SLADE: A Self-Training Framework For Distance Metric Learning

Jiali Duan[1]* Yen-Liang Lin[2] Son Tran[2] Larry S. Davis[2] C.-C. Jay Kuo[1]
[1] University of Southern California [2] Amazon
{jialidua,jckuo}@usc.edu {yenliang,sontran,lrrydav}@amazon.com

Abstract

Most existing distance metric learning approaches use fully labeled data to learn the sample similarities in an embedding space. We present a self-training framework, SLADE, to improve retrieval performance by leveraging additional unlabeled data. We first train a teacher model on the labeled data and use it to generate pseudo labels for the unlabeled data. We then train a student model on both labels and pseudo labels to generate final feature embeddings. We use self-supervised representation learning to initialize the teacher model. To better deal with noisy pseudo labels generated by the teacher network, we design a new feature basis learning component for the student network, which learns basis functions of feature representations for unlabeled data. The learned basis vectors better measure the pairwise similarity and are used to select high-confident samples for training the student network. We evaluate our method on standard retrieval benchmarks: CUB-200, Cars-196 and In-shop. Experimental results demonstrate that with additional unlabeled data, our approach significantly improves the performance over the state-of-the-art methods.

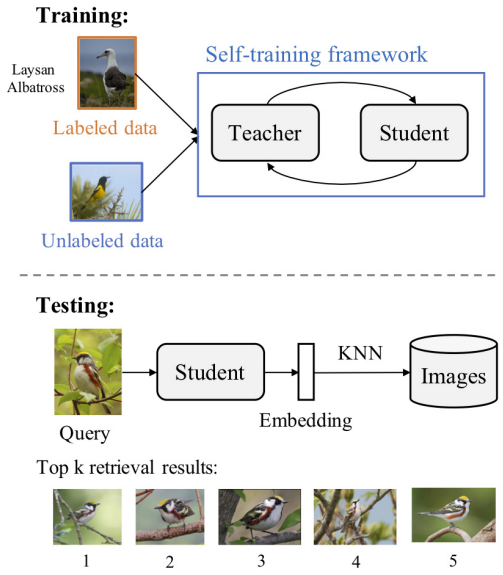

Figure 1. A self-training framework for retrieval. In the training phase, we train the teacher and student networks using both labeled and unlabeled data. In the testing phase, we use the learned student network to extract embeddings of query images for retrieval.

1. Introduction

Existing distance metric learning methods mainly learn sample similarities and image embeddings using labeled data [22, 17, 2, 29], which often require a large amount of data to perform well. A recent study [21] shows that most methods perform similarly when hyper-parameters are properly tuned despite employing various forms of losses. The performance gains likely come from the choice of network architecture. In this work, we explore another direction that uses unlabeled data to improve retrieval performance.

Recent methods in self-supervised learning [14, 6, 5] and self-training [32, 7] have shown promising results using unlabeled data. Self-supervised learning leverages unlabeled data to learn general features in a task-agnostic manner. These features can be transferred to downstream tasks by fine-tuning. Recent models show that the features produced by self-supervised learning achieve comparable performance to those produced by supervised learning for downstream tasks such as detection or classification [5]. Self-training methods [32, 7] improve the performance of fully-supervised approaches by utilizing a teacher/student paradigm. However, existing methods for self-supervised learning or self-training mainly focus on classification but not retrieval.

We present a **SeL**f-tr**A**ining framework for **D**istance m**E**tric learning (SLADE) by leveraging unlabeled data. Figure 1 illustrates our method. We first train a teacher model on the labeled dataset and use it to generate pseudo labels for the unlabeled data. We then train a student model on both labels and pseudo labels to generate a final feature

* Work done during an internship at Amazon.

embedding.

We utilize self-supervised representation learning to initialize the teacher network. Most deep metric learning approaches use models pre-trained on ImageNet ([17], [29], etc). Their extracted representations might over-fit to the pre-training objective such as classification and not generalize well to different downstream tasks including distance metric learning. In contrast, self-supervised representation learning [5, 6, 7, 14] learns task-neutral features and is closer to distance metric learning. For these reasons, we initialize our models using self-supervised learning approaches. Our experimental results (Table 3) provide an empirical justification for this choice.

Once the teacher model is pre-trained and fine-tuned, we use it to generate pseudo labels for unlabeled data. Ideally, we would directly use these pseudo labels to generate positive and negative pairs and train the student network. However in practice, these pseudo labels are noisy, which affects the performance of the student model (cf. Table 4). Moreover, due to their different sources, it is likely that the labeled and unlabeled data include different sets of categories (see Section 4.1 for details about labeled and unlabeled datasets). The features extracted from the embedding layer may not adequately represent samples from those unseen classes. To tackle these issues, we propose an additional representation layer after the embedding layer. This new layer is only used for unlabeled data and aims at learning basis functions for the feature representation of unlabeled data. The learning objective is contrastive, i.e. images from the same class are mapped close while images from different classes are mapped farther apart. We use the learned basis vectors to compute the feature representation of each image and measure pairwise similarity for unlabeled data. This enables us to select high-confident samples for training the student network. Once the student network is trained, we use it to extract embeddings of query images for retrieval.

We evaluate our model on several standard retrieval benchmarks: CUB-200, Cars-196 and In-shop. As shown in the experimental section, our approach outperforms several state-of-the-art methods on CUB-200 and Cars-196, and is competitive on In-shop. We also provide various ablation studies in the experimental section.

The main technical contributions of our work are:

- A self-training framework for distance metric learning, which utilizes unlabeled data to improve retrieval performance.

- A feature basis learning approach for the student network, which better deals with noisy pseudo labels generated by the teacher network on unlabeled data.

