terms: *step* and *generation*. The first step starts with the original set of labeled samples Ω_L. Each generation indexed by g involves a number of steps such that the size of the training set keeps doubled until the labeled training set is R_L times larger than the original labeled set. In our experiment, we set R_L to be 10, depending on the convergence speed of the learning process. In each generation indexed by g, our method uses the model obtained from the previous generation to find critical samples from the remaining unlabeled samples.

Specifically, at generation g and step k, let the corresponding student and master network models be $\mathbf{\Theta}_s^{g,k}$ and $\mathbf{\Theta}_m^{g,k}$. Let $\mathbf{U}_g$ be the set of newly discovered critical samples at generation g. The label for each sample in $\mathbf{U}_g$ is determined by Eq. (3.8) and Eq. (3.11). Let

$$\Omega_G = \Omega_L \cup \mathbf{U}_1 \cup \mathbf{U}_2 \cup \cdots \cup \mathbf{U}_g \tag{3.13}$$

be the training set at generation g. We use Ω_{G-1} to train or update the student model $\mathbf{\Theta}_s^{g,k}$ and Ω_G to train the model composition network to obtain the master model $\mathbf{\Theta}_m^{g,k}$. In other words, the master model always has access to more samples so that it can provide more effective guidance for the student model training. Typically, in our experiments, the proposed LMCS semi-supervised learning converges fast within about 3 generations. Figure 3.7 visualizes the sample distribution on each generation. We use the t-SNE approach to reduce the high feature dimension into 2D. It shows the testing samples from the CIFAR-10 dataset after three generations. We can

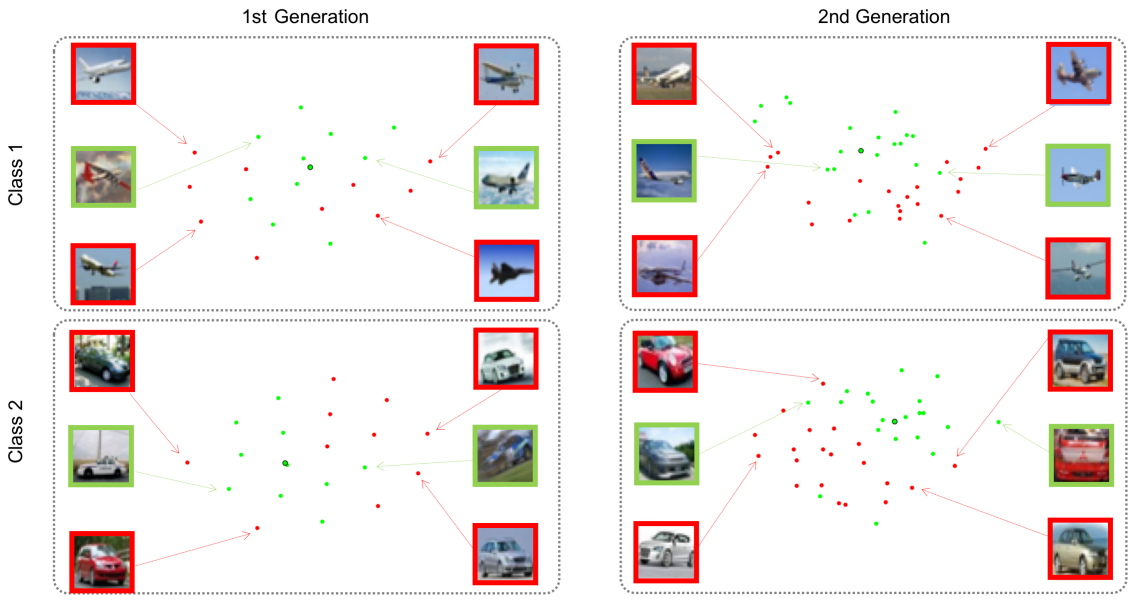


Figure 3.6: The demonstration of critical sample discovery in two generations. We use 10 initial labeled samples from each class as original labeled samples and transfer the high dimension feature to 2-D feature by t-SNE. Each row shows the distribution of clean and discovered samples in two different classes. The green points and green points with black circle show the labeled samples and the center of all labeled samples. The red points show the newly discovered critical samples. Examples of labeled samples and critical samples are highlighted with green and red, respectively.

see that the model performs better and samples are better separated when more and more critical samples are discovered. Figure 3.6 shows the detail of critical sample discovery in the first two generations. We use 10 initial labeled samples from each class as original labeled samples and transfer the high dimension feature to 2-D feature by t-SNE. Each row shows the distribution of clean and discovered samples in two different classes. The green points and green points with black circle show the labeled samples and the center of all labeled samples. The red points show the newly discovered critical samples. Examples of labeled samples and critical samples are highlighted with green and red, respectively.

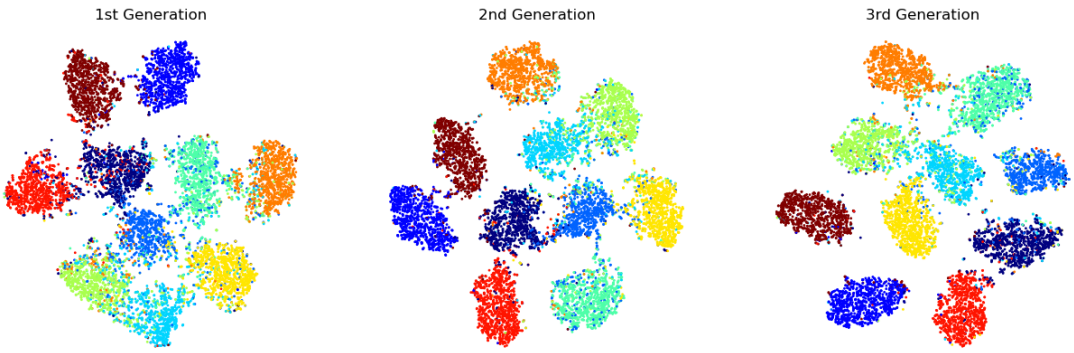


Figure 3.7: Demonstration of sample distribution on three generations. We use the t-SNE to transfer the high-dimension feature to 2-D feature on the CIFAR-10 testing dataset

3.4 Experimental Results

In this section, we present our experimental results on benchmark datasets, performance comparison with state-of-the-art methods, and ablation studies.

3.4.1 Benchmark Datasets and Networks

Following existing work in the literature [17, 19], we implement and evaluate our proposed LMCS method on three standard benchmark datasets: CIFAR-10 [67], CIFAR-100 [67], and SVHN (Street View House Number) [68] **(1) CIFAR-10.** CIFAR-10 consists of 32×32 pixel RGB images in 10 categories. There are 50000 labeled training samples and 10000 testing samples. Fully supervised learning with all 50000 labeled training set achieves 4.17% error rate on ResNet-28 [19]. **(2) CIFAR-100.** As an extension of CIFAR-10, CIFAR-100 is a more complex dataset with 100 classes and each class has 500 training samples and 100 testing samples. The fully supervised learning error rate of 50000 labeled samples with ConvNet-13 on this 100 classes

Table 3.1: Error rate percentage of ResNet-28 on CIFAR-10 compared to the state-of-the-art.

Methods	250 Labels	500 Labels
Π Model [15]	53.02 ± 2.05%	41.82 ± 1.52%
PseudoLabel [14]	49.98 ± 1.17%	40.55 ± 1.70%
Mixup [18]	47.43 ± 0.92%	36.17 ± 1.36%
VAT [34]	36.03 ± 2.82%	26.11 ± 1.52%
Mean-Teacher [16]	47.32 ± 4.71%	42.01 ± 5.86%
MixMatch [19]	11.08 ± 0.87%	9.65 ± 0.94%
Ours	**10.59 ± 0.41%**	**9.40 ± 0.25%**

dataset is 26.42% [56]. **(3) SVHN.** The street view house numbers (SVHN) is another benchmark dataset for semi-supervised learning. It consists of 32×32 RGB images of the house number from 10 classes. There are 73257 labeled training samples and 26032 testing samples. Previous state-of-the-art method [19] nearly matched the performance of fully-supervised learning with 73257 labeled training set (2.59% error rate).

We also follow the same semi-supervised learning network structures in the literature [17, 19]. Specifically, we use the 28-layer Wide ResNet (ResNet-28) with 1.47 million parameters from [12, 19] on CIFAR-10 and SVHN, the 13-layer convolutional neural network (ConvNet-13) with 3.13 million parameters from [15, 16, 34, 56, 17] on CIFAR-10 and CIFAR-100 for fair comparisons with these results.

We follow the random sample strategy of MixMatch [19] and report the error rate by the mean and variance across different seeds. The results are averaged over multiple runs with different random seeds. We use the same augmentation, hyper-parameters, and training steps as MixMatch [19] and set the EMA decay rate to 0.99 for decision network of master model.

Table 3.2: Error rate percentage of ResNet-28 on SVHN compared to the state-of-the-art.

Methods	250 Labels	500 Labels
Π Model [15]	17.65 ± 0.27%	11.44 ± 0.39%
PseudoLabel [14]	21.16 ± 0.88%	14.35 ± 0.37%
Mixup [18]	39.97 ± 1.89%	29.62 ± 1.54%
VAT [34]	8.41 ± 1.01%	7.44 ± 0.79%
Mean-Teacher [16]	6.45 ± 2.43%	3.82 ± 0.17%
MixMatch [19]	3.78 ± 0.26%	3.64 ± 0.46%
Ours	**3.23 ± 0.15**%	**2.83 ± 0.08**%

Table 3.3: Error rate percentage of ConvNet-13 on CIFAR-10 and CIFAR-100 compared to the state-of-the-art.

	CIFAR-10	CIFAR-100
Methods	1000 Labels	10000 Labels
Π Model [15]	31.65 ± 1.20%	39.19 ± 0.36%
Mean-Teacher [16]	18.78 ± 0.31%	35.96 ± 0.77%
Label Propagation [62]	16.93 ± 0.70%	35.92 ± 0.47%
MT + FSWA [56]	15.58 ± 0.12%	33.62 ± 0.54%
Dual-Student [17]	14.17 ± 0.38%	32.77 ± 0.24%
MixMatch [19]	9.08 ± 0.05%	32.95 ± 0.06%
Ours	**8.74 ± 0.20**%	**31.16 ± 0.08**%

3.4.2 Performance Comparison with Existing Methods

To evaluate the effectiveness of our LMCS method, we randomly extract 250 and 500 labeled samples from the training set of SVHN and CIFAR-10, then compare the performance of our proposed method on the benchmark architecture Resnet-28 [12, 19] with these state-of-the-art methods. Tabel 3.1 shows the error rates of our method in comparison with other methods based on 250 and 500 labeled samples on CIFAR-10. We can see that our proposed method outperforms all other methods, including the current state-of-the-art algorithm MixMatch from Google [19]. On SVHN, we also perform experiments with 250 and 500 labeled samples to show the generalization ability of our proposed method. We provide the results in Table 3.2 and show that our method outperforms existing methods, reducing the error rate by 2.83% with 500 labeled samples and 3.23% with only 250 labeled samples, which is significant.

Early work on semi-supervised learning also used a ConvNet-13 architecture [15, 16, 17, 34, 56] as the benchmark architecture. To demonstrate the effectiveness of our LMCS on different benchmark architecture, we randomly extract 1000 labeled samples from CIFAR-10 and 10000 labeled samples from CIFAR-100 and conduct further experiments on ConvNet-13. Table 3.3 shows the error rates of our LMCS method and performance comparison of ConvNet-13 with other state-of-the-art methods [15, 16, 17, 34, 56]. We can see that our proposed method improves the performance and achieves a new state-of-the-art error rate 8.74% on CIFAR-10 with 1000 labeled samples. On CIFAR-100, we select 100 labeled samples from each of 100 classes with a total of 10000 labeled samples. Table 3.3 shows that our method improves the state-of-the-art result from 32.95% to 31.16%.

Table 3.4: Error rate percentage of ResNet-28 on CIFAR-10 compared to the state-of-the-art on small set of labeled samples.

Methods	80 Labels	100 Labels	250 Labels	500 Labels
MixMatch [19]	26.44 ± 1.67%	19.61 ± 2.10%	11.08 ± 0.87%	9.65 ± 0.94%
Ours	**16.05 ± 1.10%**	**12.75 ± 0.12%**	**10.59 ± 0.41%**	**9.40 ± 0.25%**

Table 3.5: Error rate percentage of ResNet-28 on SVHN compared to the state-of-the-art on small set of labeled samples.

Methods	100 Labels	250 Labels	500 Labels
MixMatch [19]	3.87 ± 1.86%	3.78 ± 0.26%	3.64 ± 0.46%
Ours	**3.51 ± 0.09%**	**3.23 ± 0.15%**	**2.83 ± 0.08%**

3.4.3 Performance Evaluations on Small Sets of Labeled Samples

In the following, we evaluate the efficiency of our LMCS method on small sets of labeled samples, which is very important yet very challenging in semi-supervised learning. To this end, we reduce the size of labeled set to 80 and 100 on CIFAR-10, 100 on SVHN and 4000, 5000 on CIFAR-100, which is much smaller than the sizes of training sets used in existing semi-supervised learning methods reported in the literature [15, 16, 17, 34, 56]. Existing semi-supervised methods compare their performance on larger sets of 500, 1000 and 2000 labeled samples, but they did not provide the evaluation on small labeled sets.

In the following experiments, we evaluate the performance of our LMCS method on ResNet-28. Table 3.4 and Table 3.5 show the results on the CIFAR-10 dataset and

Table 3.6: Error rate percentage of ConvNet-13 on CIFAR-10 compared to the state-of-the-art on small set of labeled samples.

Methods	100 Labels	250 Labels
MixMatch [19]	33.25 ± 4.05%	13.51 ± 2.04%
Ours	**17.71 ± 2.46%**	**12.11 ± 0.32%**

SVHN dataset with the size of training set reduced from 250 to 100. We also evaluate the performance of our LMCS with 80 labels on CIFAR-10. For the convenience of comparison, we copy over the results of 250 and 500 from Table 3.1 and Table 3.2. We can see that our method significantly outperforms the MixMatch [19], reducing the error rate from 26.44% to 16.05% with 80 labeled samples. On SVHN, Table 3.5 shows that our proposed method could also improve the result on SVHN with only 100 labels from 3.87% to 3.51%. We also conduct experiments with CIFAR-10 and CIFAR-100 on ConvNet-13. Table 3.6 shows that our LMCS achieves 17.71% with 100 labels and 13.51% with 250 labels on CIFAR-10. The performance on small set of labeled samples scale outperforms MixMatch by a large margin. Table 3.7 shows the results with only 40 and 50 labels per class (4000 and 5000 labels in total) on CIFAR-100. We are able to improve the best result from 40.83% and 38.41% to 38.83% and 36.70%, respectively.

In the above tables of performance comparison, we only have the MixMatch [19] algorithm for comparison for the following reason: (1) as reported in [19], MixMatch is the current state-of-the-art semi-supervised learning method, significantly outperforming other recent methods in the literature, such as the Mean Teacher [16] and VAT method [34]. (2) MixMatch has published their code. We have managed to run their code and achieve claimed results in the paper. But in Table 3.7 for CIFAR-100, we did include another very recent method, Label Propagation [62], since they have published their result on ConvNet-13 with 4000 labeled samples. But for the CIFAR-10, they did not provide codes and models with ResNet-28. It was very challenging for us to produce their results on these new sets with small set of labeled samples.

