Improving Self-supervised Lightweight Model Learning via Hard-aware Metric Distillation

Hao Liu[2], Mang Ye[1,3⋆]

[1] School of Computer Science, Wuhan University
[2] School of Computer Science, Beijing Institute of Technology
[3] Hubei Luojia Laboratory
https://github.com/liuhao-lh/SMD

Abstract. The performance of self-supervised learning (SSL) models is hindered by the scale of the network. Existing SSL methods suffer a precipitous drop in lightweight models, which is important for many mobile devices. To address this problem, we propose a method to improve the lightweight network (as student) via distilling the metric knowledge in a larger SSL model (as teacher). We exploit the relation between teacher and student to mine the positive and negative supervision from the unlabeled data, which captures more accurate supervision signals. To adaptively handle the uncertainty in positive and negative sample pairs, we incorporate a dynamic weighting strategy to the metric relation between embeddings. Different from previous self-supervised distillers, our solution directly optimizes the network from a metric transfer perspective by utilizing the relationships between samples and networks, without additional SSL constraints. Our method significantly boosts the performance of lightweight networks and outperforms existing distillers with fewer training epochs on the large-scale ImageNet. Interestingly, the SSL performance even beats the teacher network in several settings.

1 Introduction

As an effective way to explore the information in the data itself, self-supervised learning (SSL) can obtain discriminative task-agnostic representations, while allowing the training without prohibitively expensive data annotation. Recently, there has been impressive progress in SSL [4,6,7,11,15,35]. Some SSL models trained on the large-scale ImageNet without labels have achieved comparable or even better accuracy than the supervised models when transferred to downstream tasks, such as semi-supervised image classification and object detection.

Generally, the gap between supervised and self-supervised is much smaller when increasing the capacity of the network architecture [1]. Therefore, the top-performing SSL algorithms usually require large networks as their backbone. This greatly limits the applicability of SSL for many mobile devices with limited capacity, *i.e.,* the lightweight model is preferred in edge computing applications.

⋆ Corresponding Author: *Mang Ye*

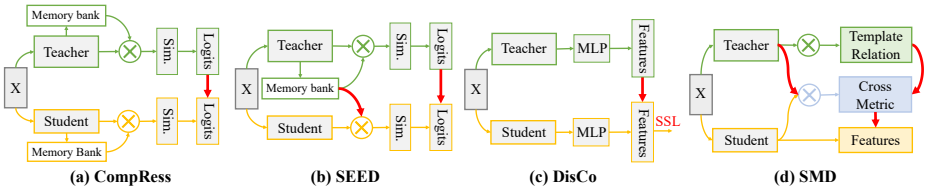

Fig. 1: Comparison with existing self-supervised distillers. X is the input images. The red arrow indicates the knowledge transfer direction. Both (a) CompRess [1] and (b) SEED [9] transfer the knowledge of the similarity between a sample and a negative memory bank. (c) DisCo [10] constrains the last embedding of the student to be consistent with that of the teacher. (d) Our SMD mines and optimizes the metric relationship between samples in a batch, which does not need to maintain a memory bank. It also utilizes embedding relations between the teacher and student models.

However, existing SSL methods do not work well on the lightweight network. For example, the linear evaluation of MobileNet-V3-Large [19], its Top-1 accuracy on ImageNet using MoCo-V2 is only 36.2%, which is far from satisfying compared to its fully supervised counterparts 75.2% [10]. This motivates us to investigate a feasible solution to design powerful lightweight SSL models.

To learn better representations for lightweight self-supervised models, many researchers set out to solve this problem from the perspective of knowledge distillation. Most supervised distillers [18,29] cannot be extended to self-supervised distillation since the self-supervision paradigm does not contain any training labels. We illustrate some recently proposed self-supervised distillers in Fig.1. CompRess [1] and SEED [9] utilize the memory bank in MoCo [15] for knowledge transfer. DisCo [10] and SimDis [12] extract extra supervision from the representation and optimize the student model by aligning the features between the same samples. However, they are constrained by the self-supervised learning framework that simply assumes non-target images as negative samples. None of them has explored the relationships between samples under the self-supervised knowledge distillation task.

In the well-trained larger SSL teacher space, we assume that the distance between positive sample pairs is small, and the distance between negative sample pairs is large. During the distillation, the lightweight student optimizes toward the teacher, *i.e.*, for the same sample, the distance between the teacher features and student features should not be large, where we name this distance as *teacher-to-student metric difference*. In the teacher space, if the distance between two samples is smaller than the above *teacher-to-student metric difference*, we can consider that the distance is small enough so that these two samples are treated as positive sample pairs. Otherwise, these two samples are negative pairs. Based on this idea, we divide unlabeled data into positive and negative sample pairs and thus perform effective knowledge distillation from the perspective of the metric relation under the self-supervised learning setting, *i.e.*, making positive

pairs concentrated and negative pairs separated. We utilize the metric differences between the well-trained teacher and to-be-optimized student model, acting as a dynamic decision boundary to divide positive and negative sets without category label supervision.

In this work, we propose a novel Self-supervised Metric Distillation (SMD) for lightweight model learning. Specifically, we optimize the metric relation between teacher and student embeddings. Unlike previous self-supervised training and distillation methods, we focus on optimizing difficult pairs rather than all sample pairs. Therefore, we introduce a hard mining strategy to improve the widely adopted *Info-NCE* [23]. Inevitably, there will be some incorrectly divided positive and negative sample pairs. To reduce their influence, we generate weight coefficients based on the metric relation between teacher and student embeddings. This prevents incorrectly divided samples from affecting the optimization and also gives different sample pairs different optimization strengths, thereby effectively improving the performance of the student model. With the help of the above strategies, our SMD has achieved compelling results without any complex projectors [10,12] or a huge negative sample queue [9,1].