2. Related work

Distance metric learning is an active research area with numerous publications. Here we review those that are relevant to our work. While a common objective is to push similar samples closer to each other and different samples away from each other, approaches differ on their losses and sample mining methods. One can train a model using cross entropy loss [37], hinge loss [22], triplet loss [31], proxy-NCA loss [20, 17, 24], etc. [20] used the proxy-NCA loss to minimize the distance between a sample and their assigned anchor(s). These set of anchors were learnable. [17] further improved the proxy-based loss by combining it with a pair-based loss. We also use a set of learnable vectors but do not optimize directly on their distances to samples. Rather, we use them as a basis (anchor set) to represent output features operated on by a distribution loss. Our intuition is that while individual pseudo labels can be noisy, representing features using these anchors makes them more robust to noise [9], [35].

As mentioned previously, we use self-supervised training to seed our teacher model before fine-tuning it. There has been a significant progress recently in self-supervised learning of general visual representation [6, 7, 11, 5]. [6] learns representation by minimizing the similarity between two transformed versions of the same input. Transformations include various data augmentation operations such as crop, distort or blur. [7] narrowed the gap between self-supervised and supervised learning further by using larger models, a deeper projection head and a weight stabilization mechanism. In [5], a clustering step was applied on the output presentation. Their algorithm maximizes the consistency of cluster assignments between different transformations of the same input image. Most of these works aimed at learning a generic visual representation that can later be used in various downstream tasks.

Self-training involves knowledge distillation from larger, more complex models or from ensembles of models (teachers) to less powerful, smaller students (e.g., [15, 36, 33]). Their end purpose is often reducing model size. Recently, [32] and [39] used iterative self-training to improve classification accuracy of both teacher and student models. At a high level, our self-training is similar to these approaches, but it is designed for distance metric learning and semi-supervised learning settings.

Unlabeled data has been used to improve performance in various computer vision tasks such as classification and semantic segmentation (e.g., [23]). They have been also used in self-supervised and unsupervised representation learning such as in [32] or [38]. But there is still a performance gap compared to the fully supervised setting. For distance metric learning, most algorithms that are competitive on popular benchmarks (CUB-200, Cars-196 and In-shop) used fully labeled data ([17], [24], [29], etc). Here, we addi-

tionally use external unlabeled data to push performance on these datasets further.

3. Method

Figure 2 illustrates the system overview of our self-training framework. Our framework has three main components. First, we use self-supervised learning to initialize the teacher network, and then fine-tune it on labeled data. We use a pre-trained model [5] and fine-tune it on our data to initialize the teacher network (we experimented with different approaches to pre-train our model. They all led to improvements over the pre-trained ImageNet model. SwAV [5] was chosen as it led to the best performance - see the experimental section). After pre-training, we fine-tune the teacher network with a ranking loss (e.g., contrastive loss) on labeled data. The details of self-supervised pre-training and fine-tuning of the teacher network are presented in section 3.1.

Second, we use the fine-tuned teacher network to extract features and cluster the unlabeled data using k-means clustering. We use the cluster ids as pseudo labels. In practice, these pseudo labels are noisy. Directly optimizing the student network with these pseudo labels does not improve the performance of the teacher network. Therefore, we introduce a feature basis learning approach to select high-confidence samples for training the student network. The details of pseudo label generation are presented in Section 3.2.

Third, we optimize the student network and basis vectors using labeled and unlabeled data. The basis vectors are defined as a set of weights that map the feature embedding $\mathbf{f}$ of each image into a feature representation $\mathbf{r}$. We train the basis vectors such that images from the same class are mapped close and images from different classes are mapped farther apart. The basis vectors are used to select high-confidence samples for the ranking loss. The student network and basis vectors are optimized in an end-to-end manner. The details of student network optimization and feature basis learning are in Section 3.3.

3.1. Self-Supervised Pre-Training and Fine-Tuning for Teacher Network

Existing deep metric learning methods often use the ImageNet pre-trained model [10] for initialization, which may over-fit to the pre-training objective, and not generalize well to downstream tasks. Instead, we use self-supervised learning to initialize the teacher model. We use self-supervised pre-trained models ([5], [8], [4]) and fine-tune them on our data. As shown in the experimental section, this choice leads to improvement in retrieval performance as compared to the pre-trained ImageNet models (see Table 3). We conjecture that this might be because deep metric learning and self-supervised learning are related, they both learn embeddings that preserve distances between similar and dissimilar data.

Specifically, we are given a set of labeled images: $D^l = \{(x_1, y_1), (x_2, y_2), ..., (x_n, y_n)\}$ and unlabeled images: $D^u = \{\hat{x}_1, \hat{x}_2, ..., \hat{x}_m\}$. We denote the parameters of the teacher network as θ^t and the parameters of the student network as θ^s. In the pre-training stage, we fine tune the self-supervised model on the union of the labeled and unlabeled images without using the label information to initialize the teacher model. Once the teacher network is pre-trained, we fine-tune the teacher network using a ranking loss (for example, a constrastive loss [12]) on the labeled data:

$$L_{rank} = \sum_{(x_i,y_i)\in P} \max(d(x_i, y_i) - m_{\text{pos}}, 0) + \sum_{(x_i,y_i)\in N} \max(m_{\text{neg}} - d(x_i, y_i), 0) \quad (1)$$

where P is the positive pairs, N is the negative pairs, and m_{pos} and m_{neg} are the margins.