Table 3.7: Error rate percentage of ConvNet-13 on CIFAR-100 compared to the state-of-the-art on small set of labeled samples.

Methods	4000 Labels	5000 Labels
Label Propagation [62]	43.73 ± 0.20%	-
MixMatch [19]	40.83 ± 0.14%	38.41 ± 0.10%
Ours	**38.82 ± 0.16%**	**36.70 ± 0.50%**

Table 3.8: Error rate comparison of different sample discovery methods on CIFAR-10.

Sample Discovery Methods	Error Rate (%)
No Sample Discovery (Baseline)	17.64%
Minimum Distance	17.44%
Confined Maximum Entropy	**12.63%**
Confined Minimum Entropy	15.71%
Unconfined Maximum Entropy	19.13%
Unconfined Minimum Entropy	16.18%

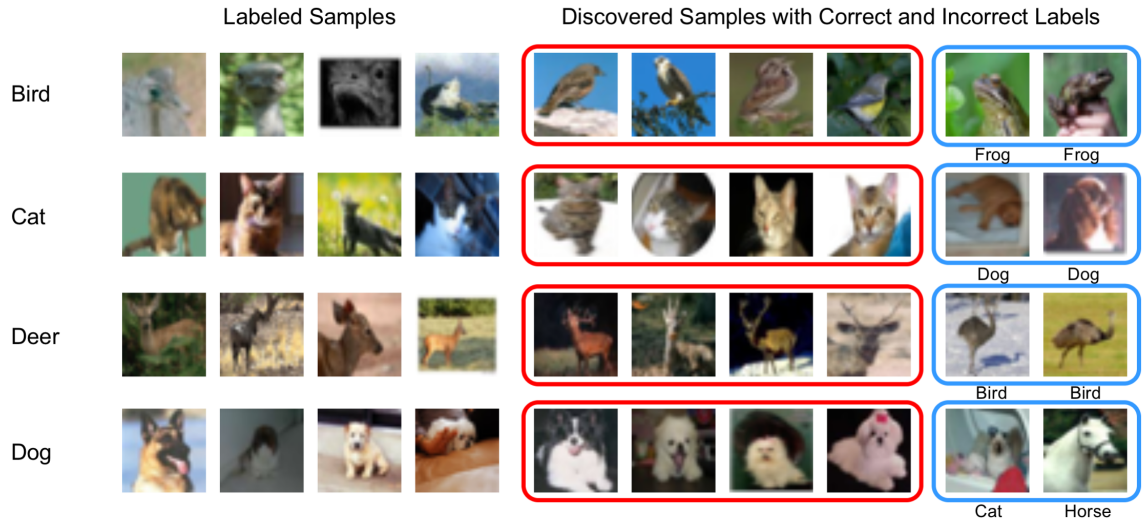


Figure 3.8: Examples of labeled samples and discovered samples by our confined maximum entropy search on CIFAR-10. Discovered samples with correct and incorrect labels are highlighted with red and blue, respectively.

3.4.4 Ablation Studies and Algorithm Analysis

In this section, we provide ablation study results on CIFAR-10 to perform in-depth analysis of our algorithm and its different components.

(1) Impact of different selection methods for critical sample discovery. In our critical sample discovery, we find that confined samples with maximum entropy of soft labels can provide useful information for the model learning. In this ablation study, we are trying to understand the following questions: (a) why is it necessary to choose confined samples? (b) Why is it necessary to use the maximum entropy principle? To answer these question, we conduct performance comparison with the following five sample discovery methods. **(1) Confined Maximum Entropy.** Samples are chosen from the confined set with maximum entropy. **(2) Confined Minimum Entropy.** Samples are chosen from the confined set but with minimum entropy. **(3) Unconfined Maximum Entropy.** Samples are chosen from all unlabeled images based on the maximum entropy principle. **(4) Unconfined Minimum Entropy.** Samples are chosen from all unlabeled images but with minimum entropy. **(5) Minimum Distance.** We choose samples with minimum distance to the class center.

In the following experiments, we set the number of labeled samples to be 100 on the CIFAR-10 dataset. The error rate of the initial model is 17.64%. The next task is to discover 400 unlabeled samples as the critical samples using the above five different methods. Figure 3.9 shows the comparison results. We can see that the error rates with confined methods perform similarly in the first 200 epochs since they couple the initial labeled samples with the high accuracy newly discovered labels. Our confined maximum entropy search method achieves the best performance after 200 epochs.

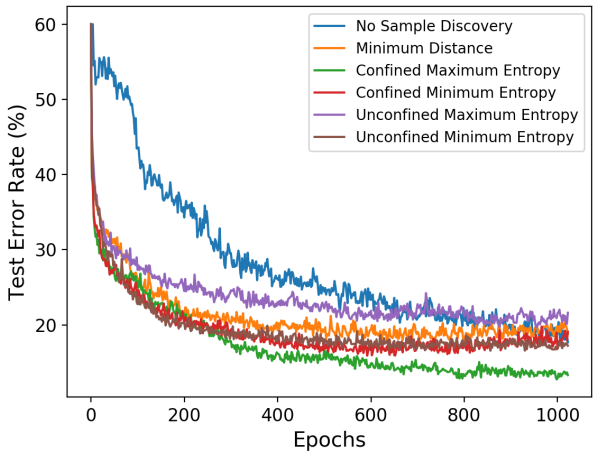


Figure 3.9: Error rate test curve comparison of different sample discovery methods on CIFAR-10.

Table 3.9: Error rate comparison of different initial models with ResNet-28 on CIFAR-10.

Methods	100 Labels	250 Labels
Random Initialized Model	11.93%	10.52%
Self-Supervision Pre-trained Model	**11.63**%	**10.38**%

Samples with minimum distance might be too similar to the initial labeled samples and their contribution to the model learning is very limited. Table 3.8 summarizes the classification error rates using these five different critical sample discovery methods. We can see that the confined maximum entropy method performs best. Confined methods are much better than those unconfined discovery methods.

(2) Efficiency of critical sample discovery and relationship to the initial

Table 3.10: Ablation study: test error rates of ResNet-28 on CIFAR-10 with 80, 100, 250 labels to show performance of different components in our LMCS method.

Ablation	80 Labels	100 Labels	250 Labels
Our LMCS Method (Full Algorithm)	**14.96**%	**11.93**%	**10.52**%
− without Critical Sample Discovery	24.75%	17.64%	11.00%
− without Learned Model Composition	26.52%	21.70%	11.46%

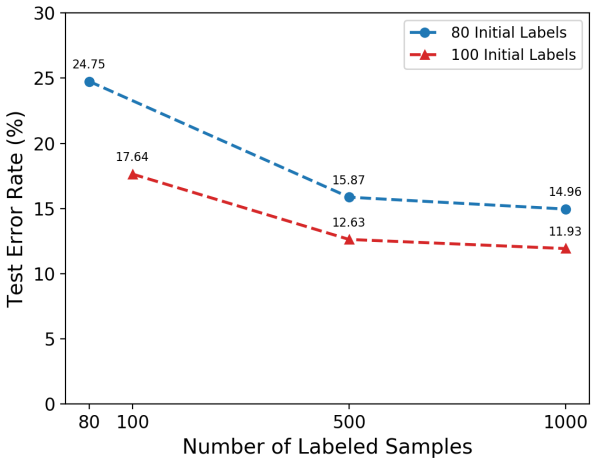


Figure 3.10: Error rate comparison of different initial models with 80 labels and 100 labels on CIFAR-10.

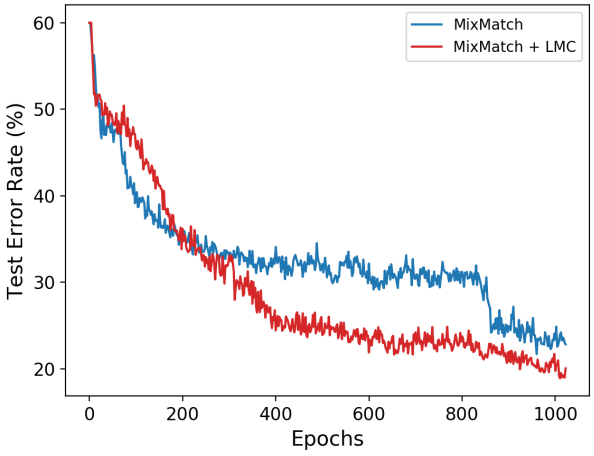


Figure 3.11: Error rate test curve comparison between MixMatch and MixMatch + LMC on CIFAR-10 with 100 labels.

model. In this ablation study, we study how the size of labeled samples and its corresponding initial model affect the overall semi-supervised learning performance. For example, given two different sets of labeled samples from CIFAR-10, one with 80 labels and the other with 100 labels. Their initial models will have much different performance. As shown in Figure 3.10, the initial model with 80 labels has a classification error rate of 24.75%, but the model with 100 labels has an error rate of 17.64%. Starting with these two models and two sets of labeled samples, we use our critical sample discovery method to discover new samples. When the number of samples, original labeled samples plus discovered samples, reaches to 500, the gap between their error rates is significantly reduced from 7.11% to 3.24%. When the total number reaches to 1000, the gap between these models is further reduced to 3.03%. From this ablation study, we can see that our critical sample discovery method does not heavily depend on the initial model. It is able to find helpful critical samples to gradually improve its performance.

Figure 3.8 shows more examples of the labeled samples and the newly discovered samples. The discovered samples with correct and incorrect labels are highlighted with red and blue, respectively. The newly discovered samples with correct labels are visually different from the labeled samples, since they are close to the decision boundary, while the discovered samples with incorrect labels are also difficult for human to tell the difference.

(3) Efficiency of the learned model composition. Another important component is our learned model composition (LMC). We compare two methods: MixMatch and MixMatch + LMC. In this way, we can demonstrate the efficiency of our LMC method. The constructed master model from LMC improves the discrimina-

tive power on the unlabeled samples when converters are well trained to weight the feature maps from different student networks. Figure 3.11 shows the test error rates of these two methods on the CIFAR-10 with same 100 labeled samples over different training epochs. We can see that our LMC method is able to significantly reduce the classification error rate on top of the MixMatch method, which is already powerful enough.

(4) Performance of self-supervision pre-trained model. In this ablation study, we analyze how the pre-trained model with self-supervision tasks affects the overall semi-supervised learning performance. Self-supervised learning utilizes pretext tasks to train the network with unlabeled samples. In our experiment, we use the pretext task of classifying image rotations with four degrees (0, 90, 180, 270) [26] to train the network and use this pre-trained model as the initial model for the semi-supervised learning. Table 3.9 shows the performance comparison using the ResNet-28 with and without the self-supervision pre-trained models. We can see that the improvement achieved by the self-supervision pre-trained model is very small.

(5) Performance summary of different algorithm components. Our LMCS method has two major components: learned model composition and critical sample discovery. In this ablation study, we aim to identify the contribution of each algorithm component. Table 3.10 summarizes the results on CIFAR-10 with 80, 100, and 250 labels using three different method configurations: (1) the full algorithm of LMCS, (2) our LMCS method without critical sample discovery, and (3) our LMCS method without learned model composition and without critical sample discovery. We can see that these two component are both very important, contributing to the overall performance significantly.

3.5 Conclusion

In this work, we have developed a new method called *learned model composition with critical sample look-ahead* (LMCS) to achieve successful semi-supervised learning on small sets of labeled samples. We have introduced a new learned model composition structure to construct the master network from student models of past steps through a network learning process. We have also developed a new method, called *confined maximum entropy search* to discover new critical samples near the model decision boundary to refine the master network. Our extensive experiments have demonstrated that the proposed LMCS network outperforms the state-of-the-art semi-supervised learning methods, especially on small labeled training sets. Our ablation studies have demonstrated that the learned model composition and critical sample discovery are tightly coupled, allowing the network to learn from a small set of labeled samples, discover new critical samples to enlarge its training set, evolve the network model using a learning process, and gradually improve its semi-supervised learning performance.

Chapter 4

Unsupervised Deep Metric Learning with Transformed Attention Consistency and Contrastive Clustering Loss

4.1 Introduction

In this work, we propose to explore a new approach to unsupervised deep metric learning. We observe that existing methods for unsupervised metric learning focus on learning a network to analyze the input image itself. As we know, when examining and classifying images, human eyes compare images back and forth in order to identify discriminative features [69]. In other words, comparison plays an important role in human visual learning. When comparing images, they often pay attention to certain keypoints, image regions, or objects which are discriminative between image classes but highly consistent across image within classes. Even when the image is being

transformed, the attention areas will be consistent. To further illustrate this, we provide three examples in Figure 4.1. In (a), human eyes can easily tell the top image A of the first column and the bottom image B are the same bird since they have the same visual characteristics. The attention will be on the feather texture and head shape. In the pixel domain, A and B are up to a spatial transform, specifically, cropping plus resizing. When the human eyes moves from image A to its transformed version B, the attention will be also transformed so that it can be still focused on the head and feather. If we represent this attention using the attention map in deep neural networks, the attention map $\mathcal{M}(A)$ for image A and the attention map $\mathcal{M}(B)$ for image B should also follow the same transform, as shown in the second column of Figure 4.1(a). We can also see this consistency of attention across image under different transforms in other examples in Figs. 4.1(b) and (c).

This lead to our idea of transformed attention consistency. Based on this idea, we develop a new approach to unsupervised deep metric learning based on image comparison. Specifically, using this consistency, we can define a pairwise self-supervision loss, allowing us to learn a Siamese deep neural network to encode and compare images against their transformed or matched pairs. To further enhance the inter-class discriminative power of the feature generated by this network, we adapt the concept of triplet loss from supervised metric learning to our unsupervised case and introduce the contrastive clustering loss. Our extensive experimental results on benchmark datasets demonstrate that our proposed method outperforms current state-of-the-art methods by a large margin.

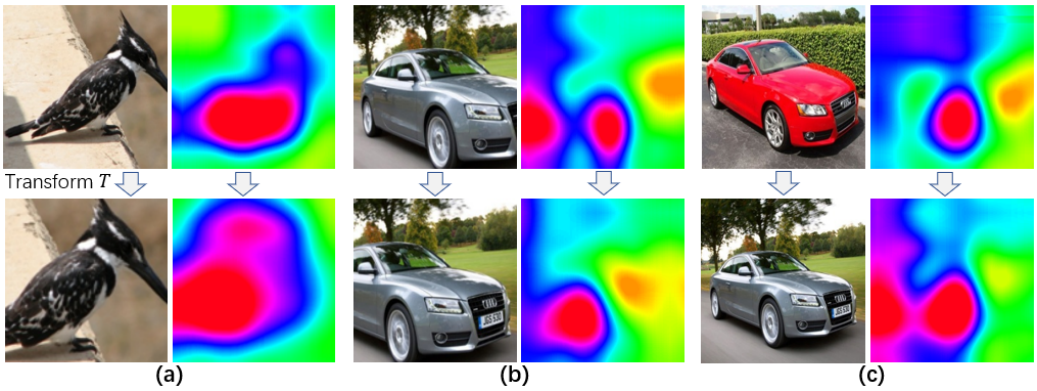


Figure 4.1: Consistency of visual attention across images under transforms.

4.2 Related Work and Major Contributions

This work is related to deep metric learning, self-supervised representation learning, unsupervised metric learning, and attention mechanisms.