Our method performs more accurate optimization based on the positive and negative sample pairs mined by the teacher-guided metric difference. We present comprehensive experiments on CIFAR100 and ImageNet to evaluate our distillation method in the self-supervised manner. Furthermore, our method can also be deployed in a supervised distillation framework, which not only surpasses many supervised distillation methods but also obtains state-of-the-art results. Overall, our contributions can be summarized as follows:

- In the self-supervised distillation framework, we are the first to attempt to explore the explicit relationship between samples and find an effective strategy to divide positive and negative pairs from unlabeled data through the *teacher-to-student metric difference*.
- We design a self-supervised hard-aware metric distillation approach that optimizes the student embeddings by adaptively utilizing the teacher relation supervision, which can weaken the impact of incorrectly divided samples.
- Our method significantly improves the performance of lightweight models in the self-supervised knowledge distillation task and achieves a much higher accuracy than other methods in self-supervised knowledge distillation.
- Our method can also be seamlessly applied in the supervised distillation framework without network modification and achieves much a higher accuracy than existing supervised distillers. The learned student network even outperforms the teacher in some settings.

2 Related Work

Supervised Knowledge Distillation is commonly used in the supervised paradigm to improve the performance of lightweight models under extra supervision from powerful but large models, which is also called model compression [3].

Early work achieved this by aligning the network logits [18] or the intermediate representations [26]. Recently, many works have proposed to transfer carefully designed statistics from teacher to student networks, which includes attention maps [38], mutual information [2,29], probability distributions [25], maximum mean discrepancies [20] and activation boundaries of the hidden neurons [17].

Self-supervised Knowledge Distillation. Most effective distillation have a supervised loss term [18,29,5]. Thus, they cannot be directly applied to self-supervised models since the self-supervision paradigm does not contain any training labels. Some recent works have made some attempts to solve this problem. Both CompR [1] and SEED [9] are inherited from the idea of the negative sample queue in MoCo [15]. They use KL-divergence to align two probability distributions, which is obtained by computing the similarity between the sample features and the features stored in the queue. For DisCo [10] and SimDis [12], their core idea is to optimize the output of the teacher model and the student model for the same sample via L_2 loss. All these methods are constrained by the idea of the self-supervision training framework, where all samples except the target are considered as negative samples. Our SMD attempts to dig deeper into the relationship between samples from unlabeled data in the knowledge distillation framework and optimize specific sample pairs directly from a metric learning perspective, which as far as we know has not yet been investigated.

3 Proposed Method

Our SMD aims at optimizing the student features by distilling the feature embeddings learned from the teacher. The main idea is to find the set of positive and negative sample pairs from the unlabeled data, then optimize the relative distances between the fixed teacher features and optimizable student features, reinforcing the student features' discriminability under the teacher's supervision.

3.1 Difference-Guided Positive and Negative Mining

Given a well-trained self-supervised teacher model $f^t(\cdot)$ and a randomly initialized student model $f^s(\cdot)$, the student/teacher feature of an unlabeled input image x_i is represented as $\boldsymbol{f}^s(x_i)/\boldsymbol{f}^t(x_i)$. The core idea of the previous approaches[12,10] is to directly optimize the features between $\boldsymbol{f}^s(x_i)$ and $\boldsymbol{f}^t(x_i)$. This obviously ignores the influence of the critical negative samples [14,32]. However, it is not feasible to randomly choose one of the unlabeled data as the negative sample because the network performance will be affected if the selected one is a positive sample. Since there is a huge gap between a well pre-trained teacher model and a randomly initialized student model. This leads us to the following intuition: *Could we find the positive and negative samples in the unlabeled data based on the differences between teacher and student models?*

Take x_i as the anchor. For another image x_a in the batch, we calculate its teacher embedding $\boldsymbol{f}^t(x_a)$. Let $D(\cdot)$ be a metric function measuring the distance between two images in the embedding space. A larger D indicates a

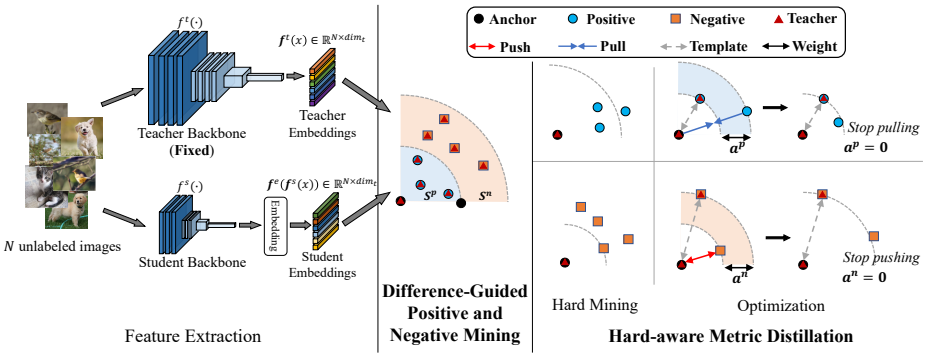

Fig. 2: Schematization of SMD: 1) *Feature Extraction:* Teacher and student extract embeddings for unlabeled images. 2) *Difference-Guided Positive and Negative Mining* § 3.1: The positive and negative samples are divided according to the embeddings of teacher and student networks. The embeddings with red triangles (▲) are from the teacher network and those without red triangles are from the student network. 3) *Hard-aware Metric Distillation* § 3.2: Find the hardest samples from the positive and negative set respectively, and then optimize the cross metric between student and teacher embeddings based on template relations. Best viewed in color. Zoom in for details.