3.2. Pseudo Label Generation

We use the teacher model to extract features, and cluster the unlabeled images using k-means. The unlabeled images are assigned to the nearest cluster centers. The assigned cluster ids are then used as pseudo labels. One can train a student network with a ranking loss that uses the positive and negative pair samples sampled from the pseudo labels. However, in practice, the pseudo labels can be noisy, and unlabeled data may have unseen categories. The features extracted from the teacher model may not work well on those unseen categories (the pseudo labels are incorrectly estimated). To alleviate these issues, we constrain pseudo label generation to linear combinations of a set of basis vectors. These basis vectors are trained in a supervised manner using labeled data and can be considered as class centers of the labeled data [37]. Details are given in the next section.

3.3. Optimization of Student Network and Basis Vectors

We first explain the ideas of feature basis learning, and the use of basis vectors for sample mining and describe the training for student network and basis vectors.

3.3.1 Feature Basis Learning

Basis vectors are a set of learnable weights that map a feature embedding $\mathbf{f}$ of an image to a feature representation $\mathbf{r}$. We denote a set of basis vectors as $\{\mathbf{a}_1, \mathbf{a}_2, ..., \mathbf{a}_k\}$, where each basis vector $\mathbf{a}_i$ is a $d \times 1$ vector. For simplicity, we represent the basis vectors as a $k \times d$ matrix $\mathbf{W}_a$. Given an

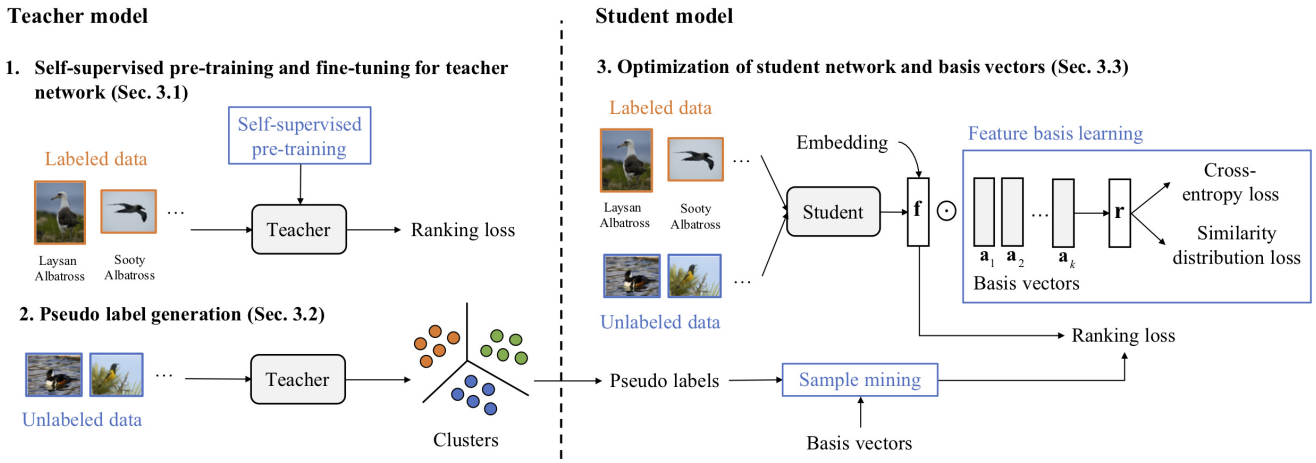

Figure 2. An overview of our self-training framework. Given labeled and unlabeled data, our framework has three main steps. (1) We first initialize the teacher network using self-supervised learning, and fine-tune it by a ranking loss on labeled data; (2) We use the learned teacher network to extract features, cluster and generate pseudo labels on unlabeled data; (3) We optimize the student network and basis vectors on labeled and unlabeled data. The purpose of feature basis learning is to select high-confidence samples (e.g., positive and negative pairs) for the ranking loss, so the student network can learn better and reduce over-fitting to noisy samples.

image $\mathbf{I}$, we use the student network to obtain the feature embedding $\mathbf{f}$ of the input image. The feature representation $\mathbf{r}$ is computed by $\mathbf{r} = \mathbf{W}_a \cdot \mathbf{f}$, where $r_i = \mathbf{a_i}^T \cdot \mathbf{f}$.

We train the basis vectors using two losses, a cross-entropy loss and a similarity distribution loss. The loss function for feature basis learning is defined as:

$$L_{Basis} = L_{CE} + L_{SD} \tag{2}$$

where the first term is the cross entropy loss on the labeled data and the second term is the similarity distribution loss on the unlabeled data.

The cross-entropy loss is applied on labeled data. The ground truth class labels can be used as a strong supervision signal to regularize the basis vectors to separate different classes. The cross entropy loss on labeled data is:

$$L_{CE} = \sum_{i=1}^{n} y_i \log(\sigma(\mathbf{W}_a f(x_i, \theta^s))) \tag{3}$$

where σ is the softmax function, $\mathbf{W}_a$ is the matrix for basis vectors, and θ^s is the parameters for the student network. Note that since a cross-entropy loss is used here, the columns of $\mathbf{W}$ approximate class centers [37]. Therefore, $\mathbf{r}$ can be viewed as a class-wise similarity representation or projections on the bases. It uses the class responses to interpolate the samples from unseen classes. This representation has also shown promising results on unlabeled data for other tasks (e.g., person Re-ID [35]).

For unlabeled data, one can also train a cross-entropy loss on the pseudo labels similar to labeled data. However, we found that this leads to poor performance since the model tends to over-fit to the noisy pseudo labels. Instead, we optimize with a global similarity distribution loss on the unlabeled data.