(1) Deep metric learning. The main objective of deep metric learning is to learn a non-linear transformation of an input image by deep neural networks. In a common practice [70, 71], the backbone in deep metric learning can be pre-trained on 1000 classes ImageNet [72] classification, and is then jointly trained on the metric learning task with an additional linear embedding layer. Many recent deep metric learning methods are built on pair-based [73, 74, 75] and triplet relationships [76, 77, 78]. Triplet loss [77] defines a positive pair and a negative pair based on the same anchor point. It encourages the embedding distance of positive pair to be smaller than the distance of negative pair by a given margin. Multi-similarity loss [70] considers multiple similarities and provides a more powerful approach for mining and weighting informative pairs by considering multiple similarities. The ability of mining informative pairs in existing methods is limited by the size of mini-batch.

Cross-batch memory (XBM) [79] provides a memory bank for the feature embeddings of past iterations. In this way, the informative pairs can be identified across the dataset instead of a mini-batch.

(2) Self-supervised representation learning. Self-supervised representation learning directly derives information from unlabeled data itself by formulating predictive tasks to learn informative feature representations. DeepCluster [21] uses k-means clustering to assign pseudo-labels to the features generated by the deep neural network and introduces a discriminative loss to train the network. Gidaris *et al.* [26] explore the geometric transformation and propose to predict the angle (0°, 90°, 180°, and 270°) of image rotation as a four-way classification. Zhang *et al.* [22] propose to predict the randomly sampled transformation from the encoded features by Auto-encoding transformation (AET). The encoder is forced to extract the features with visual structure information, which are informative enough for the decoder to decode the transformation. Self-supervision has been widely used to initialize and pre-train backbone on unlabeled data, and is then fine-tuned on a labeled training data for evaluating different tasks.

(3) Unsupervised metric learning. Unsupervised metric learning is a relatively new research topic. It is a more challenging task since the training classes have no labels and it does not overlap with the testing classes. Iscen *et al.* [80] propose an unsupervised method to mine hard positive and negative samples based on manifold-aware sampling. The feature embedding can be trained with standard contrastive and triplet loss. Ye *et al.* [71] propose to utilize the instance-wise relationship instead of class information in the learning process. It optimizes the instance feature embedding directly based on the positive augmentation invariant and negative separated

properties.

(4) Attention mechanism. The goal of the attention mechanism is to capture the most informative feature in the image. It explores important parts of features and suppress unnecessary parts [81, 82, 83]. Convolutional block attention module (CBAM) [84] is an effective attention method with channel and spatial attention module which can be integrated into existing convolutional neural network architectures. Fu *et al.* [85] propose to produce the attention proposals and train the attention module and embedding module in an iterative two-stage manner. Chen *et al.* [86] propose the hybrid-attention system by random walk graph propagation for object attention and the adversary constraint for channel attention.

Compared to existing methods, the ***unique contributions*** of this work can be summarized as follows. (1) Unlike existing methods which focus on information analysis of the input image only, we explore a new approach for unsupervised deep metric learning based on image comparison and cross-image consistency. (2) Motivated by the human visual experience, we introduce the new approach of transformed attention consistency to effectively learn a deep neural network which can focus on discriminative features. (3) We extend the existing triplet loss developed for supervised metric learning to unsupervised learning using k-mean clustering to assign pseudo labels and memory bank to allow its access to all training samples, instead of samples in the current mini-batch. (4) Our experimental results demonstrate that our proposed approach has improved the state-of-the-art performance by a large margin.

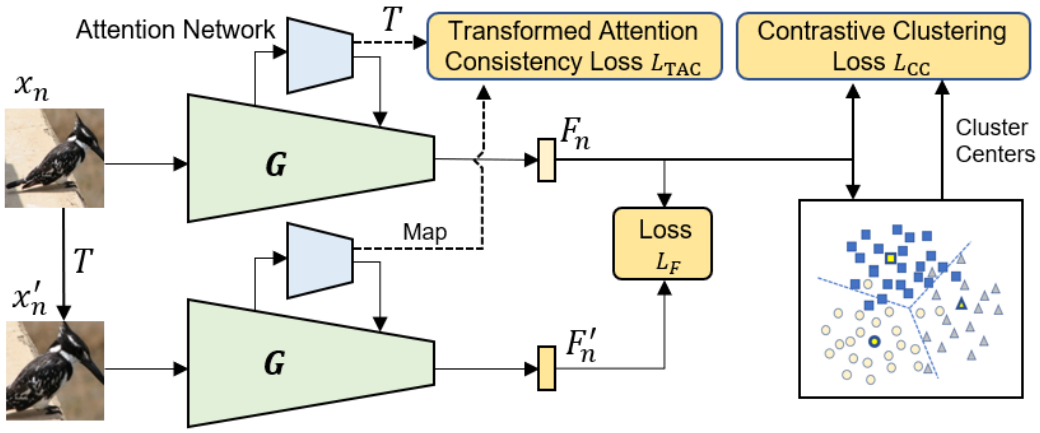


Figure 4.2: Overview of the proposed approach for unsupervised deep metric learning with transformed attention consistency and contrastive clustering loss.

4.3 Method

4.3.1 Overview

Suppose that we have a set of unlabeled images $\mathcal{X} = \{x_1, x_2, ..., x_N\}$. Our goal is to learn a deep neural network to extract their features $\mathbf{G}(x_n) \in \mathbb{R}^d$, where d is the feature dimension. Figure 2 shows the overall design of our proposed method for unsupervised deep metric learning based on transformed attention consistency and contrastive clustering loss (TAC-CCL). Given an input image x_n, we apply a transform T, which is randomly sampled from a set of image transforms $\mathcal{T}$, to x_n, to obtain its transformed version $x'_n = T(x_n)$. In our experiments, we mainly consider spatial transforms, including cropping (sub-image), rotation, zooming, and perspective transform. Each transform is controlled by a set of transform parameters. For example, the cropping is controlled by its bounding box. The perspective transform is controlled by its 6 parameters. Image pairs (x_n, x'_n) are inputs to the Siamese deep

neural network. These two identical networks will be trained to extract features F_n and F'_n for these two images. As illustrated in Figure 2, each network is equipped with an attention network to learn the attention map which will modulate the output feature map. The attention map can enforce the network to focus on discriminative local features for the specific learning tasks or datasets. Let M_n and M'_n be the attention maps for images x_n and x'_n, respectively. According to the transformed attention consistency, we shall have

$$M'_n = T(M_n). \tag{4.1}$$

Based on this constraint, we introduce the transformed attention consistency loss L_{TAC} to train the feature embedding network $\mathbf{G}$, which will be further explained in Section 4.3.3. Besides this attention consistency, we also require that the output features F_n and $F_n^{'}$ should be similar to each other since the corresponding input images x_n and $x_n^{'}$ are visually the same. To enforce this constraint, we introduce the feature similarity loss $L_F = ||F_n - F_n^{'}||_2$ which is the L_2-normal between these two features. To ensure that image features from the same class aggregate into compact clusters while image features from different classes are pushed away from each other in the high-dimensional feature space, we introduce the contrastive clustering loss L_{CC}, which will be further explained in the Section 4.3.3.

4.3.2 Baseline System

In this work, we first design a baseline system. Recently, a method called multi-similarity (MS) loss [70] has been developed for supervised deep metric learning. In this work, we adapt this method from supervised metric learning to unsupervised

metric learning using k-means clustering to assign pseudo labels. Also, the original MS method computes the similarity scores between image samples in the current mini-batch. In this work, we extend this similarity analysis to the whole training set using the approach of memory bank [79]. The features of all training samples generated by the network are stored in the memory bank by the enqueue method. When the memory bank is full, the features and corresponding labels of the oldest mini-batch are removed by the dequeue method. Using this approach, the current mini-batch has access to the whole training set. We can then compute the similarity scores between all samples in the mini-batch and all samples in the training set. Our experiments demonstrate that this enhanced similarity matrix results in significantly improved performance in unsupervised metric learning. In this work, we use this network as the baseline system, denoted by TAC-CLL (baseline).

4.3.3 Loss Functions

To further improve the performance of the baseline system, we introduce the ideas of transformed attention consistency and contrastive clustering loss, which are explained in the following.

The transformed attention consistency aims to enforce the feature embedding network $\mathbf{G}$ to focus visually important features instead of other background noise. Let $M_n(u, v)$ and $M_n^{'}(u, v)$ be the attention maps for input image pair x_n and $x_n^{'}$, where (u, v) represents a point location in the attention map. Under the transform T, this point is mapped to a new location denoted by $(T_u(u, v), T_v(u, v))$. According to (4.1), if we transform the attention map $M_n(u, v)$ for the original image x_n by T, it should match the attention map $M_n'(u, v)$ for the transformed image $x_n' = T(x_n)$.

Based on this, the proposed transformed attention consistency loss L_{TAC} is defined as follows

$$L_{TAC} = \sum_{(u,v)} |M_n(u,v) - M'_n(T_u(u,v), T_v(u,v))|^2, \tag{4.2}$$

where $u' = T_u(u,v)$ and $v' = T_v(u,v)$ are the mapped location of (u,v) in image x'_n.

The constrastive clustering loss extends the triplet loss [77] developed in supervised deep metric learning, where an anchor sample x is associated with a positive sample x_+ and a negative sample x_-. The triplet loss aims to maximize the ratio $S(x,x_+)/S(x,x_-)$, where $S(\cdot,\cdot)$ represents the cosine similarity between two features. It should be noted that this triplet loss requires the knowledge of image labels, which however are not available in our unsupervised case. To extend this triplet loss to unsupervised metric learning, we propose to cluster the image features into K clusters. In the high-dimensional feature space, we wish these clusters are compact and are well separated from each other by large margins. Let $\{C_k\}$, $1 \leq k \leq K$, be the cluster centers. Let $C_+(F_n)$ be the nearest center which has the minimum distance to the input image feature F_n and the corresponding distance is denoted by $d_+(F_n) = ||F_n - C_+(F_n)||_2$. Let $C_-(F_n)$ be the cluster center which has the second minimum distance to F_n and the corresponding distance is denoted by $d_-(F_n) = ||F_n - C_-(F_n)||_2$. If the contrastive ratio of $d_+(F_n)/d_+(F_n)$ is small, then this feature has more discriminative power. We define the following contrastive clustering loss

$$L_{CC} = \mathcal{E}_{F_n} \left\{ \frac{||F_n - C_+(F_n)||_2}{||F_n - C_-(F_n)||_2} \right\}, \tag{4.3}$$

which is the average contrastive ratio of all input image features. During the training process, the network $\mathbf{G}$, as well as the feature for each input, is progressively updated.

For example, the clustering is performed and the cluster centers are updated for every 20 epochs.

4.3.4 Transformed Attention Consistency with Cross-Images Supervision

Note that, in our proposed approach, we transform or augment the input image x_n to create its pair $x_n^{'}$. These two are from the same image source. We also notice that most of existing self-supervision methods, such as predicting locations of image patches and classifying the rotation of an image [26], and reconstructing the transform of the image [22], all focus on self-supervision information within the image itself. The reason behind this is that image patches from the same image will automatically have the same class label. This provides an important self-supervision constraint to train the network. However, this one-image approach will limit the learning capability of the network since the network is not able to compare multiple images. As we know, when human eyes are examining images to determine which features are discriminative, they need to compare multiple images to determine which set of features are consistent across images and which set of features are background noise [69]. Therefore, in unsupervised learning, it is highly desirable to utilize the information across images.

Figs. 4.3(a)-(c) show image samples from the Cars and SOP benchmark datasets. We can see that images from the same class exhibit strong similarity between images, especially in the object regions. The question is how to utilize these unlabeled images to create reliable self-supervision information for unsupervised learning? In this work, we propose to perform keypoint or sub-image matching across images. Specifically, as illustrated in Figure 4.3(d) and (e), for a given image sample I_n, in the pre-processing

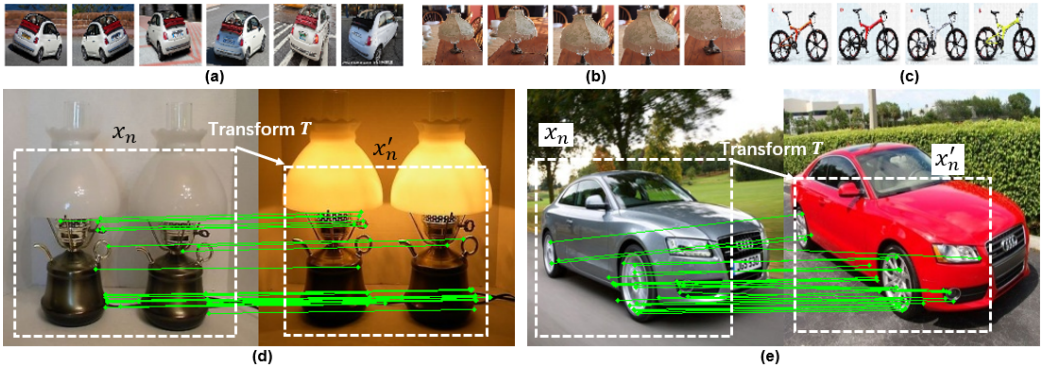


Figure 4.3: Sub-image matching for cross-image supervision.

stage, we perform affine-SIFT [87] keypoint matching between I_n and other images in the dataset and find the top matches with confidence scores about a very high threshold. We then crop out the sub-images containing high-confidence keypoints as x_n and x'_n which are related by a transform T. This high-confidence constraint aims to ensure that x_n and x'_n are having the same object class or semantic label. In this way, for each image in the k-means cluster, we can find multiple high-confidence matched sub-images. For example, on the CUB dataset, for top-2 matching in each cluster, the label accuracy is 77.0%, which is much higher than the true positive pair rate obtained by k-means clustering (39.1%). This will significantly augment the training set, establish cross-image self-supervision, and provide significantly enhanced visual diversity for the network to learn more robust and discriminative features. In this work, we combine this cross-image supervision with the transformed attention consistency. Let $\{(u_i, v_i)\}$ and $\{(u'_i, v'_i)\}$, $1 \leq i \leq N$, be the set of matched keypoints in x_n and x'_n. We wish that, within the small neighborhoods of these matched keypoints, the attention maps M_n and M'_n are consistent. To define a small neighborhood around

a keypoint (u_i, v_i), we use the following 2-D Gaussian kernel,

$$\phi(u - u_i, v - v_i) = e^{-\frac{(u-u_i)^2}{2\sigma_u^2} - \frac{(v-v_i)^2}{2\sigma_v^2}}. \tag{4.4}$$

Let

$$\Gamma(u, v) = \sum_{i=1}^{M} \phi(u - u_i, v - v_i), \quad \Gamma'(u, v) = \sum_{i=1}^{M} \phi(u - u_i', v - v_i'), \tag{4.5}$$

which define two masks to indicate the neighborhood areas around these matched keypoints in these two attention maps. The extended transformed attention consistency becomes

$$L_{TAC} = \sum_{(u,v)} |M_n(u, v) \cdot \Gamma(u, v) - M_n'(u, v) \cdot \Gamma'(u, v)|^2, \tag{4.6}$$

which compares the difference between these two attention maps around these matched keypoints. Compared to the label propagation method developed for semi-supervised learning [46, 40], our cross-image supervision method is unique in the following aspects: (1) it discovers sub-images of the same label (with very high probability) from unlabeled images. (2) It establishes the transform between these two sub-images and combines with the transformed attention consistency to achieve efficient unsupervised deep metric learning.