lower similarity between two embeddings. Here, we adopt the Euclidean distance $D(\boldsymbol{f}(x_i), \boldsymbol{f}(x_j)) = \|\bar{\boldsymbol{f}}(x_i) - \bar{\boldsymbol{f}}(x_j)\|_2$, in which $\bar{\boldsymbol{f}}(x_i)$ and $\bar{\boldsymbol{f}}(x_j)$ are normalized embedding features. Since the teacher is well trained, $D(\boldsymbol{f}^t(x_i), \boldsymbol{f}^t(x_a))$ is generally small if x_i and x_a belong to the same category. We identify x_i and x_a as a positive pair when $D(\boldsymbol{f}^t(x_i), \boldsymbol{f}^t(x_a))$ is smaller than a decision boundary, and vice versa [34]. Here, the decision boundary about D is critical. Setting it as a hyper-parameter is a spontaneous idea for tackling this problem, but it is impossible to separate all positive and negative sample pairs with only one hyper-parameter. During the distillation, the lightweight student optimizes toward the teacher, but the cumbersome teacher network and the lightweight student models are usually very different in structure, which will make it impossible for $\boldsymbol{f}^t(x_i)$ and $\boldsymbol{f}^s(x_i)$ to match perfectly. We name this little gap *teacher-to-student metric difference*, and tentatively believe that it can provide some useful information:

$$(x_i, x_a) \in \begin{cases} S_i^p, & D(\boldsymbol{f}^t(x_i), \boldsymbol{f}^t(x_a)) < D(\boldsymbol{f}^t(x_i), \boldsymbol{f}^s(x_i)), \\ S_i^n, & D(\boldsymbol{f}^t(x_i), \boldsymbol{f}^t(x_a)) \geq D(\boldsymbol{f}^t(x_i), \boldsymbol{f}^s(x_i)). \end{cases} \tag{1}$$

For x_i, other samples in the batch are divided into the positive set S_i^p and negative set S_i^n. Moreover, different samples have different *teacher-to-student metric differences* $D(\boldsymbol{f}^t(x_i), \boldsymbol{f}^s(x_i))$, which serve as the dynamical decision boundaries.

3.2 Hard-aware Metric Distillation

We assume that the obtained S_i^p and S_i^n are both reliable. The primary goal of our hard-aware metric distillation is to optimize the cross metric relation between teacher and student features in the embedding space, *i.e.*, for every sample x_i, pull $\boldsymbol{f}^t(x_i)$ and $\boldsymbol{f}^s(x_j)$ closer while pushing $\boldsymbol{f}^t(x_i)$ and $\boldsymbol{f}^s(x_k)$ apart, where $x_j \in S_i^p$, and $x_k \in S_i^n$. For simplicity, we use the shorthand notation $d_{ij}^p = D(\boldsymbol{f}^t(x_i), \boldsymbol{f}^s(x_j))$, $d_{ik}^n = D(\boldsymbol{f}^t(x_i), \boldsymbol{f}^s(x_k))$. The learning target is to increase all d_{ik}^n and decrease all d_{ij}^p. To achieve this, we adopt an *Info-NCE*-like loss [23] to distill the metric knowledge learned in the teacher network:

$$\mathcal{L}_i = -\log \frac{\sum_{k=1}^{K} \exp(d_{ik}^n/\tau)}{\sum_{k=1}^{K} \exp(d_{ik}^n/\tau) + \sum_{j=1}^{J} \exp(d_{ij}^p/\tau)}, \tag{2}$$

where τ is a temperature hyper-parameter. K and J represent the number of samples in S_i^n and S_i^p, respectively. For every x_i, the student network pulls all positive embeddings $\boldsymbol{f}^s(x_j)$ towards $\boldsymbol{f}^t(x_i)$, and pushes all negative embeddings $\boldsymbol{f}^s(x_k)$ away from $\boldsymbol{f}^t(x_i)$.

However, it is cost-prohibitive to perform the above optimization for all sample pairs even in a batch. To improve this, we revisit Eq.(2), in which τ is always set to quite small [33,15,9]. This will sharpen the probability distribution [18], which makes a few key samples dominate the entire loss. From this, we speculate that the hardest case should play a key role. We utilize an online batch hard mining strategy based on the S_i^p and S_i^n. Specifically, we select the positive and negative cross-model pairs within each sampled batch online. First, we calculate the cross-relation embedding distance of each image pair for the student and teacher networks in a batch. For each anchor's teacher embedding $\boldsymbol{f}^t(x_i)$, we select its hardest positive and hardest negative samples from the student embeddings. The online mining process is formulated as:

$$d_i^p = \max_{j \in S_i^p} d_{ij}^p, \quad d_i^n = \min_{k \in S_i^n} d_{ik}^n, \tag{3}$$

where d_i^p and d_i^n represent the hardest positive and negative pair mined from the student features for anchor x_i. With the selected hard positives and negatives, our learning target evolves from Eq.(2) to:

$$\mathcal{L}_i = -\log \frac{\exp(d_i^n/\tau)}{\exp(d_i^n/\tau) + \exp(d_i^p/\tau)}. \tag{4}$$

The denominator only contains one hardest positive pair and one hardest negative pair. By minimizing Eq.(4), we pull the hardest positive pair closer and push the hardest negative pair farther.

There will inevitably be some incorrectly divided sample pairs in S_i^p and S_i^n. Though Eq.(4) allows the optimization process to focus on a few key samples, it also amplifies the impact of these incorrectly divided pairs. We can prevent these wrong samples from hindering the optimization process if they can be ruled out,

i.e., assign zero weight to these samples. Here, we improve Eq.(4) by a teacher-guided self-supervised weighting mechanism, which re-assigns the contribution of different d_i^p and d_i^n in the overall optimization process:

$$\mathcal{L}_i^{md} = -\log \frac{\exp(a_i^n d_i^n/\tau)}{\exp(a_i^n d_i^n/\tau) + \exp(a_i^p d_i^p/\tau)}, \tag{5}$$

where a_i^p and a_i^n are non-negative weighting factors.