We use the pseudo labels to sample a set of pseudo positive pairs and pseudo negative pairs, where the pseudo positive pairs are sampled from the same pseudo class and the pseudo negative pairs are sampled from different pseudo classes. We compute the similarity of each image pair by using the cosine similarity of two normalized feature representation:

$$s(\hat{x}_i, \hat{x}_j) = \cos(\mathbf{r}_i, \mathbf{r}_j) = \cos(\mathbf{W}_a \mathbf{f}_i, \mathbf{W}_a \mathbf{f}_j) \tag{4}$$

Directly optimizing the model with individual similarity based on pseudo labels can lead to inferior performance due to noise. Instead, we model the similarities as two "global" Gaussian distributions G^+ and G^- and maximize the margin between them. These distributions are less sensitive to individual pseudo label noise (results in Table 5 give an empirical justification). Specifically, we aim to separate the two Gaussian distributions by maximizing the difference between their means and penalizing the variance of each distribution. The similarity distribution loss is defined as:

$$L_{SD}(G^+||G^-) = \max(\mu^- - \mu^+ + m, 0) + \lambda(\upsilon^+ + \upsilon^-) \tag{5}$$

where μ^+ (μ^-) and υ^+ (υ^-) are the mean and variance of the Gaussian distributions respectively, and m is the margin. We update the parameters in a batch-wise manner:

$$\begin{aligned} \mu^+ &= (1-\beta) \times \mu_b^+ + \beta \times \mu^+ \\ \upsilon^+ &= (1-\beta) \times \upsilon_b^+ + \beta \times \upsilon^+ \end{aligned} \tag{6}$$

where μ_b^+ and υ_b^+ are the mean and variance in a batch. β is the updating rate. The parameters of G^- are updated in a similar way.

3.3.2 Sample Mining

We use the basis vectors to select high-confidence sample pairs from the unlabeled images for training the student network. Given a set of samples in a batch, we compute the pair-wise similarity for all samples using equation 4 and select positive and negative pairs by:

$$\begin{aligned} P &= \{(\hat{x}_i, \hat{x}_j) | s(\hat{x}_i, \hat{x}_j) \geq T_1\} \\ N &= \{(\hat{x}_i, \hat{x}_j) | s(\hat{x}_i, \hat{x}_j) \leq T_2\} \end{aligned} \tag{7}$$

We set the confidence thresholds of T_1 and T_2 using μ^+ and μ^-. The positive and negative pairs will be used in the ranking function (see Equation 9).

3.3.3 Joint Training

We train the student network and basis vectors by minimizing a function L:

$$\min_{\theta^s, \mathbf{W}_a} L(\theta^s, \mathbf{W}_a) \tag{8}$$

$$\begin{aligned} L =& L_{rank}(D^l; \theta^s) + \lambda_1 L_{rank}(D^u; \theta^s) \\ &+ \lambda_2 L_{Basis}(D^l, D^u; \theta^s, \mathbf{W}_a) \end{aligned} \tag{9}$$

where L_{rank} is a ranking loss (see Equation 1). λ_1 and λ_2 are empirically chosen to make the magnitudes of the losses similar in scale. We train the ranking loss on both labeled and unlabeled images. Note that for unlabeled data, we use the sample mining to obtain the positive and negative pairs. Our framework is generic and applicable to different pair-based ranking losses. We report the results of different losses, e.g., contrastive loss and multi-similarity loss in Table 1.

We first train the basis vectors for a few iterations to get a good initialization, then train the student network and basis vectors end-to-end. After training the student network, we use the student as a new teacher and go back to the pseudo label generation step. We iterate this a few times. During testing, we discard the teacher model and only use the student model to extract the embedding of a query image for retrieval.

4. Experiments

We first introduce the experimental setup including datasets, evaluation criteria and implementation details. Then we report the results of our method on three common retrieval benchmarks CUB-200, Cars-196 and In-shop ([26, 18, 19]) [1]. Finally, we conduct ablation studies to analyze different design choices of the components in our framework.

[1]We did not carry out experiments on the SOP dataset [22], since we could not find an unlabeled dataset that is publicly available and is similar to it in content.

4.1. Datasets

CUB-200/NABirds: We use CUB-200-2011 [26] as the labeled data and NABirds [25] as the unlabeled data. CUB-200-2011 contains 200 fine-grained bird species, with a total number of 11,788 images. NABirds is the largest publicly available bird dataset with 400 species and 743 categories of North America's birds. It has 48,000 images with approximately 100 images for each species. We measure the overlaps between CUB-200 and NABirds, where there are 655/743 unseen classes in NABirds compared to CUB, showing the challenges of handling the out-of-domain images for unlabeled data.

Cars-196/CompCars: We use Cars-196 [18] as the labeled data, which contains 16,185 images of 196 classes of cars. Classes are annotated at the level of make, model, and year (e.g., 2012 Tesla Model S). We use CompCars [34] as the unlabeled data. It is collected at model level, so we filter out unbalanced categories to avoid being biased towards minority classes, resulting in 16,537 images categorized into 145 classes.

In-shop/Fashion200k: In-shop Clothes Retrieval Benchmark [19] includes 7,982 clothing items with 52,712 images. Different from CUB-200 and Cars196, In-shop is an instance-level retrieval task. Each article is considered as an individual category (each article has multiple views, such as front, back and side views), resulting in an average of 6.6 images per class. We use Fashion-200k [13] as the unlabeled data, since it has similar data organization (e.g., catalog images) as the In-shop dataset.

4.2. Evaluation Criteria

For CUB-200 and Cars-196, we follow the settings in [20, 21, 17] that use half of the classes for training and the other half for testing, and use the evaluation protocol in [21] to fairly compare different algorithms. They evaluate the retrieval performance by using MAP@R, RP and P@1. For each query, P@1 (also known as Recall@1 in previous metric learning papers) reflects whether the i-th retrieval result is correct. However, P@1 is not stable. For example, if only the first of all retrieval results is correct, P@1 is still 100%. RP measures the percentage of retrieval results that belong to the same class as the query. However, it does not take into account ranking of correct retrievals. $MAP@R = \frac{1}{R}\sum_{i=1}^{R} P@i$ combines the idea of mean average precision with RP and is a more accurate measure. For In-shop experiment, we use the Recall@K as the evaluation metric [19].