4.4 Experimental Results

In this section, we conduct extensive experiments on benchmark datasets in image retrieval settings to evaluate the proposed TAC-CCL method for unsupervised deep metric learning.

4.4.1 Datasets

We follow existing papers on unsupervised deep metric learning [71] to evaluate our proposed methods on the following three benchmark datasets. **(1) CUB-200-2011 (CUB)** [88] is composed of 11,788 images of birds from 200 classes. The first 100 classes (5864 images) are used for training, with the rest 100 classes (5924 images) for testing. **(2) Cars-196 (Cars)** [89] contains 16,185 images of 196 classes of car models. We use the first 98 classes with 8054 images for training, and remaining 98 classes (8131 images) for testing. **(3) Stanford Online Products (SOP)** [75] has 22,634 classes (120,053 images) of online products. We use the first 11,318 products (59,551 images) for training and the remaining 11,316 products (60,502 images) for testing. The training classes are separated from the test classes. We use the standard image retrieval performance metric (Recall@K), for performance evaluations and comparisons.

4.4.2 Implementation Details

We implement our proposed method by PyTorch and follow the standard experimental settings in existing papers [75, 70, 71] for performance comparison. We use the same GoogLeNet [90] pre-trained on ImageNet as the backbone network [91, 75, 92] and a CBAM [84] attention module is placed after the *inception_5b* layer. A fully connected layer is then added on the top of the network as the embedding layer. The default dimension of embedding is set as 512. For the clustering, we set the number of clusters K to be 100 for the CUB and Cars datasets, and $K = 10000$ for the SOP dataset. For each batch, we follow the data sampling strategy in multi-similarity loss [70] to sample 5 images per class. For data augmentation, images in the training set

Table 4.1: Recall@K (%) performance on CUB and Cars datasets in comparison with other methods.

Methods	Backbone	CUB				Cars			
		R@1	R@2	R@4	R@8	R@1	R@2	R@4	R@8
Supervised Methods									
ABIER [94]	GoogLeNet	57.5	68.7	78.3	86.2	82.0	89.0	93.2	96.1
ABE [95]	GoogLeNet	60.6	71.5	79.8	87.4	85.2	90.5	94.0	96.1
Multi-Similarity [70]	BN-Inception	65.7	77.0	86.3	91.2	84.1	90.4	94.0	96.5
Unsupervised Methods									
Examplar [96]	GoogLeNet	38.2	50.3	62.8	75.0	36.5	48.1	59.2	71.0
NCE [97]	GoogLeNet	39.2	51.4	63.7	75.8	37.5	48.7	59.8	71.5
DeepCluster [21]	GoogLeNet	42.9	54.1	65.6	76.2	32.6	43.8	57.0	69.5
MOM [80]	GoogLeNet	45.3	57.8	68.6	78.4	35.5	48.2	60.6	72.4
Instance [71]	GoogLeNet	46.2	59.0	70.1	80.2	41.3	52.3	63.6	74.9
TAC-CCL (baseline)	GoogLeNet	**53.9**	**66.2**	**76.9**	**85.8**	**43.0**	**53.8**	**65.3**	**76.0**
TAC-CCL	GoogLeNet	**57.5**	**68.8**	**78.8**	**87.2**	**46.1**	**56.9**	**67.5**	**76.7**
Gain		**+11.3**	**+9.8**	**+8.7**	**+7.0**	**+4.8**	**+4.6**	**+3.9**	**+1.8**

are randomly cropped at size 227 $\times$ 227 with random horizontal flipping, while the images in testing set is center cropped. Adam optimizer [93] is used in all experiments and the weigh decay is set as $5e^{-4}$.

4.4.3 Performance Comparisons with State-of-the-Art Methods

We compare the performance of our proposed methods with the state-of-the-art unsupervised methods on image retrieval tasks. The mining on manifolds (MOM) [80] and the invariant and spreading instance feature method (denoted by Instance) [71] are current state-of-the-art methods for unsupervised metric learning. They both use the GoogLeNet [90] as the backbone encoder. In the Instance paper [71], the authors have also implemented three other state-of-the-art methods originally developed for feature learning and adapted them to unsupervised metric learning tasks: Examplar [96], NCE (Noise-Contrastive Estimation) [97], and DeepCluster [21]. We include the

results of these methods for comparisons. We have also included the performance of recent supervised deep metric learning methods for comparison so that we can see the performance difference between unsupervised metric learning and supervised one. These methods include: ABIER [94], and ABE [95], and MS (Multi-Similarity) [70]. Both ABIER and ABE methods are using the GoogLeNet as the backbone encoder. The MS method is using the BN-Inception network [98] as the backbone encoder.

The results for the CUB, Cars, and SOP datasets are summarized in Tables 4.1 and 4.2, respectively. We can see that our proposed TAC-CCL method achieves new state-of-the-art performance in unsupervised metric learning on both fine-grained CUB and Cars datasets and the large-scale SOP dataset. On the CUB dataset, our TAC-CCL improves the Recall@1 by 11.3% and is even competitive to some supervised metric learning methods, e.g., ABIER [94]. On the Cars dataset, our TAC-CCL outperforms the current state-of-the-art Instance method [71] by 4.8%. On SOP, our method achieves 63.9% and outperforms existing methods by a large margin of 15%. For other Recall@K rates with large values of k, the amount of improvement is also very significant. Note that our baseline system achieves a large improvement over existing methods. The proposed TAC-CCL approach further improves upon this baseline system by another 1.4-3.6%.

Figure 4.4 shows examples of retrieval results from the CUB, Cars, and SOP datasets. In each row, the first image highlighted with a blue box is the query image. The rest images are the top 15 retrieval results. Images highlighted with red boxes are from different classes. It should be noted that some classes have very small number of samples. We can see that our TAC-CCL can learn discriminitive features to achieve satisfying retrieval results, even for these challenging tasks. For example, at the

Table 4.2: Recall@K (%) performance on SOP dataset in comparison with other methods.

Methods	Backbone	SOP		
		R@1	R@10	R@100
Supervised Methods				
ABIER [94]	GoogLeNet	74.2	86.9	94.0
ABE [95]	GoogLeNet	76.3	88.4	94.8
Multi-Similarity [70]	BN-Inception	78.2	90.5	96.0
Unsupervised Methods				
Examplar [96]	GoogLeNet	45.0	60.3	75.2
NCE [97]	GoogLeNet	46.6	62.3	76.8
DeepCluster [21]	GoogLeNet	34.6	52.6	66.8
MOM [80]	GoogLeNet	43.3	57.2	73.2
Instance [71]	GoogLeNet	48.9	64.0	78.0
TAC-CCL (baseline)	GoogLeNet	**62.5**	**76.5**	**87.2**
TAC-CCL	GoogLeNet	**63.9**	**77.6**	**87.8**
Gain		**+15.0**	**+13.6**	**+9.8**

first row of the SOP dataset, our model is able to learn the glass decoration feature under the lampshade, which is a unique feature of the query images. In addition, the negative retrieved results are also visually closer to the query images.

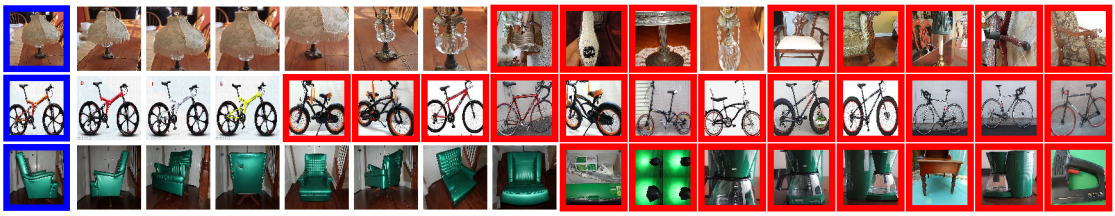
4.4.4 Ablation Studies

In this section, we conduct ablation studies to perform in-depth analysis of our proposed method and its different components.

(1) Impact of the number of clusters. The proposed contrastive clustering loss is based on clustering in the feature space. The number of clusters K is a critical parameter for the proposed method since it determines the number of pseudo labels. We conduct the following ablation study experiment on the CUB data to study the impact of K. The first plot in Figure 4.5(a) shows the Recall@1 results with different values of K: 50, 100, 200, 500, and 1000. The other three plots show the results for

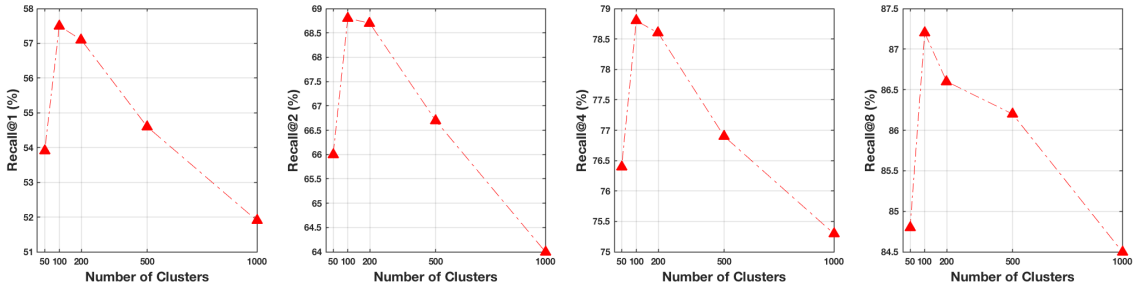
(a) CUB

(b) Cars

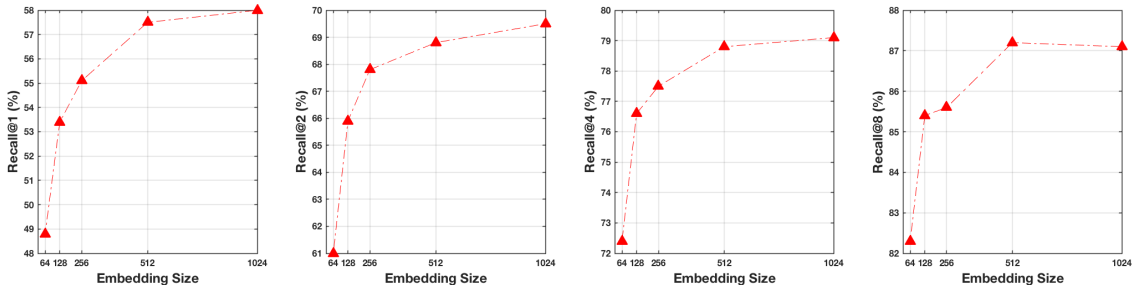


(c) SOP

Figure 4.4: Retrieval results of some example queries on CUB, Cars, and SOP datasets. The query images and the negative retrieved images are highlighted with blue and red.



(a) Different number of clusters



(b) Different embedding sizes

Figure 4.5: Recall@K (%) performance on CUB dataset in comparison with different number of clusters and different embedding size.

Table 4.3: Recall@K (%) performance on SOP dataset using Resnet-18 network without pre-trained parameters.

Methods	SOP		
	R@1	R@10	R@100
Random	18.4	29.4	46.0
Examplar [96]	31.5	46.7	64.2
NCE [97]	34.4	49.0	65.2
MOM [80]	16.3	27.6	44.5
Instance [71]	39.7	54.9	71.0
Ours	**47.0**	**62.6**	**77.5**

Recall@2, 4, and 8. We can see that, on this dataset, the best value of K is 100, which is the number of test classes in the CUB dataset. The performance drops when K increases. This study suggests that the best value of K is close to the truth number test classes of the dataset.

(2) Impact of different embedding sizes. In this ablation study, we follow existing supervised metric learning methods [70, 94] to study the impact of different embedding sizes, or the size of the embedded feature. For example, the feature size ranges from 64, 128, 256, 512, to 1024. The first plot of Figure 4.5(b) shows the Recall@1 results for different embedding size. The results for Recall@2, 4, 8 are shown in the other three plots. We can see that unsupervised metric learning performance increases with the embedding size since it contains more feature information with enhanced discriminative power.

(3) Impact of the pre-trained model. We follow the recent state-of-the-art unsupervised metric learning Instance method [71] and evaluate the performance of our proposed method on the large-scale SOP dataset by using the Resnet-18 network without pre-trained parameters. From Table 4.3, we can see our proposed method outperforms Instance method [71] by more than 7%.

Table 4.4: The performance of different components from our TAC-CCL method on CUB, Cars, and SOP datasets.

	CUB				**Cars**				**SOP**		
	R@1	R@2	R@4	R@8	R@1	R@2	R@4	R@8	R@1	R@10	R@100
Baseline	53.9	66.2	76.9	85.8	43.0	53.8	65.3	76.0	62.5	76.5	87.2
+CCL	55.7	67.8	77.5	86.2	44.7	55.6	65.9	75.7	63.0	76.8	87.2
+TAC	**57.5**	**68.8**	**78.8**	**87.2**	**46.1**	**56.9**	**67.5**	**76.7**	**63.9**	**77.6**	**87.8**

(4) Performance contributions of different algorithm components. Our proposed system has three major components: the baseline system for unsupervised deep metric learning, transformed attention consistency (TAC), and contrastive clustering loss (CCL). In this ablation study, we aim to identify the contribution of each algorithm component on different datasets. Table 4.4 summarizes the performance results on the CUB, Cars, and SOP datasets using three different method configurations: (1) the baseline system, (2) baseline with CCL, and (3) baseline with CCL + TAC. We can see that both the CCL and TAC approaches significantly improve the performance.

4.5 Conclusion

In this work, we have developed a new approach to unsupervised deep metric learning based on image comparisons, transformed attention consistency, and constrastive clustering loss. This transformed attention consistency leads to a pairwise self-supervision loss, allowing us to learn a Siamese deep neural network to encode and compare images against their transformed or matched pairs. To further enhance the inter-class discriminative power of the feature generated by this network, we have adapted

the concept of triplet loss from supervised metric learning to our unsupervised case and introduce the contrastive clustering loss. Our extensive experimental results on benchmark datasets demonstrate that our proposed method outperforms current state-of-the-art methods by a large margin.

Chapter 5

Spatial Assembly Networks for Image Representation Learning

5.1 Introduction

A key challenge in computer vision and machine learning is to construct or learn discriminative representations for the semantic content of images, which should be invariant to changes in camera positions, perspective transforms, object scales, poses, part deformations, spatial displacement, and scene configurations [99, 100]. Recently, deep neural networks have emerged as a powerful approach for visual learning and representation. With its shared weights for convolution across different spatial locations, average or maximum pooling, coupled with sufficient training image augmentations, they are able to generate relatively invariant features or decisions under small spatial variations or transforms. However, researchers have recognized that deep neural networks are still vulnerable to relatively large geometric transformations and spatial

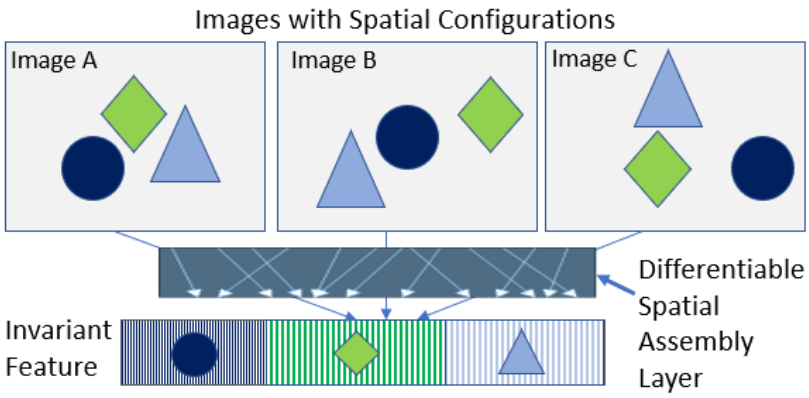


Figure 5.1: Illustration of invariant image representation learning under generic spatial variations.

variations [101]. This limitation originates from the fixed geometric structures of deep neural modules. For example, the convolution processes the input image or feature images on a fixed grid structure with a small reception field. The pooling layers then process the outputs from the convolution layers with a fixed spatial mapping or channel structures. There is lack of internal mechanisms to handle the flexible spatial variations, including spatial transforms and changes in object poses, spatial layout, and scene structures [102]. In their recent study [103], Kayhan and Gemert even found out that deep neural networks are exploiting the absolution spatial locations and image boundary conditions for object recognition and image classification, challenging the common assumption that convolution layers in modern CNNs are translation invariant.