Given an anchor sample x_i with teacher features $\boldsymbol{f}^t(x_i)$, we first get its hardest positive x_{j^h} and the hardest negative samples x_{k^h} from S_i^p and S_i^n. After identifying the hardest sample index from all the student features, we calculate the distances $D(\boldsymbol{f}^t(x_i), \boldsymbol{f}^t(x_{j^h}))$ and $D(\boldsymbol{f}^t(x_i), \boldsymbol{f}^t(x_{k^h}))$ within the teacher embedding space. These two distances are encoded as the main guidance for optimizing d_i^p and d_i^n, so a_i^p and a_i^n in Eq.(5) are defined in a teacher-guided self-supervised manner as:

$$\begin{cases} a_i^p = [d_i^p - D(\boldsymbol{f}^t(x_i), \boldsymbol{f}^t(x_{j^h}))]_+, \\ a_i^n = [D(\boldsymbol{f}^t(x_i), \boldsymbol{f}^t(x_{k^h})) - d_i^n]_+. \end{cases} \tag{6}$$

where $D(\boldsymbol{f}^t(x_i), \boldsymbol{f}^t(x_{j^h}))$ and $D(\boldsymbol{f}^t(x_i), \boldsymbol{f}^t(x_{k^h}))$ are template relations; $[]_+$ is the "cut-off at zero" operation to ensure a_i^p and a_j^n are non-negative, and also stops the optimization process when the student network outperforms the teacher network. When the sample pair is incorrectly divided, optimizing Eq.(5) will go against our goal if all weighting factors are 1. With Eq.(6), the optimization will be stopped when the false-negative samples are pushed far beyond the template relation or false-positive samples are pulled close within the template relation. a_i^p and a_i^n will set the optimization of the incorrect pair right. Here the template relation is the lower bound of the worst case. When the sample pair is correctly divided, the ideal convergence status is that d_i^p and d_i^n are better than the template relation instead of pulling positive samples as close as possible and pushing negative samples as far as possible. This moderate training goal can also improve the generalization ability.

3.3 Learning Constraint

Embedding Alignment. Since student and teacher networks have different structures, their embedding spaces are often different. It is difficult for the student network to find the correct embedding space where the teacher network features are located since our method is to optimize the metric relationship between the features. To solve this, we added $\mathcal{L}_2$ constraint in the first few training epochs to directly align the embedding space of the teacher and student networks:

$$\mathcal{L}_i^{align} = ||\bar{\boldsymbol{f}}^t(x_i) - \bar{\boldsymbol{f}}^s(x_i)||^2.$$

Feature Embedding Transformation. To calculate d_i^p and d_i^n in Eq.(5), a straightforward solution is to directly adopt the backbone features of the student and teacher networks as $\boldsymbol{f}^s(x)$ and $\boldsymbol{f}^t(x)$. However, the student and teacher

usually have different backbone structures, thus $\boldsymbol{f}^s(x)$ and $\boldsymbol{f}^t(x)$ are not directly comparable due to different dimensions dim_s and dim_t. Therefore, we introduce an embedding layer $e(\cdot)$ to encode the student backbone features $\boldsymbol{f}^s(x)$. The embedding layer is composed of a `FC` layer. The output of embedding layer $\boldsymbol{f}^e(\boldsymbol{f}^s(x))$ with dimension dim_t takes the place of student backbone features $\boldsymbol{f}^s(x)$ for hard-aware metric distillation. Compared with the backbone features, the feature dimension of $\boldsymbol{f}^e(\boldsymbol{f}^s(x))$ is easier to adjust when the teacher and student networks have different structures.

The overall training loss for our proposed SMD is:

$$\mathcal{L}_{SMD} = \frac{1}{N}\sum\nolimits_{i=1}^{N}(\mathcal{L}_i^{md} + \mathcal{L}_i^{align}), \tag{7}$$

where N is the batch size. Experimentally, it is sufficient to add the $\mathcal{L}_2$ constraint only in the first one or two epochs, which only accounts for 0.2% to 1% of the whole training process. Without this constraint, the training would not converge in most scenarios from experiments in § 4.4.

4 Experiments

4.1 Self-supervised Distillation on CIFAR100

Dataset. In this section, all the experiments are conducted on CIFAR-100 [21], which contains 50K training images and 10K test images from a total of 100 classes. The image size is 32 × 32.

Self-Supervised Pre-training of Vanilla Network. We use SimSiam [7] to pre-train the vanilla teacher and student network. ResNet [16] and Wide-ResNet [37] are selected as the teacher network. A projector (a two-layer multi-layer-perceptron (MLP) with batch normalization (BN)) and a predictor (a two-layer MLP and the output linear layer has no BN and ReLU) are appended at the end of the encoder (backbone) after average pooling. All networks are pre-trained for 800 epochs by a standard SGD optimizer with a momentum of 0.9 and a weight decay parameter of 5e-4. The initial learning rate is set as 0.06 and updated by a cosine decay scheduler with 10 warm-up epochs. The batch size is 512. For MobileNet and ShuffleNet, the initial learning rate is 0.012.

Self-Supervised Distillation on Student Network. We choose multiple lightweight networks as the student network: ResNet and Wide-ResNet with fewer layers, MobileNet [27] and ShuffleNet [22]. The hyper-parameter τ is set to 0.02. All training strategies are consistent with the vanilla network.

Linear and kNN Evaluation. In order to validate the effectiveness of our SMD, we treat the network as a frozen feature extractor after self-supervised pre-training/distillation and train a linear classifier on the labeled training set. The initial learning rate is set as 30 and updated by a cosine decay scheduler. The batch size is 256. SGD optimizer with momentum 0.9 is used for 100 epochs training and the weight decay is 0. We also perform classification using K-Nearest Neighbors (kNN) based on the backbone feature by taking the most frequent label of its K (K = 200) nearest neighbors.