We compare our model with full-supervised baselines that are trained with 100% of labeled data and are fine-tuned end-to-end. Our settings are different from the ones in self-supervised learning frameworks [6, 5], where they evaluate the models in a label fraction setting (e.g., using 1% or

Methods	Frwk	Init	Arc / Dim	CUB-200-2011			Cars-196		
				MAP@R	RP	P@1	MAP@R	RP	P@1
Contrastive [12]	[21]	ImageNet	BN / 512	26.53	37.24	68.13	24.89	35.11	81.78
Triplet [31]	[21]	ImageNet	BN / 512	23.69	34.55	64.24	23.02	33.71	79.13
ProxyNCA [20]	[21]	ImageNet	BN / 512	24.21	35.14	65.69	25.38	35.62	83.56
N. Softmax [37]	[21]	ImageNet	BN / 512	25.25	35.99	65.65	26.00	36.20	83.16
CosFace [27, 28]	[21]	ImageNet	BN / 512	26.70	37.49	67.32	27.57	37.32	85.52
FastAP [3]	[21]	ImageNet	BN / 512	23.53	34.20	63.17	23.14	33.61	78.45
MS+Miner [29]	[21]	ImageNet	BN / 512	26.52	37.37	67.73	27.01	37.08	83.67
Proxy-Anchor[1] [17]	[17]	ImageNet	R50 / 512	-	-	69.9	-	-	87.7
Proxy-Anchor[2] [17]	[21]	ImageNet	R50 / 512	25.56	36.38	66.04	30.70	40.52	86.84
ProxyNCA++ [24]	[24]	ImageNet	R50 / 2048	-	-	72.2	-	-	90.1
Mutual-Info [1]	[1]	ImageNet	R50 / 2048	-	-	69.2	-	-	89.3
Contrastive [12] (T_1)	[21]	ImageNet	R50 / 512	25.02	35.83	65.28	25.97	36.40	81.22
Contrastive [12] (T_2)	[21]	SwAV	R50 / 512	29.29	39.81	71.15	31.73	41.15	88.07
SLADE (Ours) (S_1)	[21]	ImageNet	R50 / 512	29.38	40.16	68.92	31.38	40.96	85.8
SLADE (Ours) (S_2)	[21]	SwAV	R50 / 512	**33.59**	**44.01**	**73.19**	**36.24**	**44.82**	**91.06**
MS [29] (T_3)	[21]	ImageNet	R50 / 512	26.38	37.51	66.31	28.33	38.29	85.16
MS [29] (T_4)	[21]	SwAV	R50 / 512	29.22	40.15	70.81	33.42	42.66	89.33
SLADE (Ours) (S_3)	[21]	ImageNet	R50 / 512	30.90	41.85	69.58	32.05	41.50	87.38
SLADE (Ours) (S_4)	[21]	SwAV	R50 / 512	**33.90**	**44.36**	**74.09**	**37.98**	**46.92**	**91.53**

Table 1. MAP@R, RP, P@1 (%) on the CUB-200-2011 and Cars-196 datasets. Pre-trained Image-Net model is denoted as ImageNet and the fine-tuned SwAV model on our data is denoted as SwAV. The teacher networks (T_1, T_2, T_3 and T_4) are trained with the different losses, which are then used to train the student networks (S_1, S_2, S_3 and S_4) (e.g., the teacher T_1 is used to train the student S_1). Note that the results may not be directly comparable as some methods (e.g., [17, 24, 1]) report the results based on their own frameworks with different settings, e.g., embedding dimensions, batch sizes, data augmentation, optimizer etc. More detailed explanations are in Section 4.4.

10% labeled data from the same dataset for supervised fine-tuning or linear evaluation). Evaluating our model in such setting is important, since in practice, we often use all available labels to fine-tune the entire model to obtain the best performance. These settings also pose several challenges. First, our fully supervised models are stronger as they are trained with 100% of labels. Second, since our labeled and unlabeled data come from different image distributions, the model trained on labeled data may not work well on unlabeled data - there will be noisy pseudo labels we need to deal with. (Note that our focus is not the direct comparison to previous fully-supervised methods but to investigate the performance gain when using additional unlabeled data such as NABirds).

4.3. Implementation Details

We implement our model using the framework [21]. We use 4-fold cross validation and a batch size of 32 for both labeled and unlabeled data. ResNet-50 is used as our backbone network. In each fold, a student model is trained and outputs an 128-dim embedding, which will be concatenated into a 512-dim embedding for evaluation. We set the updating rate β to 0.99, and λ_1 and λ_2 in Equation 9 to 1 and 0.25 respectively to make the magnitude of each loss in a similar scale.

Recall@K	1	10	20	40
N. Softmax [37]	88.6	97.5	98.4	-
MS [29]	89.7	97.9	98.5	99.1
ProxyNCA++ [24]	90.4	98.1	98.8	99.2
Cont. w/M [30]	91.3	97.8	98.4	99.0
Proxy-Anchor [17]	91.5	98.1	98.8	99.1
SLADE (Ours) (S_4)	91.3	**98.6**	**99.0**	**99.4**

Table 2. Recall@K (%) on the In-shop dataset.

For iterative training, we train a teacher and a student model in each fold, then use the trained student model as the new teacher model for the next fold. This produces a better teacher model more quickly compared to getting a new teacher model after the student network finishes all training folds.