Learning invariant features and visual representation with deep neural networks has become an important yet challenging research problem. Recent research has been focusing on developing various methods on transform-aware data augmentations [104, 105], geometry adversarial training [106], and transform-invariant network

modules and structures [83, 107, 92] to improve the robustness of deep neural networks under spatial transforms of images and objects, such as affine or perspective transforms. In this work, we aim to address the challenging problem of invariant feature representation learning under more generic spatial variations, include changes in object poses, part configurations, and scene structures. For example, Figure 5.1 shows three example images with different spatial configurations of objects due to object motion. Semantically, they should be the same or belong to the same class. However, existing deep neural networks will generate different features for them. Our goal is to design a new spatial assembly network (SAN), which is able to examine the input image and perform a learned re-organization or optimized assembly of feature points from different spatial locations so as to generate invariant features for these three images. This learned spatial assembly is conditioned by feature maps of previous network layers. This differentiable module can be flexibly incorporated into existing network architectures, improving their capabilities in handling spatial variations and structural changes of the image scene, and maximizing the discriminative power of the final feature representation. We will demonstrate that the proposed SAN module is able to improve the performance of various metric / representation learning tasks, in both supervised and unsupervised learning settings.

5.2 Related Work and Major Contributions

Recently, a set of methods have been developed in the literature to improve the robustness of deep neural networks under spatial transforms of images and objects. Analytically, [108] has studied the equivalence and invariance of DNN representa-

tions to input image transformations. Lenc and and Vedaldi [101] investigated the linear relationships between representations of the original and transformed images. In [102], the training dataset is augmented with different spatial transformations and used to train different networks. The weights are the shared between networks and their generated features are then fused together using maximum pooling. To increase the robustness of deep neural network under spatial transforms, a random transformation module is developed in [107] to transform the feature maps obtained from the neural network and suppress its sensitivity to spatial transforms in the input image. The spatial transformer network has been developed in [83] which is able to locate and predict the spatial transforms of objects in the scene based on previous feature maps and re-align the feature maps of objects based on these transforms. This new spatial transformer layer can be inserted into existing network and used to improve the robustness of deep neural network under spatial transforms. To handle object-level spatial variations, an end-to-end network architecture that perform joint detection, orientation estimation, and feature description has been explored in [109]. To achieve adaptive part localization for objects with different shapes, deformable convolution and pooling are developed in [110], which adds 2D offsets to the grid sampling locations and bin positions in the standard convolution and RoI (region of Interest) pooling. Gens and Domingos [111] proposed a generalization of CNN that forms feature maps over arbitrary symmetry groups based on the theory of symmetry groups in [111], resulting in feature maps that were more invariant to symmetry groups. Sohn and Lee [112] proposed a transform-invariant restricted Boltzmann machine (RBM) which is able to generate compact and invariant representation of the input image using probabilistic max pooling. This framework can also be extended to

unsupervised learning. To handle the orientation changes, Wang *et al.* [113] proposes to transform the weighted region features into the final orientation invariant feature vector by clustering key points into four orientation-based region proposals. The feature vectors from these four orientation regions are then fused by the aggregation module that outputs an orientation-invariant feature vector. A Laplacian pyramid network structure has been developed in [114] to produce a set of feature maps with different scales which then fused together to improve the robustness of image features under scale changes.

This work is also related to spatial permutation. Permutation optimization is a long standing problem arising in operations research, graph matching, and other applications [115]. It is also referred to as the linear and quadratic assignment problem [116]. Within the context of deep neural networks, channel shuffling has been explored in ShuffleNet [117] to improve the network performance while minimizing its computational complexity. In [100], Lyu *et al.* have developed a deep neural network approach to learn the permutation for channel shuffling. They introduced Lipschitz continuous non-convex penalty so that it can be incorporated into the stochastic gradient descent to approximate permutation. Exact permutations are then obtained by simple rounding at the end.

Compared to existing methods in the literature, our work has the following **unique novelties and contributions**. (1) Existing methods mainly focus on transform-invariant networks and image feature learning. The proposed spatial assembly goes beyond spatial image transforms. It learns to re-organize or re-assemble the feature maps across different spatial locations with the potential to handle generic spatial variations, including changes in poses, part configurations, relative motion between

objects, and scene structures. (2) This work represents one of the first efforts to explore spatial re-organization of feature maps for 2D images. The proposed spatial assembly represents a more generic feature operation than simple permutation. This differentiable module can be directly incorporated into existing deep neural network for end-to-end training to increase network robustness under spatial variations and improve the discriminative power of image features.

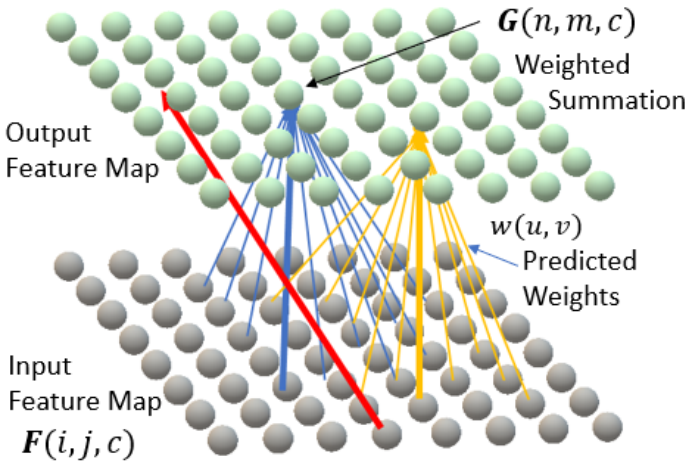


Figure 5.2: Spatial assembly of feature vectors across different spatial locations to construct the output feature map.

5.3 Methods

In this section, we describe the formulation of spatial assembly and explain its backward error propagation and gradient-based learning process.

5.3.1 Differentiable Spatial Assembly

It is a differentiable network module which learns the spatial assembly weights from previous feature maps and perform spatial assembly of the feature maps across different spatial locations based on these assembly weights within a single forward pass. The spatial assembly weights are conditioned by the feature maps of the specific input image. In other words, the spatial assembly will be different for different input images. As illustrated in Figure 5.2, let $F(i, j, c)$ be the feature map at network layer k, spatial location (i, j) and channel c. It serves as the input to the spatial assembly module. Let $G(n, m, c)$ be the output feature map after spatial assembly. The input and output feature maps share the same dimension (W_k, H_k). The output feature map is constructed using the following 2D spatial permutation operation

$$G(n, m, c) = F(i', j', c), \ (i', j') = \mathcal{P}(n, m), \tag{5.1}$$

where $(i', j') = \mathcal{P}(n, m)$ is a 2D spatial permutation. The 2D spatial permutation can be converted into a 1-D spatial permutation by introducing the location index $u = i \times W_k + j$ and $v = n \times W_k + m$. We have $0 \leq u, v \leq N$ where $N = W_k \times H_k$. With this, (5.1) can be re-written as

$$G(v, c) = F(u, c), \quad u = \mathcal{P}(v). \tag{5.2}$$

Let $\boldsymbol{P} = [w(u, v)]_{N \times N}$ be the permutation matrix, which is a binary square matrix with ones at matrix locations $(\mathcal{P}(v), v)$. It should be noted that the permutation matrix $\boldsymbol{P}$ is discrete, which has exactly a single one in every row and each column, and zeros everywhere else. These matrices form discrete points in the Euclidean

space, which makes them not differentiable.

To make this spatial permutation module differentiable, we extend this spatial permutation into spatial assembly by relaxing the binary indicator $w(u, v)$ into a continuous weight between [0, 1] which satisfies the following condition

$$\sum_{u=1}^{N} w(u, v) = 1, \ \sum_{v=1}^{N} w(u, v) = 1, \ w(u, v) > 0. \tag{5.3}$$

In spatial assembly, the output feature map is constructed by the following weighted summation

$$G(v, c) = \sum_{u=0}^{N-1} w(u, v) \cdot F(u, c), \tag{5.4}$$

as illustrated in Figure 5.2. Here, every feature vector in the output feature map $G(v, c)$ is computed using the weighted summation of the input feature vectors $F(u, c)$ at all spatial locations.

In order to achieve spatial re-organization of the feature map while maintaining the differentiable property of the spatial assembly weight function $w(u, v)$, we introduce the following two constraints. The first one is the ***minimum entropy constraint*** which aims to ensure locality of the weighting function. From a spatial transform perspective, this will ensure that one object is being moved from one location in the input feature map to another location in the output feature map. Specifically, we define the following entropy function which is the summation of entropies for all rows and all columns of the spatial assembly weight matrix:

$$\begin{aligned} \mathbb{E}[w(u, v)] = &\sum_{v} \sum_{u} \bar{w}_c(u, v) \cdot \log_2 \frac{1}{\bar{w}_c(u, v)}, \\ &+ \sum_{u} \sum_{v} \bar{w}_r(u, v) \cdot \log_2 \frac{1}{\bar{w}_r(u, v)}. \end{aligned} \tag{5.5}$$

Note that when the entropy is 0, each row or each column will have a single unit value with the rest entries to be 0. During training, this minimum entropy constraint will be used as a part of the loss function to increase the locality of the spatial assembly operation.

The second one is the ***minimum correlation constraint***: during spatial assembly, different input feature vectors are contributing to different output feature vectors. Otherwise, if one input feature vector is contributing significantly to multiple output features, it will result in significant output information redundancy, or equivalently input information loss. From the spatial transform perspective, this constraint will ensure that different objects are being re-organized to different locations. To this end, we introduce the minimum correlation constraint which aims to minimize the following correlation within the spatial assembly weight map:

$$\begin{aligned}\mathbb{C}[w(u,v)] = &\sum_{u_1 \neq u_2} \sum_{v} w(u_1, v) \cdot w(u_2, v) \\ &+ \sum_{v_1 \neq v_2} \sum_{u} w(u, v_1) \cdot w(u, v_2).\end{aligned} \tag{5.6}$$

5.3.2 Spatial Assembly with Local Coherence

The above formulation of spatial assembly aims to achieve spatial re-assembly of the feature map in a differentiable manner. It should be noted that this spatial re-assembly operation is performed on feature vectors at individual spatial locations of the feature map, or individual feature points. Although the spatial assembly is learned by optimizing the target loss function, it is highly likely that feature points from the same object may be dis-assembled into different locations in the output feature map.

To address this issue, we propose introduce local coherence into the spatial assembly operation. While it could be more effective to develop a separate network to predict if two feature points belong to the same object or not, we choose to adopt a simple yet efficient measure to enforce the local coherence. Specifically, we define the local coherence $\alpha(i,j;i',j')$ as the correlation (cosine similarity) between feature vector $\boldsymbol{F}_{i,j} = [F(i,j,1), \cdots, F(i,j,C)]$ and its neighbor $\boldsymbol{F}_{i',j'} = [F(i',j',1), \cdots, F(i',j',C)]$, i.e.,

$$\alpha(i,j;i',j') = \begin{cases} \frac{\boldsymbol{F}_{i,j} \cdot \boldsymbol{F}_{i',j'}}{||\boldsymbol{F}_{i,j}|| \cdot ||\boldsymbol{F}_{i',j'}||}, & (i',j') \in \Omega_{i,j}, \\ 0, & \text{elsewhere.} \end{cases} \tag{5.7}$$

where $\Omega_{i,j}$ is the set of 8 direct neighbor points of (i,j). To address the computational complexity, the number of feature vectors can be limited. During coherent spatial assembly, we expect that neighboring feature points with high local coherence should be maintained together in the output feature map. In otherwise words, they should have similar spatial assemble weights. Motivated by this, we introduce the following loss function

$$\mathcal{L}_{LC} = \sum_{(i',j') \in \Omega_{i,j}} \alpha(i,j;i',j') \cdot ||\boldsymbol{W}_{i,j} - \mathcal{S}_{i',j'}^{i,j}[\boldsymbol{W}_{i',j'}]||_2, \tag{5.8}$$

where $\boldsymbol{W}_{i,j}$ represents the 2-D spatial assembly weight map of size $N_H \times N_W$ for point (i,j). $\mathcal{S}_{i,j}^{i',j'}[\cdot]$ performs a 2-D shift of the whole weight map by one point such that point (i',j') is aligned to point (i,j).

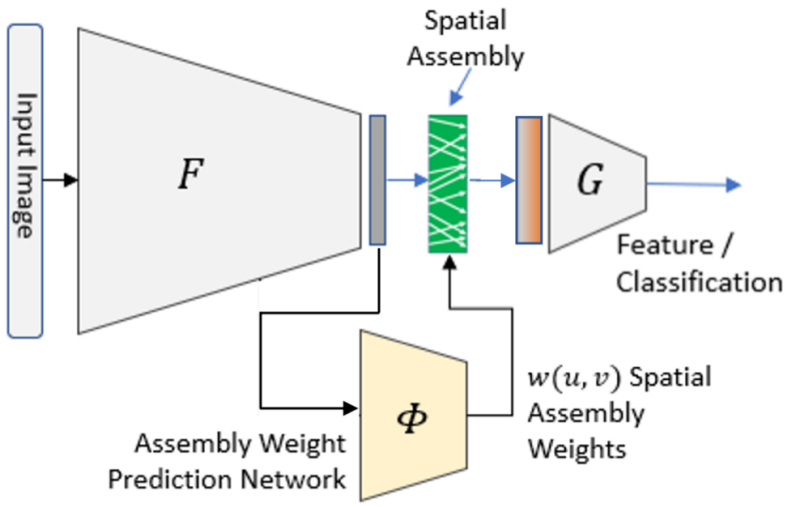


Figure 5.3: The spatial assembly networks being embedded into the deep neural network.

5.3.3 Spatial Assembly Networks

Figure 5.3 shows the design of the spatial assembly network and how it is embedded into existing deep neural networks for feature learning. The spatial assembly module is integrated into an intermediate layer of the network. The feature map $F(i, j, c)$ generated by network $\mathbb{F}$ is used as input to the spatial assembly network $\mathbf{\Phi}$, which predicts the spatial assembly weight map $w(u, v)$, as defined in the above section. Using this weight map, the input feature map $F(i, j, c)$ is re-assembled into a new feature map $G(n, m, c)$, which will be further processed by the upper network $\mathbb{G}$. In the following, we use the supervised metric learning as an example to explain the loss function design and learning process. This learning processing can be naturally extended to unsupervised feature learning and will be evaluated in our experiments.