	T:*resnet* 32×4 S:*resnet* 32		T:*WRN*-40-2 S:*WRN*-16-2		T:*resnet* 32×4 S:*MobileNetV2*		T:*ResNet*-50 S:*ShuffleNetV2*		T:*WRN*-40-2 S:*ShuffleNetV2*	
Methods	kNN	Top-1	kNN	Top-1	kNN	Top-1	kNN	Top-1	kNN	Top-1
Teacher	60.63	67.27	56.16	63.16	61.89	69.96	60.63	67.27	56.16	63.16
Student	46.35	52.54	48.45	58.58	26.88	35.87	31.78	48.90	31.78	48.90
L_2	49.68	55.48	52.60	60.16	54.34	58.43	58.52	66.25	55.37	63.73
SSL+SMD	52.52	57.97	53.42	60.92	56.89	61.32	60.12	67.43	**57.32**	67.11
SMD	**52.54**	**58.15**	**53.53**	**60.98**	**57.12**	**61.89**	**60.21**	**67.70**	57.16	**67.88**

Table 1: **Results on the CIFAR-100 under self-supervised distillation framework.** The kNN accuracy (%) and Top-1 classification accuracy (%) for transferring across some teacher and student architectures.

Table 1 compares Top-1 accuracy using kNN and using linear evaluation (Top-1) with different teacher-student combinations. We list the baseline results of self-supervised pre-training using SimSiam in the first and second rows for each teacher-student combination. We observe that our SMD consistently outperforms the student baseline. Moreover, the performance of the student network will be improved with a better teacher network. In addition, we explore several alternative distillation strategies. L_2: the core idea of [10] and [12]. The embeddings of the same image are pulled closer. *SMD+SSL*: add the self-supervised loss to our method. Our SMD has a considerable improvement compared to L_2 when the structure of the teacher and student network is quite different (the last three columns). We conjecture that optimizing more abstract metric knowledge is easier to generalize than directly pulling two embeddings that are not in the same space closer. Moreover, we study the effect of the original SSL supervision as supplementary loss (SSL+SMD) and find it is not necessary for distillation. Perhaps the goals of distillation and self-supervised learning are different. Distillation forces the student network to mimic the predictions of the teacher network while self-supervised learning attempts to discover and learn latent patterns from the data itself, where the former contains more fruitful information.

4.2 Self-supervised Distillation on ImageNet

Dataset. In this section, all the experiments are conducted on the large-scale ImageNet 2012 dataset [8], which provides 1,281,167 images from 1,000 classes for training and 50,000 for validation.

Self-Supervised Pre-training of Vanilla Network. We use the official model ResNet-50 (100 epochs) released by SimSiam [7] as the teacher network. The SimSiam baseline of the student models is trained with the official SimSiam strategy, where all networks are pre-trained for 100 epochs by an SGD optimizer with a momentum of 0.9 and a weight decay parameter of 1e-4. The initial learning rate is 0.1 with a 512 batch size and updated by a cosine decay scheduler.

Self-Supervised Distillation on Student Network. We choose multiple lightweight networks as the student network: ResNet with 18 and 34 layers, MobileNet-V3-Large [19], EfficientNet-B0 and EfficientNet-B1 [28]. τ is set to 0.04. All training strategies are consistent with the vanilla network.

		ResNet-18		ResNet-34		Efficient-B0		Efficient-B1		Mobile-V3	
	Epoch	Top-1	Top-5	Top-1	Top-5	Top-1	Top-5	Top-1	Top-5	Top-1	Top-5
Supervised		69.5	89.1	73.3	91.4	77.3	95.3	79.2	94.5	75.2	92.2
MoCo-V2	200	52.5	77.0	57.4	81.6	42.2	68.5	50.7	74.6	36.3	62.2
SEED	200	57.9	82.0	58.5	82.6	61.3	82.7	61.4	83.1	55.2	80.3
DisCo	200	60.6	83.7	62.5	85.4	66.5	87.6	66.6	87.5	64.4	86.2
SimSiam	100	30.5	57.2	33.4	59.7	-	-	-	-	-	-
CompR-1q*	100	60.5	83.0	62.1	84.2	65.2	86.3	65.8	86.5	62.7	85.1
SMD	100	**61.8**	**84.3**	**64.1**	**86.0**	**66.5**	87.4	**66.8**	87.3	**64.5**	**86.7**

Table 2: **Comparison of distillation methods on full ImageNet**. Top-1 and Top-5 linear evaluation accuracy (%) for multiple students on full ImageNet validation set. The teacher network for all methods is ResNet-50. MoCo-V2 and SimSiam denote self-supervised learning baselines before distillation. We denote by * methods where we use our re-implementation based on the paper; for all other methods we use the results reported by their original papers. The first row shows the supervised performances of student networks. "-" means the student network collapses or has poor results.

Linear Evaluation. We train a linear classifier on top of the frozen network encoder after self-supervised pre-training/distillation. Use the LARS optimizer [36] for 90 epochs training with a cosine decay learning rate schedule. The base learning rate is 0.1, scaled linearly with the batch size (LearningRate = 0.1 $\times$ BatchSize/256). The results are reported in terms of Top-1 and Top-5 accuracy.

Table 2 compares the results of different distillation objectives on multiple teacher-student pairs. The SimSiam(baseline) even collapses in some cases, and related studies [9] conjecture that smaller models with fewer parameters cannot effectively learn instance-level discriminative representation with a large amount of unlabeled data. The student models distilled by SMD outperform the counterparts pre-trained by SimSiam by a large margin. Besides, we observe that our SMD consistently outperforms all other distillation objectives. Note that SEED [9] and DisCo [10] use the MoCo-V2 [15] training strategy that performs better on lightweight models and needs more training epochs. Due to the computational limitation, unlike SEED and DisCo, we use the SimSiam baseline that only needs to train 100 epochs. Almost all papers about self-supervised learning and distillation have verified that the final results benefit from more training epochs [6,10]. In addition, DisCo needs an MLP with a large dimension of the hidden layer to deal with the *distilling bottleneck* phenomenon. SEED needs to maintain a large negative sample queue with a length of 65,536. The additional parameters required by our SMD are only a fully connected layer used to align the feature dimensions of the teacher and student models. Our method achieves better results than other methods on a weaker baseline and fewer additional parameters, verifying that SMD can learn effective metric relationships from unlabeled data.