4.4. Results

The retrieval results for CUB-200 and Cars-196 are summarized in Table 1. We compare our method with state-of-the-art methods reported in [21] and some recent methods [17, 24, 1]. Note that numbers may not be directly comparable, as some methods use their own settings. For example, ProxyAnchor [17] uses a larger batch size of 120 for CUB-200 and Cars-196. It also uses the combination of global average pooling and global max pooling. Mutual-

Info [1] uses a batch size of 128 and a larger embedding size of 2048. ProxyNCA++ [24] uses a different global-pooling, layer normalization and data sampling scheme.

We evaluate the retrieval performance using original images for CUB-200 and Cars-196 rather than cropped images such as in [16] (CGD). For ProxyAnchor, in addition to reporting results using their own framework [17] (denoted as ProxyAnchor[1]), we also report the results using the framework [21] (denoted as ProxyAnchor[2]).

We use the evaluation protocol of [21] for fair comparison. We use ResNet50 instead of BN-Inception as our backbone network because it is commonly used in self-supervised frameworks. We experiment with different ranking losses in our framework, e.g., contrastive loss [12] and multi-similarity loss [29]. The teacher networks (T_1, T_2, T_3 and T_4) are used to train the corresponding student networks (S_1, S_2, S_3 and S_4) (e.g., teacher T_1 is used to train student S_1). We use the same loss for both teacher and student networks. For example, T_1 (teacher) and S_1 (student) are trained with the contrastive loss.

We also report the results of our method using different pre-trained models: pre-trained Image-Net model (denoted as ImageNet) and the self supervised pre-trained (then fine-tuned) SwAV [5] model. Note that we experimented with different approaches to pre-train our model (S2, Table 1). Their performance (MAP@R) on CUB200 are 30.0% for MoCo-v2 [8], 32.19% for Deep Cluster-v2 [4], and 33.59% for SwAV [5]. They all led to improvements over the pre-trained ImageNet model, which was also reported in Table 1 (S1: 29.38%). We chose SwAV [5] as it gives us the best performance.

We compare our method with the supervised baselines that are trained with 100% labeled data. Even in such setting, we still observe a significant improvement using our method compared to state-of-the-art approaches that use ResNet50 or BN-Inception. We boost the final performance to P@1 $= 74.09$ and P@1 $= 91.53$ for CUB-200 and Cars-196 respectively. The results validate the effectiveness of self-supervised pre-training for retrieval as well as the feature basis learning to improve the sample quality on unlabeled data. We also show that our method generalizes to different losses (e.g., contrastive loss [12] and multi-similarity (MS) loss [29]). Both losses lead to improvements using our method. The performance (P@1) of MS loss is improved from 66.31 to 74.09 on CUB-200, and 85.16 to 91.53 on Cars-196 respectively. We also report the performance of our method using the pre-trained ImageNet model (S_3), which achieves on-par performance with state-of-the-art approaches (e.g., Proxy-Anchor) even when we use a lower baseline model as the teacher (e.g., MS loss).

Table 2 summarizes the results for In-shop. Different from CUB-200 and Cars-196, In-shop is an instance-level retrieval task, where each individual article is considered as a category. Fashion200k is used as the unlabeled data. We train the teacher and student models using the multi-similarity loss [29] similar to the settings as T_4 and S_4 in Table 1. We report the results using Recall@K [19]. We achieve competitive results against several state-of-the-art methods. We note that the images in In-shop dataset are un-cropped while the images in Fashion200k dataset are cropped. So there exist notable distribution differences between these two datasets. We use the un-cropped version of In-shop to fairly compare with the baseline methods.

4.5. Ablation study

4.5.1 Initialization of Teacher Network

We first investigate the impact of using different pre-trained weights to initialize the teacher network (see Table 3). The results in the table are the final performance of our framework using different pre-trained weights. The teacher network is trained with a contrastive loss. We compare three different pre-trained weights: (1) a model trained on ImageNet with supervised learning; (2) a model trained on ImageNet with self-supervised approaches (specifically [5]); and (3) a self-supervised model with further fine-tuning on our data without using label information. From the results, we can see that the weights from self-supervised learning significantly improve the pre-trained ImageNet model, 4.21% and 4.86% improvements on CUB-200 and Cars-196 respectively. This validates the effectiveness of self-supervised pre-training for retrieval. We also find that fine-tuned model (3) further improves the performance ($\sim 1\%$ improvements) of pre-trained model (2).

Pre-trained weight	MAP@R	
	CUB-200	Cars-196
ImageNet [10]	29.38	31.38
Pre-trained SwAV [5]	32.79	35.54
Fine-tuned SwAV	33.59	36.24

Table 3. Comparison of different weight initialization schemes of the teacher network, where the teacher is trained with a contrastive loss. The results are the final performance of our framework.