In supervised metric or feature learning, the network aims to generate discriminative features such that intra-class image feature distance is minimized and the

inter-class feature distance is maximized. As illustrated in Figure 5.1, the proposed spatial assembly has the capability to handle spatial variations caused by changes in object poses, part configurations, spatial layout, and scene structures, and significantly reduce the intra-class feature variations. This can be driven by the metric loss defined at the network output. For example, in our experiments, our baseline system includes the multi-similarity (MS) loss [70]. The MS method computes the similarity scores between image samples in the current mini-batch. The similarity matrix between features of the current mini-batch $\mathbf{S}$. For each sample I_k, we determine the set of positive pairs $\mathcal{P}_k$ and the set of hard negative pairs $\mathcal{N}_k$ based on their similarity scores. $\mathbf{S}_{kp}$ and $\mathbf{S}_{kp}$ are similarity scores of the positive and negative pairs. We define the loss for all samples $\{I_k\}$ in the mini-batch as follows

$$\begin{aligned}\mathcal{L}_{FEN} = \frac{1}{N_B}\sum_{k=1}^{N_B}\{\frac{1}{\lambda_P}\log[1+\sum_{p\in\mathcal{P}_k}(e^{-\lambda_P(\mathbf{S}_{kp}-\delta)})] \\ + \frac{1}{\lambda_N}\log[1+\sum_{q\in\mathcal{N}_k}(e^{\lambda_N(\mathbf{S}_{kq}-\delta)})]\},\end{aligned} \tag{5.9}$$

where δ is a margin threshold, λ_P and λ_N are hyper-parameters for positive and negative pairs. We follow [70] for the setting of these hyper-parameters.

During training, the error gradients from metric loss will back propagated through network $\mathbb{G}$ to the spatial assembly layer, which will be further propagated to the spatial assembly network and the bottom network $\mathbb{F}$. According to (5.4), the gradients of the output feature map $G(n,m,c)$ with respect to the input feature map $F(i,j,c)$ and the spatial assembly weights are given by

$$\frac{\partial G(n,m,c)}{\partial F(i,j,c)} = w(i\times N+j,\ n\times N+m), \tag{5.10}$$

and

$$\begin{aligned} &\frac{\partial G(n,m,c)}{\partial w(u,v)} = F(i,j,c), \\ &u = i \times N + j, \quad v = n \times N + j. \end{aligned} \tag{5.11}$$

In addition to the error gradients back propagated from the network output, we also use the minimum entropy, minimum correlation constraints, and the local coherence penalty to regulate the training of the spatial assembly weight prediction network $\mathbf{\Phi}$ through the following combined loss

$$\mathcal{L}_{\mathbf{\Phi}} = \lambda_1 \cdot \mathbb{E}[w(u,v)] + \lambda_2 \cdot \mathbb{C}[w(u,v)] + \lambda_3 \cdot \mathcal{L}_{LC}, \tag{5.12}$$

as illustrated in Figure 5.3. λ_i are the weighting parameters. Once successfully trained, the SAN module will analyze the incoming feature map, predict the spatial assembly weight map. Figure 5.4 shows two examples of predicted spatial assembly weight map. In each example, we show the maximum weight of the first two rows from the weight map $w(u,v)$. It should be noted that we have re-organized 1-D weights of each row into an 2-D vector. Each 2-D vector represents the assembly weight vector for one output feature point. We only mark the maximum weight point in each 2-D vector by red for a better visualization. The whole weight map is used to re-assembly the feature map to generate the output feature map, which will be further analyzed by the network to produce the feature or decision.

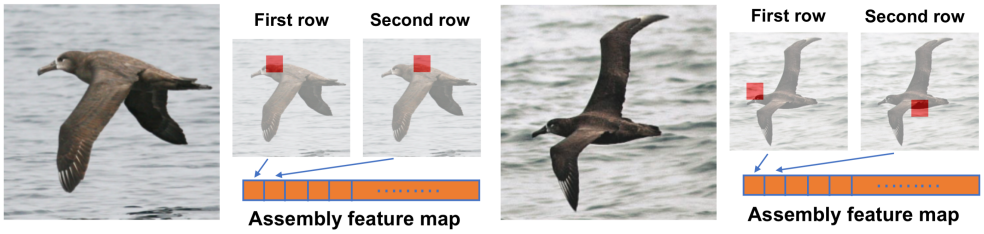


Figure 5.4: Examples of the first two rows in the predicted spatial assembly weight map.

5.4 Experimental Results

In this section, we conduct experiments mainly on the supervised metric learning and unsupervised metric learning to evaluate the performance of the spatial assembly network.

5.4.1 Datasets

For supervised and unsupervised deep metric learning, we use the following four benchmark datasets, following the same procedure used by existing papers [94, 79, 118, 119]. **(1) CUB-200-2011 (CUB)** [88] is a fine-grained bird dataset. It contains 11,788 images of birds from 200 categories. The first 100 classes are used for training, the remaining 100 classes are used for testing. **(2) Cars-196 (Cars)** [89] consists of 16,185 car model images (196 classes). We split the first 98 classes (8,054 images) for training, and remaining 98 classes (8,131 images) for testing. **(3) Stanford Online Products (SOP)** [75] consists of 120,053 online product images (22,634 classes) from Ebay. The first 11,318 classes are used for training and the remaining 11,316 classes are used for testing. **(4) In-Shop Clothes Retrieval (In-Shop)** [120] contains 54,642 images with 11,735 clothing classes. We use the predefined 25,882 training

images of 3,997 classes for training. The testing set includes 14,218 query images of 3,985 classes and 12,612 gallery images of 3,985 classes.

5.4.2 Supervised Metric Learning

We follow the recent state-of-the-art methods [70, 121, 94, 79] and conduct the experiments on the fine-grained CUB , Cars, SOP, and In-Shop datasets which are challenging for learning discriminative features. We utilize the GoogleNet network [90] pre-trained on ImageNet [72] as the backbone network with an one-layer embedding head to embed feature representation to the 512-dimensional feature space on all datasets for the benchmark performance comparison. We implement our algorithm with PyTorch. The Adam optimizer [93] is used in all experiments with $5e^{-4}$ weight decay. In the following experiments, we use the standard image retrieval performance metric (Recall@K), for performance evaluations and comparisons. Note that the major challenge here is that the training classes are totally different from the test classes.

The performance comparisons with existing state-of-the-art supervised metric learning methods on the CUB, Cars, and In-Shop datasets are summarized in Table 5.1. These methods include: LiftedStruct Loss [75], Histogram Loss [122], N-Pair Loss [92], Clustering [123], BIER (boosting independent embeddings robustly) [124], Angular Loss [125], MS (Multi-Similarity) Loss [70], HDML (hardness-aware deep metric learning) [121], ABIER [94] and XBM (Cross-Batch Memory) [79]. We use the multi-similarity loss [70] with momentum memory bank [27] as the baseline system. The momentum memory bank has a contrastive-based loss [126]. Our proposed method is the baseline system with SAN module. From Table 5.1, we can see that

Table 5.1: Recall@K (%) performance on the CUB and Cars, and In-Shop datasets with GoogleNet in comparison with other supervised metric learning methods. Some papers did not report results on specific datasets, which are marked with -.

Methods	CUB				Cars				In-Shop			
	R@1	R@2	R@4	R@8	R@1	R@2	R@4	R@8	R@1	R@10	R@20	R@30
LiftedStruct [75] CVPR16	47.2	58.9	70.2	80.2	49.0	60.3	72.1	81.5	-	-	-	-
Histogram Loss [122] NIPS16	50.3	61.9	72.6	82.4	-	-	-	-	-	-	-	-
N-Pair Loss [92] NIPS16	51.0	63.3	74.3	83.2	71.1	79.7	86.5	91.6	-	-	-	-
Clustering [123] CVPR17	48.2	61.4	71.8	81.9	58.1	70.6	80.3	87.8	-	-	-	-
BIER [124] ICCV17	55.3	67.2	76.9	85.1	78.0	85.8	91.1	95.1	76.9	92.8	95.2	96.2
Angular Loss [125] ICCV17	54.7	66.3	76.0	83.9	71.4	81.4	87.5	92.1	-	-	-	-
MS [70] CVPR19	58.2	69.8	79.9	87.3	75.7	84.6	90.1	94.4	85.1	96.7	97.8	98.3
HDML [121] CVPR19	53.7	65.7	76.7	85.7	79.1	87.1	92.1	95.5	-	-	-	-
A-BIER [94] TPAMI18	57.5	68.7	78.3	86.2	82.0	89.0	93.2	96.1	83.1	95.1	96.9	97.5
XBM [79] CVPR20	61.9	72.9	81.2	88.6	80.3	87.1	91.9	95.1	89.1	97.3	98.1	98.4
Proposed	**63.3**	**74.5**	**83.8**	**90.4**	**83.5**	**89.7**	**93.4**	**96.1**	**92.5**(88.5)	**98.9**(97.5)	**99.3**(98.2)	**99.5**(98.6)
Gain	**+1.4**	**+1.6**	**+2.6**	**+1.8**	**+1.5**	**+0.7**	**+0.2**	**+0.0**	**+3.4**(-)	**+1.6**(0.2)	**+1.2**(0.1)	**+1.1**(0.2)

our method outperforms the state-of-the-art methods by up to 2.6% on the Recall@1, 2, 4, and 8 rates on the CUB dataset. We evaluate the performance of the In-Shop dataset in two settings. One setting uses the whole testing set as the query set and gallery set, the other setting splits the testing set into query set (14,218 query images) and gallery set (12,612 gallery images). The performance of the second setting shows in the brackets.

5.4.3 Unsupervised Deep Metric Learning

In the following experiments, we evaluate the performance of the spatial assembly network for unsupervised metric learning where image labels are not available. We compare the performance of our proposed methods with the state-of-the-art unsupervised methods: MOM (mining on manifolds) [80], AND (anchor neighborhood discovery) [127], CBSwR (center-based softmax with reconstruction) [128], PSLR (probabilistic structural latent representation) [118], ISIF [71], and aISIF [119] (augmentation

invariant and spreading instance feature). For fair comparisons, the authors of the aISIF paper [119] have implemented three other state-of-the-art methods developed for feature learning and adapted them to unsupervised deep metric learning tasks: Examplar [96], NCE (Noise-Contrastive Estimation) [97], and DeepCluster [21], which are included into our comparison. We use the same baseline system in the supervised metric learning.

We use the ImageNet [72] pre-trained GoogleNet [90] as the backbone network and set the embedding feature dimension to 128 on CUB, Cars, and SOP datasets for performance comparison. We use the k-means clustering to cluster the embedding features of training samples and assign pseudo labels to them. We set the cluster number K to be 100 for the CUB and Cars datasets, and set K to 10,000 for the SOP dataset. From Table 5.2, we can see that our proposed method outperforms the state-of-the-art unsupervised methods by a large margin.

Following the recent state-of-the-art PSLR [118], ISIF [71], and aISIF [119] methods, we also evaluate the performance of our proposed method on Resnet-18 without pre-trained parameters. In this experiment, we use the randomly initialized Resnet-18 network with an one-layer embedding head to verify the effectiveness of our proposed SAN module. We set the feature embedding dimension to 128 and conduct experiments on the large-scale SOP dataset. The results in Table 5.3 show that our proposed method has improved the Recall@1, Recall@10, and Recall@100 rates by 4.0%, 4.2%, and 4.5%, respectively.

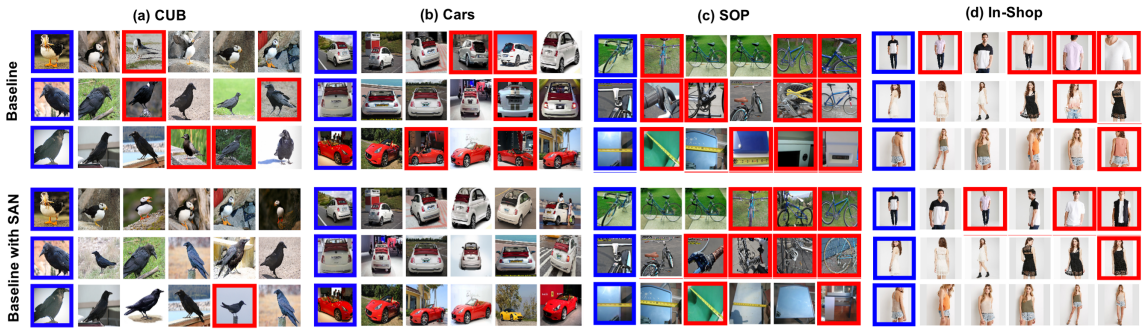


Figure 5.5: Retrieval examples by the baseline with and without our SAN module on the CUB, Cars, SOP, and In-Shop datasets. The query images and the incorrect retrieved images are highlighted with *blue* and *red*.

Figure 5.6: Retrieval examples by the baseline with our SAN module on the CUB, Cars, and SOP datasets from unsupervised metric leaning. The query images and the incorrect retrieved images are highlighted with *blue* and *red*.

Table 5.2: Recall@K (%) performance on the CUB, Cars, and SOP datasets with GoogleNet in comparison with other unsupervised metric learning methods.

Methods	CUB				Cars				SOP		
	R@1	R@2	R@4	R@8	R@1	R@2	R@4	R@8	R@1	R@10	R@100
Examplar [96] *TPAMI16*	38.2	50.3	62.8	75.0	36.5	48.1	59.2	71.0	45.0	60.3	75.2
NCE [97] CVPR18	39.2	51.4	63.7	75.8	37.5	48.7	59.8	71.5	46.6	62.3	76.8
DeepCluster [21] ECCV18	42.9	54.1	65.6	76.2	32.6	43.8	57.0	69.5	34.6	52.6	66.8
MOM [80] CVPR18	45.3	57.8	68.6	78.4	35.5	48.2	60.6	72.4	43.3	57.2	73.2
AND [127] ICML19	47.3	59.4	71.0	80.0	38.4	49.6	60.2	72.9	47.4	62.6	77.1
ISIF [71] CVPR19	46.2	59.0	70.1	80.2	41.3	52.3	63.6	74.9	48.9	64.0	78.0
aISIF [119] TPAMI20	47.7	59.9	71.2	81.4	41.2	52.6	63.8	75.1	49.7	65.4	79.5
CBSwR [128] BMVC20	47.5	59.6	70.6	80.5	42.6	54.4	65.4	76.0	-	-	-
PSLR [118] CVPR20	48.1	60.1	71.8	81.6	43.7	54.8	66.1	76.2	51.1	66.5	79.8
Proposed	**55.9**	**68.0**	**78.6**	**86.8**	**44.2**	**55.5**	**66.8**	**76.9**	**58.7**	**73.1**	**84.6**
Gain	**+7.8**	**+7.9**	**+6.2**	**+5.2**	**+0.5**	**+0.7**	**+0.7**	**+0.7**	**+7.6**	**+6.6**	**+4.8**

5.4.4 Ablation Studies

In the following, we perform ablation studies to further understand the performance of the proposed spatial assembly network.