The self-supervised distillation performance is greatly influenced by the training strategy, so we re-implement CompR [1] by the training strategy of SMD. Under the same training strategy, our SMD also surpasses CompR-1q by a large

	WRN-40-2 WRN-16-2	WRN-40-2 WRN-40-1	resnet56 resnet20	resnet110 resnet20	resnet110 resnet32	vgg13 vgg8
Teacher	75.61	75.61	72.34	74.31	74.31	74.64
Student	73.26	71.98	69.06	69.06	71.14	70.36
KD[18]	74.92	73.54	70.66	70.67	73.08	72.98
FitNet[26]	73.58	72.24	69.21	68.99	71.06	71.02
SP[30]	73.83	72.43	69.67	70.04	72.69	72.68
RKD[24]	73.35	72.22	69.61	69.25	71.82	71.48
KDCL[13]	73.93	73.12	70.58	70.36	72.67	72.94
CRD[29]	75.48	74.14	71.16	71.46	73.48	73.94
WCoRD[5]	75.88	74.73	71.56	71.57	73.81	74.55
SMD	**76.39**	**74.76**	**71.59**	**71.62**	**73.94**	**74.84**

Table 3: **CIFAR-100 Top-1 *accuracy* (%) under supervised distillation framework,** when the student and teacher share similar network architecture.

margin. CompR relies heavily on a huge negative sample queue with a length of 128,000. When the size of the negative sample queue decreases, their performance drops rapidly [1]. It is worth noting that our method can outperform CompR by directly optimizing real-time instance features with online hard mining in a 256-size batch. Both SEED and CompR are derived from the idea of latent negative sampling [33]. They treat all non-target unlabeled samples as negative, which misclassifies some positive samples. Our SMD mines positive and negative pairs from unlabeled samples, which is a good solution to this problem.

4.3 Applicability On Supervised Distillation Framework

To further highlight the superiority of SMD, we deploy our method in a supervised distillation framework, the baseline of which is usually a supervised classification problem. In the distillation part, the state-of-the-art supervised distillation methods usually use label information for knowledge transfer. We apply SMD to the supervised distillation framework without any modification. The only difference between the supervised distillation and the unsupervised distillation framework is that we must add a fully connected layer after the backbone of the student network and a classification loss will be trained together with our Eq.(7). Note that the distillation part of our SMD does not use any label information.

We evaluate our SMD on CIFAR-100 following the training procedure of CRD [29] in all the experiments for fair comparisons. All the models are trained for 240 epochs by SGD, and the learning rate drops by 0.1 after 150, 180, and 210 epochs. We set the weight decay to 5e-4, the batch size N to 64, and the momentum to 0.9. The initial learning rate is 0.05 for all models except MobileNet and ShuffleNet, where it is 0.01. τ is set to 0.04.

Table 3 and Table 4 list different teacher-student pairs to verify that our method is robust. We also listed some classic methods (KD [18] and FitNet [26]) and recently proposed methods with better performance [5,29,13,30,24].

	ResNet-50 MobileNetV2	ResNet-50 vgg8	resnet32x4 ShuffleNetV1	resnet32x4 ShuffleNetV2	WRN-40-2 ShuffleNetV1
Teacher	79.34	79.34	79.42	79.42	75.61
Student	64.60	70.36	70.50	71.82	70.50
KD[18]	67.35	73.81	74.07	74.45	74.83
FitNet[26]	63.16	70.69	73.59	73.54	73.73
SP[30]	68.08	73.34	73.48	74.56	74.52
RKD[24]	64.43	71.50	72.28	73.21	72.21
KDCL[13]	67.64	73.03	74.32	75.35	74.79
CRD[29]	69.11	74.30	75.11	75.65	76.05
WCoRD[5]	70.45	74.86	75.40	75.96	76.32
SMD	**70.76**	**74.95**	**76.21**	**76.82**	**76.89**

Table 4: **CIFAR-100 Top-1 *accuracy* (%) under supervised distillation framework,** for transfer across very different teacher and student architectures.

For WCoRD and KDCL, we use the results that are reported in their original paper. For all other methods, we use author-provided or author-verified code from the CRD repository. We can observe that our SMD can consistently outperform all other distillation methods with a large margin, including the recent state-of-the-art CRD and WCoRD. Note that most of these methods require the label information in the distillation process to mine the relationships between samples. For example, CRD heavily relies on accurate negative samples provided by annotated labels. Our SMD does not require any label information in the distillation process and effectively completes knowledge transfer in a self-supervised manner. In particular, our students outperform the teachers in some cases (WRN-40-2 to WRN-16-2, vgg13 to vgg8, and WRN-40-2 to ShuffleNetV1). This benefits from "cut-off at zero" strategy in Eq.(6). The knowledge learned in the teacher model may be redundant, so the teacher is not the optimal solution. In the distillation task, the student uses the optimization directions provided by the teacher, combined with the directions of the student's gradients, jointly finding an optimal solution. "cut-off at zero" retains this optimization state, rather than letting students continue optimizing towards the teacher.

4.4 Analysis

In this section, all experiments are conducted on two teacher-student combinations under the supervised distillation framework, MobileNetV2 supervised by ResNet-50 and resnet20 supervised by resnet56. We first comprehensively verify the correctness of the positive and negative sets obtained by Eq.(1). Following that, we show the necessity of the weighting factors, hard mining and L_2 loss.

Difference-Guided Positive and Negative Mining. To verify the accuracy of the difference-guided positive and negative mining, we list the changes of the total accuracy with the number of iterations in Table 5. The overall accuracy increases rapidly as the number of iterations increases, and soon reaches 99%. This verifies that our method can make a preliminary division. The accuracy

Epoch	Accuracy(%)	TP	TN	FP	FN
1 (10 iter)	2.25	92	0	4004	0
1 (30 iter)	76.73	102	3041	953	0
1 (100 iter)	99.37	104	3966	26	0
2	99.53	108	3969	17	2
150	99.17	81	3981	1	33
240	99.12	74	3986	0	36

Table 5: **Statistical information for positive and negative mining.** Total Accuracy (%), TP, TN, FP, and FN for MobileNetV2 with batch=64.