4.5.2 Components in Student Network

Table 4 analyzes the importance of each component in our self-training framework. The results are based on the teacher network trained with a contrastive loss. Training the student with the positive and negative pairs sampled from pseudo labels only improves the teacher slightly, 1.52% and 0.26% on CUB-200 and Cars-196 respectively. The improvements are not very significant because the pseudo labels are noisy on unlabeled data. The performance is further improved by using the feature basis learning and sample mining, which supports the proposed method for bet-

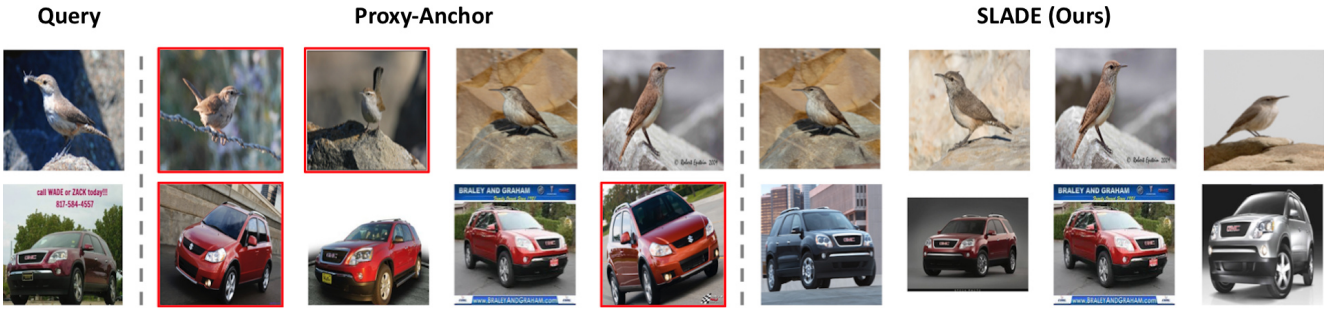

Figure 3. Retrieval results on CUB-200 and Cars-196. We show some challenging cases where our self-training method improves Proxy-Anchor [17]. Our results are generated based on the student model S_2 in Table 1. The red bounding boxes are incorrect predictions.

ter regularizing the embedding space with the feature basis learning and selecting the high-confidence sample pairs. We boost the final performance to 33.59 and 36.24 for CUB-200 and Cars-196 respectively.

Components	MAP@R	
	CUB-200	Cars-196
Teacher (contrastive)	29.29	31.73
Student (pseudo label)	30.81	31.99
+ Basis	32.45	35.78
+ Basis + Mining	33.59	36.24

Table 4. Ablation study of different components in our framework on CUB-200 and Cars-196. The teacher network is trained with a contrastive loss.

4.5.3 Pairwise Similarity Loss

We also investigate different design choices of the loss functions (Equation 5) used for feature basis learning. One alternative is to assign a binary label to each constructed pair according to the pseudo labels (e.g., 1 for pseudo-positive pair and 0 for pseudo-negative pair) and calculate a batch-wise cross entropy loss between pairwise similarities against these binary labels. We denote this option as local-CE. Another alternative is to first update the global Gaussian means using Equation 6, and then assign a binary label to the global means, followed by a cross entropy loss. We denote this option as global-CE. Similarity distribution loss performs better than cross-entropy, either locally or globally. The reason could be that basis vectors need not generalize to pseudo labels as they could be noisy.

Regularization	CUB-200		
	MAP@R	RP	P@1
Local-CE	32.69	43.20	72.64
Global-CE	32.23	42.68	72.45
SD (Ours)	33.59	44.01	73.19

Table 5. Accuracy of our model in MAP@R, RP and P@1 versus different loss designs on CUB-200.

4.5.4 Number of Clusters

Tables 6 analyzes the influence of different numbers of clusters on unlabeled data. The results are based on the teacher network trained with a contrastive loss. The best performance is obtained with $k = 400$, which is not surprising, as our model (student network) is trained on the NABirds dataset, which has 400 species. We can also see that our method is not sensitive to the number of clusters.

k	NABirds		
	MAP@R	RP	P@1
100	31.83	42.25	72.19
200	32.61	43.02	72.75
300	32.81	43.18	72.21
400	33.59	44.01	73.19
500	33.26	43.69	73.26

Table 6. Influence of using different numbers of clusters (k) on NABirds, which is used as the unlabeled data for CUB-200.

5. Conclusion

We presented a self-training framework for deep metric learning that improves the retrieval performance by using unlabeled data. Self-supervised learning is used to initialize the teacher model. To deal with noisy pseudo labels, we introduced a new feature basis learning approach that learns basis functions to better model pairwise similarity. The learned basis vectors are used to select high-confidence sample pairs, which reduces the noise introduced by the teacher network and allows the student network to learn more effectively. Our results on standard retrieval benchmarks demonstrate our method outperforms several state-of-the art methods, and significantly boosts the performance of fully-supervised approaches.

Acknowledgement: The authors would like to thank Zhibo Yang for helpful comments on SwAV experiments.

References

[1] Ismail Ben Ayed. A unifying mutual information view of metric learning: cross-entropy vs. pairwise losses. In *ECCV*, 2020. 6, 7

[2] Andrew Brown, Weidi Xie, Vicky Kalogeiton, and Andrew Zisserman. Smooth-ap: Smoothing the path towards large-scale image retrieval. In *ECCV*, 2020. 1

[3] Fatih Cakir, Kun He, Xide Xia, Brian Kulis, and Stan Sclaroff. Deep metric learning to rank. In *CVPR*, 2019. 6

[4] Mathilde Caron, Piotr Bojanowski, Armand Joulin, and Matthijs Douze. Deep clustering for unsupervised learning of visual features. In *European Conference on Computer Vision*, 2018. 3, 7

[5] Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. In *NeurIPS*, 2020. 1, 2, 3, 5, 7

[6] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *ICML*, 2020. 1, 2, 5

[7] Ting Chen, Simon Kornblith, Kevin Swersky, Mohammad Norouzi, and Geoffrey Hinton. Big self-supervised models are strong semi-supervised learners. In *NeurIPS*, 2020. 1, 2

[8] Xinlei Chen, Haoqi Fan, Ross Girshick, and Kaiming He. Improved baselines with momentum contrastive learning. *arXiv preprint arXiv:2003.04297*, 2020. 3, 7