(1) Performance contribution of the SAN module. In this ablation study, we aim to identify the contribution of our proposed SAN module on different datasets. Table 5.4 summarizes the performance results on the CUB and SOP datasets with and without using the SAN module in both supervised and unsupervised deep metric learning. The baseline system is using the multi-similarity [70] with momentum memory bank [27]. The momentum memory bank has a contrastive-based loss [126]. We can see that our proposed SAN module significantly improves the performance by a large margin. Figure 5.5 shows the retrieval examples by the baseline with and without our SAN module on the CUB, Cars, SOP, and In-Shop datasets from supervised metric learning. The top row shows the retrieval results by the baseline, and the bottom row shows the results for the baseline plus the SAN module. Samples

Table 5.3: Recall@K (%) performance on the SOP dataset using Resnet-18 network without pre-trained parameters.

Methods	SOP		
	R@1	R@10	R@100
Random	18.4	29.4	46.0
Examplar [96] TPAMI16	31.5	46.7	64.2
NCE [97] CVPR18	34.4	49.0	65.2
MOM [80] CVPR18	16.3	27.6	44.5
AND [127] ICML19	36.4	52.8	67.2
ISIF [71] CVPR19	39.7	54.9	71.0
aISIF [119] TPAMI20	40.7	55.9	72.2
PSLR [118] CVPR20	42.3	57.7	72.5
Proposed	**46.3**	**61.9**	**77.0**
Gain	**+4.0**	**+4.2**	**+4.5**

highlighted with blue and red boxes are query images and incorrect retrieval results. We can see that, using the SAN module, the number of incorrect retrieval results have been significantly reduced because the learned feature is much more discriminative. Figure 5.6 shows the retrieval examples by the baseline with our SAN module on the CUB, Cars, and SOP datasets from unsupervised metric learning. We can see that our SAN can also learn discriminitive features, even image labels are not available.

(2) Performance of SAN module with different metric learning losses. In order to verify the generalization capability of our method, we conduct experiments to show the performance of our proposed SAN module with different metric learning losses and different backbone networks. It should be noted that the momentum memory bank [27, 126] in the baseline system is not included in this experiment. We evaluate the MS loss [70] with SAN on GoogleNet backbone and Proxy-Anchor [129] loss with SAN on BN-inception backbone. From the Table 5.5, we can see that the MS loss [70] with SAN has improved the Recall@1 rate by 1.5% and the Proxy-Anchor

Table 5.4: The Recall@K performance of the baseline and baseline with our proposed SAN module on the CUB and SOP datasets.

Supervised Metric Learning				
Methods	**CUB**			
	R@1	R@2	R@4	R@8
Baseline	61.8	73.2	82.3	88.9
+ SAN	**63.3**	**74.5**	**83.8**	**90.4**
Gain	**+1.5**	**+1.3**	**+1.5**	**+1.5**
Methods	**SOP**			
	R@1	R@10	R@100	R@1000
Baseline	73.7	87.9	95.0	98.4
+ SAN	**75.8**	**89.2**	**95.5**	**98.6**
Gain	**+2.1**	**+1.3**	**+0.5**	**+0.2**
Unsupervised Metric Learning				
Methods	**CUB**			
	R@1	R@2	R@4	R@8
Baseline	53.3	66.1	77.4	85.6
+ SAN	**55.9**	**68.0**	**78.6**	**86.8**
Gain	**+2.6**	**+1.9**	**+1.2**	**+1.2**
Methods	**SOP**			
	R@1	R@10	R@100	
Baseline	56.9	71.2	82.7	
+ SAN	**58.7**	**73.1**	**84.6**	
Gain	**+1.8**	**+1.9**	**+1.9**	

Table 5.5: Recall@K (%) performance on SAN with Multi-Similarity (MS) loss and Proxy-Anchor loss on the CUB dataset. 'G' denotes GoogleNet, 'BN' denotes BN-inception.

Methods		**CUB**			
		R@1	R@2	R@4	R@8
MS [129] CVPR19	G	58.2	69.8	79.9	87.3
MS with SAN	G	**59.7**	**72.0**	**81.4**	**88.4**
Gain		**+1.5**	**+2.2**	**+1.5**	**+1.1**
Proxy-Anchor [129] CVPR20	BN	68.4	79.2	**86.8**	91.6
Proxy-Anchor with SAN	BN	**69.5**	**79.3**	86.7	**92.0**
Gain		**+1.1**	**+0.1**	**-**	**+0.4**

[129] with SAN has improved the Recall@1 rate by 1.1%.

5.5 Conclusion

In this work, we have successfully developed a new spatial assembly network to explore the spatial variations caused by changes in object part configurations, spatial layout of object, and scene structures of the images. This SAN module examines the input image and perform a learned re-organization and assembly of feature points from different spatial locations conditioned by feature maps from previous network layers so as to maximize the discriminative power of the final feature representation. The proposed spatial assembly goes beyond spatial image transforms. It learns to re-organize or re-assemble the feature maps across different spatial locations. This work represents one of the first efforts to explore spatial reorganization of feature maps for 2D images. The proposed spatial assembly represents a more generic feature operation than simple permutation. This differentiable module can be directly incorporated into existing deep neural network for end-to-end training to increase network robustness under spatial variations and improve the discriminative power of image features In our experiments, we have demonstrated that the proposed SAN module is able to significantly improve the performance of various metric / representation learning tasks, in both supervised and unsupervised learning scenarios.

Chapter 6

Summary and Concluding Remarks

To achieve the desired performance in many computer vision applications, deep neural networks often need a tremendously large set of labeled training samples. Labeling a large dataset is labor-intensive, time-consuming, and sometimes requiring expert knowledge. In this dissertation, we study the topic: deep learning with very few and no labels. The proposed methods mainly follow these two topics: semi-supervised learning and unsupervised learning.

In **Chapter 2**, we develop a joint sample discovery and iterative model evolution method for semi-supervised learning on small labeled training sets. We propose a master-teacher-student model framework to provide multi-layer guidance during the model evolution process with multiple iterations and generations. The master network combines the knowledge of the student and teacher models with additional access to newly discovered samples. The master and teacher models are then used to guide the training of the student network by enforcing the consistency between their predictions of unlabeled samples and evolve all models when more and more samples

are discovered.

In **Chapter 3**, we propose to push the performance limit of semi-supervised learning on very small sets of labeled samples by developing a new method called *learned model composition with critical sample look-ahead* (LMCS). Specifically, our proposed LMCS method explores two major ideas. First, it introduces a new learned model composition structure so that we can compose a more efficient master network from student models of past iterations through a network learning process. Second, we develop a new method, called *confined maximum entropy search*, to discover new critical samples near the model decision boundary and provide the master model with look-ahead access to these samples to enhance its guidance capability.

In **Chapter 4**, we develop a new approach to unsupervised deep metric learning. To characterize the consistent pattern of human attention during image comparisons, we introduce the idea of transformed attention consistency. It assumes that visually similar images, even undergoing different image transforms, should share the same consistent visual attention map. This consistency leads to a pairwise self-supervision loss, allowing us to learn a Siamese deep neural network to encode and compare images against their transformed or matched pairs. To further enhance the inter-class discriminative power of the feature generated by this network, we adapt the concept of triplet loss from supervised metric learning to our unsupervised case and introduce the contrastive clustering loss.

In **Chapter 5**, we introduce a new learnable module for supervised and unsupervised representation learning, called spatial assembly network (SAN). This SAN module examines the input image and performs a learned re-organization and assembly of feature points from different spatial locations conditioned by feature maps

from previous network layers so as to maximize the discriminative power of the final feature representation. This differentiable module improving their capabilities in handling spatial variations and structural changes of the image scene.

Bibliography

[1] Chris M Bishop. Training with noise is equivalent to tikhonov regularization. *Neural computation*, 7(1):108–116, 1995.

[2] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

[3] Xiaojin Zhu, Zoubin Ghahramani, and John D Lafferty. Semi-supervised learning using gaussian fields and harmonic functions. In *Proceedings of the 20th International conference on Machine learning (ICML-03)*, pages 912–919, 2003.

[4] Avrim Blum, John Lafferty, Mugizi Robert Rwebangira, and Rajashekar Reddy. Semi-supervised learning using randomized mincuts. In *Proceedings of the twenty-first international conference on machine learning*, page 13, 2004.

[5] Jingsong Xu, Qiang Wu, Jian Zhang, Fumin Shen, and Zhenmin Tang. Boosting separability in semisupervised learning for object classification. *IEEE transactions on circuits and systems for video technology*, 24(7):1197–1208, 2014.

[6] Meng Jian and Cheolkon Jung. Semi-supervised bi-dictionary learning for image classification with smooth representation-based label propagation. *IEEE Transactions on Multimedia*, 18(3):458–473, 2016.

[7] Suo Qiu, Feiping Nie, Xiangmin Xu, Chunmei Qing, and Dong Xu. Accelerating flexible manifold embedding for scalable semi-supervised learning. *IEEE Transactions on Circuits and Systems for Video Technology*, 29(9):2786–2795, 2018.

[8] Durk P Kingma, Shakir Mohamed, Danilo Jimenez Rezende, and Max Welling. Semi-supervised learning with deep generative models. In *Advances in neural information processing systems*, pages 3581–3589, 2014.

[9] Philip Bachman, Ouais Alsharif, and Doina Precup. Learning with pseudo-ensembles. In *Advances in Neural Information Processing Systems*, pages 3365–3373, 2014.

[10] Jost Tobias Springenberg. Unsupervised and semi-supervised learning with categorical generative adversarial networks. *4th International Conference on Learning Representations*, 2015.

[11] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. In *Advances in neural information processing systems*, pages 2234–2242, 2016.

[12] Avital Oliver, Augustus Odena, Colin A Raffel, Ekin Dogus Cubuk, and Ian Goodfellow. Realistic evaluation of deep semi-supervised learning algorithms. In *Advances in Neural Information Processing Systems*, pages 3239–3250, 2018.

[13] Yves Grandvalet and Yoshua Bengio. Semi-supervised learning by entropy minimization. In *Advances in neural information processing systems*, pages 529–536, 2005.

[14] Dong-Hyun Lee. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In *Workshop on Challenges in Representation Learning, ICML*, volume 3, page 2, 2013.

[15] Samuli Laine and Timo Aila. Temporal ensembling for semi-supervised learning. In *Fifth International Conference on Learning Representations*, 2017.

[16] Antti Tarvainen and Harri Valpola. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. In *Advances in neural information processing systems*, pages 1195–1204, 2017.

[17] Zhanghan Ke, Daoye Wang, Qiong Yan, Jimmy Ren, and Rynson WH Lau. Dual student: Breaking the limits of the teacher in semi-supervised learning. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 6728–6736, 2019.

[18] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*, 2017.

[19] David Berthelot, Nicholas Carlini, Ian Goodfellow, Nicolas Papernot, Avital Oliver, and Colin A Raffel. Mixmatch: A holistic approach to semi-supervised

learning. In *Advances in Neural Information Processing Systems*, pages 5050–5060, 2019.

[20] Mehdi Sajjadi, Mehran Javanmardi, and Tolga Tasdizen. Regularization with stochastic transformations and perturbations for deep semi-supervised learning. In *Advances in Neural Information Processing Systems*, pages 1163–1171, 2016.

[21] Mathilde Caron, Piotr Bojanowski, Armand Joulin, and Matthijs Douze. Deep clustering for unsupervised learning of visual features. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 132–149, 2018.

[22] Liheng Zhang, Guo-Jun Qi, Liqiang Wang, and Jiebo Luo. Aet vs. aed: Unsupervised representation learning by auto-encoding transformations rather than data. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2547–2555, 2019.

[23] Carl Doersch, Abhinav Gupta, and Alexei A Efros. Unsupervised visual representation learning by context prediction. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1422–1430, 2015.

[24] Mehdi Noroozi and Paolo Favaro. Unsupervised learning of visual representations by solving jigsaw puzzles. In *European Conference on Computer Vision*, pages 69–84. Springer, 2016.

[25] Terrance DeVries and Graham W Taylor. Improved regularization of convolutional neural networks with cutout. *arXiv preprint arXiv:1708.04552*, 2017.

[26] Spyros Gidaris, Praveer Singh, and Nikos Komodakis. Unsupervised representation learning by predicting image rotations. *arXiv preprint arXiv:1803.07728*, 2018.

[27] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. *arXiv preprint arXiv:1911.05722*, 2019.

[28] Paul Wohlhart and Vincent Lepetit. Learning descriptors for object recognition and 3d pose estimation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3109–3118, 2015.

[29] Xinwei He, Yang Zhou, Zhichao Zhou, Song Bai, and Xiang Bai. Triplet-center loss for multi-view 3d object retrieval. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1945–1954, 2018.

[30] Alexander Grabner, Peter M Roth, and Vincent Lepetit. 3d pose estimation and 3d model retrieval for objects in the wild. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3022–3031, 2018.

[31] Yandong Wen, Kaipeng Zhang, Zhifeng Li, and Yu Qiao. A discriminative feature learning approach for deep face recognition. In *European conference on computer vision*, pages 499–515. Springer, 2016.

[32] Rui Yu, Zhiyong Dou, Song Bai, Zhaoxiang Zhang, Yongchao Xu, and Xiang Bai. Hard-aware point-to-set deep metric for person re-identification. In

Proceedings of the European Conference on Computer Vision (ECCV), pages 188–204, 2018.

[33] Alexander Hermans, Lucas Beyer, and Bastian Leibe. In defense of the triplet loss for person re-identification. *arXiv preprint arXiv:1703.07737*, 2017.

[34] Takeru Miyato, Shin-ichi Maeda, Shin Ishii, and Masanori Koyama. Virtual adversarial training: a regularization method for supervised and semi-supervised learning. *IEEE transactions on pattern analysis and machine intelligence*, 2018.

[35] Mikhail Belkin, Partha Niyogi, and Vikas Sindhwani. Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. *Journal of machine learning research*, 7(Nov):2399–2434, 2006.

[36] David Yarowsky. Unsupervised word sense disambiguation rivaling supervised methods. In *33rd annual meeting of the association for computational linguistics*, 1995.

[37] Ellen Riloff and Janyce Wiebe. Learning extraction patterns for subjective expressions. In *Proceedings of the 2003 conference on Empirical methods in natural language processing*, 2003.

[38] Chuck Rosenberg, Martial Hebert, and Henry Schneiderman. Semi-supervised self-training of object detection models. 2005.

[39] Jianqing Liang, Qinghua Hu, Wenwu Wang, and Yahong Han. Semisupervised online multikernel similarity learning for image retrieval. *IEEE Transactions on Multimedia*, 19(5):1077–1089, 2016.

[40] Xiaojin Jerry Zhu. Semi-supervised learning literature survey. Technical report, University of Wisconsin-Madison Department of Computer Sciences, 2005.

[41] Leo Breiman. Bagging predictors. *Machine learning*, 24(2):123–140, 1996.

[42] Avrim Blum and Tom Mitchell. Combining labeled and unlabeled data with co-training. In *Proceedings of the eleventh annual conference on Computational learning theory*, pages 92–100. ACM, 1998.

[43] Vincent-Wayne Mitchell. Consumer perceived risk: conceptualisations and models. *European Journal of marketing*, 33(1/2):163–195, 1999.

[44] Siyuan Qiao, Wei Shen, Zhishuai Zhang, Bo Wang, and Alan Yuille. Deep co-training for semi-supervised image recognition. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 135–152, 2018.

[45] Avrim Blum and Shuchi Chawla. Learning from labeled and unlabeled data using graph mincuts. 2001.