	SMD				SMD *without* weighting factors			
Epoch	TP	TN	FP	FN	TP	TN	FP	FN
1 (30 iter)	5	63	59	1	0	6	6	0
1 (100 iter)	49	64	15	0	9	64	55	0
2	59	64	5	0	36	64	28	0
150	62	64	2	0	14	64	50	0
240	64	64	0	0	11	64	53	0

Table 6: **Statistical information in the mined hardest negatives/positives** with and without weighting factors. TP, TN, FP, and FN for MobileNetV2 with batch=64.

seems to be satisfactory, but the number of positive and negative sample pairs is unbalanced since every batch is obtained by random sampling. To comprehensively evaluate the results, we count the True Positive (TP), True Negative (TN), False Positive (FP), and False Negative (FN) in a batch during the training. In the first few iterations, most of the samples are classified as positive pairs because the embedding spaces of teachers and students are not aligned and the *teacher-to-student metric difference* is large. After the two spaces are aligned, our method of obtaining positive and negative sets yields almost accurate results in the early epoch. This verifies that the *teacher-to-student metric difference* can indeed serve as a decision boundary that can be used to divide positive and negative samples. In the later stage, this difference will become very small because the student embedding has been trained toward the teacher embedding and our supervision mining method tends to divide more pairs as negative. Fortunately, these incorrectly divided pairs will not affect our performance due to the hard mining strategy and the weighting factors in Eq.(6). From another perspective, knowledge distillation aims to lower the gap between student and teacher spaces. The gradual shrinking of the decision boundary and positive examples will ensure that the embedding space learned by the student is similar to that of the teacher. In the limit, the two spaces will be perfectly aligned when the decision boundary is zero. This gradual way of adjusting the positive and negative pairs is conducive to SSL distillation.

Importance of Hard Mining. Hard mining is introduced based on the assumption that the hardest case plays a key role in optimization [31], and it also prevents all pairs from performing cost-prohibitive optimization. In addition to these, we find that hard mining can weaken the impact of incorrectly divided

pairs. Table 6 lists the statistical results of our SMD in the mined hardest negatives/positives. In the later training stage, almost all the hardest samples are correct. Combined with the data in Table 5, we reasonably infer that although some FNs are positive, for the well-trained teacher, these FNs are further away from the anchor compared to the hardest negative samples. Therefore, these FNs do not affect our optimization after hard mining, nor do FPs.

Weighting Factors. We conducted a comparison using our SMD and SMD without a_i^p and a_i^n. The Top-1 result for MobileNetV2 decreases from 70.76 to 63.48. For resnet20, from 71.56 to 69.05. Adding such a weighting factor brings a substantial improvement. Table 6 also lists the statistical results in the mined hardest negatives/positives of SMD without weighting factors. At the 30th iteration, only 6 out of 64 samples find the corresponding positive and negative sample pairs. Since the student network has not converged, the remaining 58 samples cannot find the corresponding negative samples and fail to join the distillation (all other samples are classified as positive). From 150 to 240 epoch, the TP even gradually decreases. Hard mining allows distillation to focus on optimizing difficult samples, but it also can exacerbate errors due to samples with inevitable wrong division, resulting in even worse performance than baseline. By introducing these two dynamic weighting factors, the impact of incorrectly divided pairs is greatly mitigated. This brilliant design can also provide a moderate optimization target for the correct sample pairs, thereby significantly improving the distillation results.

Necessity of L_2 loss. When the L_2 loss was completely removed, the Top-1 accuracy for MobileNetV2 decreases to 65.65, but only a slight change for resnet20, to 71.64. We can infer that L_2 loss plays a key role although it only acts on the first two epochs. When the structural differences between the student and the teacher are too large and the embeddings of the student network fail to find the embedding space of the teacher network, SMD cannot converge without L_2 loss. Note that the absence of L_2 loss does not degrade the performance when the student network can easily find the embedding space of the teacher network. These observations suggest that L_2 loss can help align the embedding space, but the final performance of SMD will not benefit from it.

5 Conclusion

In this work, we propose SMD to effectively improve the lightweight SSL model via distilling the metric information. Our solution utilizes the relationship between samples from unlabeled data in the knowledge distillation framework, which has not yet been investigated. Moreover, we also incorporate a dynamic weighting strategy to handle pairwise uncertainty adaptively. Extensive experiments demonstrate that our proposed SMD achieves state-of-the-art performance on various benchmarks of lightweight models.

Acknowledgment This work is supported by National Natural Science Foundation of China (62176188), Key Research and Development Program of Hubei Province (2021BAA187), Special Fund of Hubei Luojia Laboratory (220100015).