[9] Yun-Chun Chen, Chao-Te Chou, and Yu-Chiang Frank Wang. Learning to learn in a semi-supervised fashion. *ECCV*, 2020. 2

[10] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009. 3, 7

[11] Jean-Bastien et al. Grill. Bootstrap your own latent: A new approach to self-supervised learning. *arXiv preprint arXiv:2006.07733*, 2020. 2

[12] Raia Hadsell, Sumit Chopra, and Yann LeCun. Dimensionality reduction by learning an invariant mapping. In *CVPR*, 2006. 3, 6, 7

[13] Xintong Han, Zuxuan Wu, Phoenix X Huang, Xiao Zhang, Menglong Zhu, Yuan Li, Yang Zhao, and Larry S Davis. Automatic spatially-aware fashion concept discovery. In *ICCV*, 2017. 5

[14] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *CVPR*, 2020. 1, 2

[15] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015. 2

[16] HeeJae Jun, Byungsoo Ko, Youngjoon Kim, Insik Kim, and Jongtack Kim. Combination of multiple global descriptors for image retrieval. *arXiv preprint arXiv:1903.10663*, 2020. 7

[17] Sungyeon Kim, Dongwon Kim, Minsu Cho, and Suha Kwak. Proxy anchor loss for deep metric learning. In *CVPR*, 2020. 1, 2, 5, 6, 7, 8

[18] Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3d object representations for fine-grained categorization. In *ICCV workshop*, 2013. 5

[19] Ziwei Liu, Ping Luo, Shi Qiu, Xiaogang Wang, and Xiaoou Tang. Deepfashion: Powering robust clothes recognition and retrieval with rich annotations. In *CVPR*, 2016. 5, 7

[20] Yair Movshovitz-Attias, Alexander Toshev, Thomas K Leung, Sergey Ioffe, and Saurabh Singh. No fuss distance metric learning using proxies. In *ICCV*, 2017. 2, 5, 6

[21] Kevin Musgrave, Serge Belongie, and Ser-Nam Lim. A metric learning reality check. *ECCV*, 2020. 1, 5, 6, 7

[22] Hyun Oh Song, Yu Xiang, Stefanie Jegelka, and Silvio Savarese. Deep metric learning via lifted structured feature embedding. In *CVPR*, 2016. 1, 2, 5

[23] Yassine Ouali, Céline Hudelot, and Myriam Tami. Semi-supervised semantic segmentation with cross-consistency training. In *CVPR*, 2020. 2

[24] Eu Wern Teh, Terrance DeVries, and Graham W Taylor. Proxynca++: Revisiting and revitalizing proxy neighborhood component analysis. In *ECCV*, 2020. 2, 6, 7

[25] Grant Van Horn, Steve Branson, Ryan Farrell, Scott Haber, Jessie Barry, Panos Ipeirotis, Pietro Perona, and Serge Belongie. Building a bird recognition app and large scale dataset with citizen scientists: The fine print in fine-grained dataset collection. In *CVPR*, 2015. 5

[26] Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. The caltech-ucsd birds-200-2011 dataset. 2011. 5

[27] Feng Wang, Jian Cheng, Weiyang Liu, and Haijun Liu. Additive margin softmax for face verification. *IEEE Signal Processing Letters*, 2018. 6

[28] Hao Wang, Yitong Wang, Zheng Zhou, Xing Ji, Dihong Gong, Jingchao Zhou, Zhifeng Li, and Wei Liu. Cosface: Large margin cosine loss for deep face recognition. In *CVPR*, 2018. 6

[29] Xun Wang, Xintong Han, Weilin Huang, Dengke Dong, and Matthew R Scott. Multi-similarity loss with general pair weighting for deep metric learning. In *CVPR*, 2019. 1, 2, 6, 7

[30] Xun Wang, Haozhi Zhang, Weilin Huang, and Matthew R Scott. Cross-batch memory for embedding learning. In *CVPR*, 2020. 6

[31] Kilian Q Weinberger and Lawrence K Saul. Distance metric learning for large margin nearest neighbor classification. *JMLR*, 2009. 2, 6

[32] Qizhe Xie, Minh-Thang Luong, Eduard Hovy, and Quoc V Le. Self-training with noisy student improves imagenet classification. In *CVPR*, 2020. 1, 2

[33] Guodong Xu, Ziwei Liu, Xiaoxiao Li, and Chen Change Loy. Knowledge distillation meets self-supervision. In *ECCV*, 2020. 2

[34] Linjie Yang, Ping Luo, Chen Change Loy, and Xiaoou Tang. A large-scale car dataset for fine-grained categorization and verification. In *CVPR*, 2015. 5

[35] Hong-Xing Yu, Wei-Shi Zheng, Ancong Wu, Xiaowei Guo, Shaogang Gong, and Jian-Huang Lai. Unsupervised person re-identification by soft multilabel learning. In *CVPR*, 2019. 2, 4

[36] Li Yuan, Francis EH Tay, Guilin Li, Tao Wang, and Jiashi Feng. Revisiting knowledge distillation via label smoothing regularization. In *CVPR*, 2020. 2

[37] Andrew Zhai and Hao-Yu Wu. Classification is a strong baseline for deep metric learning. In *BMVC*, 2019. 2, 3, 4, 6

[38] Xiaohang Zhan, Jiahao Xie, Ziwei Liu, Yew-Soon Ong, and Chen Change Loy. Online deep clustering for unsupervised representation learning. In *CVPR*, 2020. 2

[39] Barret Zoph, Golnaz Ghiasi, Tsung-Yi Lin, Yin Cui, Hanxiao Liu, Ekin D Cubuk, and Quoc V Le. Rethinking pre-training and self-training. *arXiv preprint arXiv:2006.06882*, 2020. 2