[46] Xiaojin Zhu and Zoubin Ghahramani. Learning from labeled and unlabeled data with label propagation. Technical report, Citeseer, 2002.

[47] Chang Tang, Xinwang Liu, Pichao Wang, Changqing Zhang, Miaomiao Li, and Lizhe Wang. Adaptive hypergraph embedded semi-supervised multi-label image annotation. *IEEE Transactions on Multimedia*, 2019.

[48] Max Whitney and Anoop Sarkar. Bootstrapping via graph propagation. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*, pages 620–628. Association for Computational Linguistics, 2012.

[49] Si Wu, Qiujia Ji, Shufeng Wang, Hau-San Wong, Zhiwen Yu, and Yong Xu. Semi-supervised image classification with self-paced cross-task networks. *IEEE Transactions on Multimedia*, 20(4):851–865, 2017.

[50] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

[51] Olivier Chapelle and Alexander Zien. Semi-supervised classification by low density separation. In *AISTATS*, volume 2005, pages 57–64. Citeseer, 2005.

[52] Isaac Triguero, Salvador García, and Francisco Herrera. Self-labeled techniques for semi-supervised learning: taxonomy, software and empirical study. *Knowledge and Information systems*, 42(2):245–284, 2015.

[53] Elad Hoffer and Nir Ailon. Semi-supervised deep learning by metric embedding. *arXiv preprint arXiv:1611.01449*, 2016.

[54] Antti Rasmus, Mathias Berglund, Mikko Honkala, Harri Valpola, and Tapani Raiko. Semi-supervised learning with ladder networks. In *Advances in neural information processing systems*, pages 3546–3554, 2015.

[55] Jichang Li, Si Wu, Cheng Liu, Zhiwen Yu, and Hau-San Wong. Semi-supervised deep coupled ensemble learning with classification landmark exploration. *IEEE Transactions on Image Processing*, 29:538–550, 2019.

[56] Ben Athiwaratkun, Marc Finzi, Pavel Izmailov, and Andrew Gordon Wilson. There are many consistent explanations of unlabeled data: Why you should average. *arXiv preprint arXiv:1806.05594*, 2018.

[57] Yucen Luo, Jun Zhu, Mengxi Li, Yong Ren, and Bo Zhang. Smooth neighbors on teacher graphs for semi-supervised learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8896–8905, 2018.

[58] Si Wu, Jichang Li, Cheng Liu, Zhiwen Yu, and Hau-San Wong. Mutual learning of complementary networks via residual correction for improving semi-supervised classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6500–6509, 2019.

[59] Vikas Verma, Alex Lamb, Juho Kannala, Yoshua Bengio, and David Lopez-Paz. Interpolation consistency training for semi-supervised learning. *arXiv preprint arXiv:1903.03825*, 2019.

[60] Olivier Chapelle, Bernhard Scholkopf, and Alexander Zien. Semi-supervised learning (chapelle, o. et al., eds.; 2006)[book reviews]. *IEEE Transactions on Neural Networks*, 20(3):542–542, 2009.

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

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

[63] Robert Dupre, Jiri Fajtl, Vasileios Argyriou, and Paolo Remagnino. Improving dataset volumes and model accuracy with semi-supervised iterative self-learning. *IEEE Transactions on Image Processing*, 2019.

[64] Ellen Riloff, Janyce Wiebe, and William Phillips. Exploiting subjectivity classification to improve information extraction. In *AAAI*, pages 1106–1111, 2005.

[65] Jian Zhang and Yuxin Peng. Ssdh: semi-supervised deep hashing for large scale image retrieval. *IEEE Transactions on Circuits and Systems for Video Technology*, 29(1):212–225, 2017.

[66] Ziliang Chen, Keze Wang, Xiao Wang, Pai Peng, Ebroul Izquierdo, and Liang Lin. Deep co-space: Sample mining across feature transformation for semi-supervised learning. *IEEE Transactions on Circuits and Systems for Video Technology*, 28(10):2667–2678, 2017.

[67] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. Technical report, Citeseer, 2009.

[68] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading digits in natural images with unsupervised feature learning. 2011.

[69] Michael S Gazzaniga. *The cognitive neurosciences*. MIT Press, 2009.

[70] Xun Wang, Xintong Han, Weilin Huang, Dengke Dong, and Matthew R Scott. Multi-similarity loss with general pair weighting for deep metric learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5022–5030, 2019.

[71] Mang Ye, Xu Zhang, Pong C Yuen, and Shih-Fu Chang. Unsupervised embedding learning via invariant and spreading instance feature. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6210–6219, 2019.

[72] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252, 2015.

[73] Sumit Chopra, Raia Hadsell, and Yann LeCun. Learning a similarity metric discriminatively, with application to face verification. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 1, pages 539–546. IEEE, 2005.

[74] Raia Hadsell, Sumit Chopra, and Yann LeCun. Dimensionality reduction by learning an invariant mapping. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 2, pages 1735–1742. IEEE, 2006.

[75] Hyun Oh Song, Yu Xiang, Stefanie Jegelka, and Silvio Savarese. Deep metric learning via lifted structured feature embedding. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4004–4012, 2016.

[76] VB Kumar, Ben Harwood, Gustavo Carneiro, Ian Reid, and Tom Drummond. Smart mining for deep metric learning. *arXiv preprint arXiv:1704.01285*, 2, 2017.

[77] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 815–823, 2015.

[78] Chao-Yuan Wu, R Manmatha, Alexander J Smola, and Philipp Krahenbuhl. Sampling matters in deep embedding learning. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2840–2848, 2017.

[79] Xun Wang, Haozhi Zhang, Weilin Huang, and Matthew R Scott. Cross-batch memory for embedding learning. *arXiv preprint arXiv:1912.06798*, 2019.

[80] Ahmet Iscen, Giorgos Tolias, Yannis Avrithis, and Ondřej Chum. Mining on manifolds: Metric learning without labels. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7642–7651, 2018.

[81] Jimmy Ba, Volodymyr Mnih, and Koray Kavukcuoglu. Multiple object recognition with visual attention. *arXiv preprint arXiv:1412.7755*, 2014.

[82] Volodymyr Mnih, Nicolas Heess, Alex Graves, et al. Recurrent models of visual attention. In *Advances in neural information processing systems*, pages 2204–2212, 2014.

[83] Max Jaderberg, Karen Simonyan, Andrew Zisserman, et al. Spatial transformer networks. In *Advances in neural information processing systems*, pages 2017–2025, 2015.

[84] Sanghyun Woo, Jongchan Park, Joon-Young Lee, and In So Kweon. Cbam: Convolutional block attention module. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 3–19, 2018.

[85] Jianlong Fu, Heliang Zheng, and Tao Mei. Look closer to see better: Recurrent attention convolutional neural network for fine-grained image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4438–4446, 2017.

[86] Binghui Chen and Weihong Deng. Hybrid-attention based decoupled metric learning for zero-shot image retrieval. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2750–2759, 2019.

[87] Jean-Michel Morel and Guoshen Yu. Asift: A new framework for fully affine invariant image comparison. *SIAM journal on imaging sciences*, 2(2):438–469, 2009.

[88] Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. The caltech-ucsd birds-200-2011 dataset. 2011.

[89] Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3d object representations for fine-grained categorization. In *4th International IEEE Workshop on 3D Representation and Recognition (3dRR-13)*, Sydney, Australia, 2013.

[90] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015.

[91] Yair Movshovitz-Attias, Alexander Toshev, Thomas K Leung, Sergey Ioffe, and Saurabh Singh. No fuss distance metric learning using proxies. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 360–368, 2017.

[92] Kihyuk Sohn. Improved deep metric learning with multi-class n-pair loss objective. In *Advances in neural information processing systems*, pages 1857–1865, 2016.

[93] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[94] Michael Opitz, Georg Waltner, Horst Possegger, and Horst Bischof. Deep metric learning with bier: Boosting independent embeddings robustly. *IEEE transactions on pattern analysis and machine intelligence*, 2018.

[95] Wonsik Kim, Bhavya Goyal, Kunal Chawla, Jungmin Lee, and Keunjoo Kwon. Attention-based ensemble for deep metric learning. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 736–751, 2018.

[96] Alexey Dosovitskiy, Philipp Fischer, Jost Tobias Springenberg, Martin Riedmiller, and Thomas Brox. Discriminative unsupervised feature learning with exemplar convolutional neural networks. *IEEE transactions on pattern analysis and machine intelligence*, 38(9):1734–1747, 2015.

[97] Zhirong Wu, Yuanjun Xiong, Stella X Yu, and Dahua Lin. Unsupervised feature learning via non-parametric instance discrimination. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3733–3742, 2018.

[98] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

[99] Alhussein Fawzi and Pascal Frossard. Manitest: Are classifiers really invariant? *arXiv preprint arXiv:1507.06535*, 2015.

[100] Jiancheng Lyu, Shuai Zhang, Yingyong Qi, and Jack Xin. Autoshufflenet: Learning permutation matrices via an exact lipschitz continuous penalty in deep convolutional neural networks. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 608–616, 2020.

[101] Karel Lenc and Andrea Vedaldi. Understanding image representations by measuring their equivariance and equivalence. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 991–999, 2015.

[102] Dmitry Laptev, Nikolay Savinov, Joachim M Buhmann, and Marc Pollefeys. Ti-pooling: transformation-invariant pooling for feature learning in convolutional neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 289–297, 2016.

[103] Osman Semih Kayhan and Jan C van Gemert. On translation invariance in cnns: Convolutional layers can exploit absolute spatial location. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14274–14285, 2020.

[104] Ekin D Cubuk, Barret Zoph, Dandelion Mane, Vijay Vasudevan, and Quoc V Le. Autoaugment: Learning augmentation strategies from data. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 113–123, 2019.

[105] Daniel Ho, Eric Liang, Xi Chen, Ion Stoica, and Pieter Abbeel. Population based augmentation: Efficient learning of augmentation policy schedules. In *International Conference on Machine Learning*, pages 2731–2741. PMLR, 2019.

[106] Can Kanbak, Seyed-Mohsen Moosavi-Dezfooli, and Pascal Frossard. Geometric robustness of deep networks: analysis and improvement. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4441–4449, 2018.

[107] Xu Shen, Xinmei Tian, Anfeng He, Shaoyan Sun, and Dacheng Tao. Transform-invariant convolutional neural networks for image classification and search. In *Proceedings of the 24th ACM international conference on Multimedia*, pages 1345–1354, 2016.

[108] Taco S Cohen and Max Welling. Transformation properties of learned visual representations. *arXiv preprint arXiv:1412.7659*, 2014.

[109] Kwang Moo Yi, Eduard Trulls, Vincent Lepetit, and Pascal Fua. Lift: Learned invariant feature transform. In *European Conference on Computer Vision*, pages 467–483. Springer, 2016.

[110] Jifeng Dai, Haozhi Qi, Yuwen Xiong, Yi Li, Guodong Zhang, Han Hu, and Yichen Wei. Deformable convolutional networks. In *Proceedings of the IEEE international conference on computer vision*, pages 764–773, 2017.

[111] Robert Gens and Pedro M Domingos. Deep symmetry networks. In *Advances in neural information processing systems*, pages 2537–2545, 2014.

[112] Kihyuk Sohn and Honglak Lee. Learning invariant representations with local transformations. *arXiv preprint arXiv:1206.6418*, 2012.

[113] Zhongdao Wang, Luming Tang, Xihui Liu, Zhuliang Yao, Shuai Yi, Jing Shao, Junjie Yan, Shengjin Wang, Hongsheng Li, and Xiaogang Wang. Orientation invariant feature embedding and spatial temporal regularization for vehicle re-identification. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 379–387, 2017.

[114] Clement Farabet, Camille Couprie, Laurent Najman, and Yann LeCun. Learning hierarchical features for scene labeling. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1915–1929, 2012.

[115] RE Burkard and E Cela. The quadratic assignment problem, in "handbook of combinatorial optimization" vol. 3. edited by dz du, pm pardalos, 1999.

[116] Joshua T Vogelstein, John M Conroy, Vince Lyzinski, Louis J Podrazik, Steven G Kratzer, Eric T Harley, Donniell E Fishkind, R Jacob Vogelstein, and Carey E Priebe. Fast approximate quadratic programming for graph matching. *PLOS one*, 10(4):e0121002, 2015.

[117] Xiangyu Zhang, Xinyu Zhou, Mengxiao Lin, and Jian Sun. Shufflenet: An extremely efficient convolutional neural network for mobile devices. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6848–6856, 2018.

[118] Mang Ye and Jianbing Shen. Probabilistic structural latent representation for unsupervised embedding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5457–5466, 2020.

[119] Mang Ye, Jianbing Shen, Xu Zhang, Pong C Yuen, and Shih-Fu Chang. Augmentation invariant and instance spreading feature for softmax embedding. *IEEE transactions on pattern analysis and machine intelligence*, 2020.

[120] Ziwei Liu, Ping Luo, Shi Qiu, Xiaogang Wang, and Xiaoou Tang. Deepfashion: Powering robust clothes recognition and retrieval with rich annotations. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1096–1104, 2016.

[121] Wenzhao Zheng, Zhaodong Chen, Jiwen Lu, and Jie Zhou. Hardness-aware deep metric learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 72–81, 2019.

[122] Evgeniya Ustinova and Victor Lempitsky. Learning deep embeddings with histogram loss. In *Advances in Neural Information Processing Systems*, pages 4170–4178, 2016.

[123] Hyun Oh Song, Stefanie Jegelka, Vivek Rathod, and Kevin Murphy. Deep metric learning via facility location. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5382–5390, 2017.

[124] Michael Opitz, Georg Waltner, Horst Possegger, and Horst Bischof. Bier-boosting independent embeddings robustly. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5189–5198, 2017.

[125] Jian Wang, Feng Zhou, Shilei Wen, Xiao Liu, and Yuanqing Lin. Deep metric learning with angular loss. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2593–2601, 2017.

[126] Shichao Kan, Yigang Cen, Yang Li, Mladenovic Vladimir, , and Zhihai He. Contrastive bayesian analysis for supervised deep metric learning. In *Github*, 2020.

[127] Jiabo Huang, Qi Dong, Shaogang Gong, and Xiatian Zhu. Unsupervised deep learning by neighbourhood discovery. In *International Conference on Machine Learning*, pages 2849–2858, 2019.

[128] Binh X Nguyen, Binh D Nguyen, Gustavo Carneiro, Erman Tjiputra, Quang D Tran, and Thanh-Toan Do. Deep metric learning meets deep clustering: An novel unsupervised approach for feature embedding. *arXiv preprint arXiv:2009.04091*, 2020.

[129] Sungyeon Kim, Dongwon Kim, Minsu Cho, and Suha Kwak. Proxy anchor loss for deep metric learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3238–3247, 2020.

VITA

Yang Li was born in Zibo City, Shandong Province, China. He started Master of Science in Electrical and Computer Engineering (ECE) at University of Missouri in 2015, and transferred to Ph.D. program in ECE at University of Missouri in 2017. He joined Video Processing and Communication Lab and started his research under the guidance of Dr. Zhihai He in 2016. He spent five years in computer vision and deep learning research. His research interests include semi-supervised learning, unsupervised learning, and self-supervised learning.

He will join a company in Silicon Valley to continue his research and development in computer vision products and deep learning algorithms.