References

1. Abbasi Koohpayegani, S., Tejankar, A., Pirsiavash, H.: Compress: Self-supervised learning by compressing representations. NeurIPS **33**, 12980–12992 (2020) 1, 2, 3, 4, 10, 11
2. Ahn, S., Hu, S.X., Damianou, A., Lawrence, N.D., Dai, Z.: Variational information distillation for knowledge transfer. In: CVPR. pp. 9163–9171 (2019) 4
3. Buciluǎ, C., Caruana, R., Niculescu-Mizil, A.: Model compression. In: ACM SIGKDD. pp. 535–541 (2006) 3
4. Caron, M., Misra, I., Mairal, J., Goyal, P., Bojanowski, P., Joulin, A.: Unsupervised learning of visual features by contrasting cluster assignments. arXiv preprint arXiv:2006.09882 (2020) 1
5. Chen, L., Wang, D., Gan, Z., Liu, J., Henao, R., Carin, L.: Wasserstein contrastive representation distillation. In: CVPR. pp. 16296–16305 (2021) 4, 11, 12
6. Chen, T., Kornblith, S., Norouzi, M., Hinton, G.: A simple framework for contrastive learning of visual representations. In: ICML. pp. 1597–1607. PMLR (2020) 1, 10
7. Chen, X., He, K.: Exploring simple siamese representation learning. In: CVPR. pp. 15750–15758 (2021) 1, 8, 9
8. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: CVPR. pp. 248–255. Ieee (2009) 9
9. Fang, Z., Wang, J., Wang, L., Zhang, L., Yang, Y., Liu, Z.: Seed: Self-supervised distillation for visual representation. arXiv preprint arXiv:2101.04731 (2021) 2, 3, 4, 6, 10
10. Gao, Y., Zhuang, J.X., Li, K., Cheng, H., Guo, X., Huang, F., Ji, R., Sun, X.: Disco: Remedy self-supervised learning on lightweight models with distilled contrastive learning. arXiv preprint arXiv:2104.09124 (2021) 2, 3, 4, 9, 10
11. Grill, J.B., Strub, F., Altché, F., Tallec, C., Richemond, P.H., Buchatskaya, E., Doersch, C., Pires, B.A., Guo, Z.D., Azar, M.G., et al.: Bootstrap your own latent: A new approach to self-supervised learning. arXiv preprint arXiv:2006.07733 (2020) 1
12. Gu, J., Liu, W., Tian, Y.: Simple distillation baselines for improving small self-supervised models. arXiv preprint arXiv:2106.11304 (2021) 2, 3, 4, 9
13. Guo, Q., Wang, X., Wu, Y., Yu, Z., Liang, D., Hu, X., Luo, P.: Online knowledge distillation via collaborative learning. In: CVPR. pp. 11020–11029 (2020) 11, 12
14. Hadsell, R., Chopra, S., LeCun, Y.: Dimensionality reduction by learning an invariant mapping. In: 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06). vol. 2, pp. 1735–1742. IEEE (2006) 4
15. He, K., Fan, H., Wu, Y., Xie, S., Girshick, R.: Momentum contrast for unsupervised visual representation learning. In: CVPR. pp. 9729–9738 (2020) 1, 2, 4, 6, 10
16. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR. pp. 770–778 (2016) 8
17. Heo, B., Lee, M., Yun, S., Choi, J.Y.: Knowledge transfer via distillation of activation boundaries formed by hidden neurons. In: AAAI. vol. 33, pp. 3779–3787 (2019) 4
18. Hinton, G., Vinyals, O., Dean, J.: Distilling the knowledge in a neural network. arXiv preprint arXiv:1503.02531 (2015) 2, 4, 6, 11, 12
19. Howard, A., Sandler, M., Chu, G., Chen, L.C., Chen, B., Tan, M., Wang, W., Zhu, Y., Pang, R., Vasudevan, V., et al.: Searching for mobilenetv3. In: ICCV. pp. 1314–1324 (2019) 2, 9

20. Huang, Z., Wang, N.: Like what you like: Knowledge distill via neuron selectivity transfer. arXiv preprint arXiv:1707.01219 (2017) 4
21. Krizhevsky, A., Hinton, G., et al.: Learning multiple layers of features from tiny images (2009) 8
22. Ma, N., Zhang, X., Zheng, H.T., Sun, J.: Shufflenet v2: Practical guidelines for efficient cnn architecture design. In: ECCV. pp. 116–131 (2018) 8
23. Oord, A.v.d., Li, Y., Vinyals, O.: Representation learning with contrastive predictive coding. arXiv preprint arXiv:1807.03748 (2018) 3, 6
24. Park, W., Kim, D., Lu, Y., Cho, M.: Relational knowledge distillation. In: CVPR. pp. 3967–3976 (2019) 11, 12
25. Passalis, N., Tefas, A.: Learning deep representations with probabilistic knowledge transfer. In: ECCV. pp. 268–284 (2018) 4
26. Romero, A., Ballas, N., Kahou, S.E., Chassang, A., Gatta, C., Bengio, Y.: Fitnets: Hints for thin deep nets. arXiv preprint arXiv:1412.6550 (2014) 4, 11, 12
27. Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., Chen, L.C.: Mobilenetv2: Inverted residuals and linear bottlenecks. In: CVPR. pp. 4510–4520 (2018) 8
28. Tan, M., Le, Q.: Efficientnet: Rethinking model scaling for convolutional neural networks. In: ICML. pp. 6105–6114. PMLR (2019) 9
29. Tian, Y., Krishnan, D., Isola, P.: Contrastive representation distillation. In: ICLR (2019) 2, 4, 11, 12
30. Tung, F., Mori, G.: Similarity-preserving knowledge distillation. In: ICCV. pp. 1365–1374 (2019) 11, 12
31. Wang, F., Liu, H.: Understanding the behaviour of contrastive loss. In: CVPR. pp. 2495–2504 (2021) 13
32. Weinberger, K.Q., Saul, L.K.: Distance metric learning for large margin nearest neighbor classification. Journal of machine learning research **10**(2) (2009) 4
33. Wu, Z., Xiong, Y., Yu, S.X., Lin, D.: Unsupervised feature learning via non-parametric instance discrimination. In: CVPR. pp. 3733–3742 (2018) 6, 11
34. Ye, M., Li, H., Du, B., Shen, J., Shao, L., Hoi, S.C.: Collaborative refining for person re-identification with label noise. IEEE TIP **31**, 379–391 (2021) 5
35. Ye, M., Shen, J., Zhang, X., Yuen, P.C., Chang, S.F.: Augmentation invariant and instance spreading feature for softmax embedding. IEEE TPAMI **44**(02), 924–939 (2022) 1
36. You, Y., Gitman, I., Ginsburg, B.: Scaling sgd batch size to 32k for imagenet training. arXiv preprint arXiv:1708.03888 **6**, 12 (2017) 10
37. Zagoruyko, S., Komodakis, N.: Wide residual networks. arXiv preprint arXiv:1605.07146 (2016) 8
38. Zagoruyko, S., Komodakis, N.: Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer. In: ICLR (2017), https://arxiv.org/abs/1612.03928 4