Three Guidelines You Should Know for Universally Slimmable Self-Supervised Learning

Yun-Hao Cao[1], Peiqin Sun[2,*], Shuchang Zhou[2]

[1]State Key Laboratory for Novel Software Technology, Nanjing University
[2]MEGVII Technology

caoyh@lamda.nju.edu.cn, {sunpeiqin, zsc}@megvii.com

Abstract

We propose universally slimmable self-supervised learning (dubbed as US3L) to achieve better accuracy-efficiency trade-offs for deploying self-supervised models across different devices. We observe that direct adaptation of self-supervised learning (SSL) to universally slimmable networks misbehaves as the training process frequently collapses. We then discover that temporal consistent guidance is the key to the success of SSL for universally slimmable networks, and we propose three guidelines for the loss design to ensure this temporal consistency from a unified gradient perspective. Moreover, we propose dynamic sampling and group regularization strategies to simultaneously improve training efficiency and accuracy. Our US3L method has been empirically validated on both convolutional neural networks and vision transformers. With only once training and one copy of weights, our method outperforms various state-of-the-art methods (individually trained or not) on benchmarks including recognition, object detection and instance segmentation. Our code is available at https://github.com/megvii-research/US3L-CVPR2023.

1. Introduction

Deep supervised learning has achieved great success in the last decade, but the drawback is that it relies heavily on a large set of annotated training data. Self-supervised learning (SSL) has gained popularity because of its ability to avoid the cost of annotating large-scale datasets. Since the emergence of contrastive learning [7], SSL has clearly gained momentum and several recent works [8, 14] have achieved comparable or even better performance than the supervised pretraining when transferring to downstream tasks. However, it remains challenging to deploy trained models for edge computing purposes, due to the limited memory, computation and storage capabilities of such devices.

Table 1. Comparisons between supervised classification and SimSiam under S-Net on CIFAR-100. The accuracy for SimSiam is under linear evaluation. '-' denotes the model collapses.

Type	Method	Accuracy (%)			
		1.0x	0.75x	0.5x	0.25x
Supervised	Individual	73.8	72.8	71.4	67.3
	S-Net [32]	71.9	71.7	70.8	66.2
	S-Net+Distill [31]	73.1	71.9	70.5	67.2
SimSiam [9]	Individual	65.2	64.0	60.6	51.2
	S-Net [32]	-	-	-	-
	S-Net+Distill [31]	46.9	46.9	46.7	45.3
	Ours	**65.5**	**65.3**	**63.2**	**59.7**

To facilitate deployment, several model compression techniques have been proposed, including lightweight architecture design [29], knowledge distillation [20], network pruning [15], and quantization [33]. Among them, structured network pruning [25] is directly supported and accelerated by most current hardware and therefore the most studied. However, most structured pruning methods require fine-tuning to obtain a sub-network with a specific sparsity, and a single trained model cannot achieve instant and adaptive accuracy-efficiency trade-offs across different devices. To address this problem in the context of supervised learning, the family of slimmable networks (S-Net) and universally slimmable networks (US-Net) [2, 22, 31, 32] were proposed, which can switch freely among different widths by training only once.

Driven by the success of slimmable networks, a question arises: Can we train a *self-supervised model* that can run at arbitrary width? A naïve solution is to replace the supervised loss with self-supervised loss based on the US-Net framework. However, we find that this solution doesn't work directly after empirical studies. Table 1 shows that the phenomenon in self-supervised scenarios is very different. The model directly collapses after applying the popular SSL method SimSiam [9] to slimmable networks [32]. Although using inplace distillation [31] for sub-networks prevents the model from collapsing, there is still a big gap between the

*Corresponding author.

results of S-Net+Distill and training each model individually for SimSiam. So why is the situation so different in SSL and how to further improve the performance (i.e., close the gap)?

In this paper, we present a unified perspective to explain the differences and propose corresponding measures to bridge the gap. From a unified gradient perspective, we find that the key is that the guidance to sub-networks should be consistent between iterations, and we analyze which components of SSL incur the temporal inconsistency problem and why US-Net works in supervised learning. Based on these theoretical analyses, we propose three guidelines for the loss design of US-Net training to ensure temporal consistency. As long as one of them is satisfied, US-Net can work well, no matter in supervised or self-supervised scenarios. Moreover, considering the characteristics of SSL and the deficiencies of US-Net, we propose dynamic sampling and group regularization to reduce the training overhead while improving accuracy. Our main contributions are:

- We discover significant differences between supervised and self-supervised learning when training US-Net. Based on these observations, we analyze and summarize three guidelines for the loss design of US-Net to ensure temporal consistency from a unified gradient perspective.
- We propose a dynamic sampling strategy to reduce the training cost without sacrificing accuracy, which eases coping with the large data volumes in SSL.
- We analyze how the training scheme of US-Net limits the model capacity and propose group regularization as a solution by giving different freedoms to different channels.
- We validate the effectiveness of our method on both CNNs and Vision Transformers (ViTs). Our method requires only once training and a single model, which can exceed the results of training each model individually, and is comparable to knowledge distillation from pretrained teachers.

2. Related Works

Self-supervised Learning. To avoid time-consuming and expensive data annotations, many self-supervised methods were proposed to learn visual representations from large-scale unlabeled images or videos [4, 26]. As the driving force of state-of-the-art SSL methods, contrastive learning methods greatly improve the performance of representation learning in recent years [30]. Contrastive learning is a discriminative approach that aims at pulling similar samples closer and pushing diverse samples far from each other. SimCLR [7] and MoCo [16] both employ a contrastive loss function InfoNCE [30], which requires negative samples. BYOL [14] and SimSiam [9] discard negative sampling in contrastive learning by using an asymmetrical design.

To improve the accuracy-efficiency trade-off for self-supervised models, many works have been proposed. Fang *et al.* [13] proposed self-supervised knowledge distillation (SEED) for SSL with lightweight models. However, models at different widths (sparsities) must be trained individually, which incurs significant computational and storage overhead and is unsustainable for large data volumes. Moreover, it requires a pretrained teacher model while ours does not. Recently, SSQL [3] proposes to pretrain quantization-friendly self-supervised models to facilitate downstream deployment. Concurrent work DATA [5] proposes a neural architecture search (NAS) approach specialized for SSL. In contrast, we focus on structured pruning and we provide a unified theoretical explanation for the loss design of once-for-all training.

Slimmable Networks. Slimmable networks [32] are widely studied because of their ability to execute at different widths, permitting instant and adaptive accuracy-efficiency trade-offs at runtime. Later, [31] proposes universally slimmable networks (US-Net), which extend slimmable networks to run at arbitrary width. Follow-up work OFA [2] extends the sampling space of sub-networks to depth and kernel size dimensions but it also inherits the loss design of US-Net by combining base loss and inplace distillation loss. [6] explores finding an optimal sub-model from a vision transformer [11]. They were all done, however, under the supervised learning paradigm, whereas our method is self-supervised.

3. Method

In this section, we begin with the notations and a brief review of previous works in Sec. 3.1. Then, we introduce our method in Sec. 3.2, which we call Universally Slimmable Self-Supervised Learning (dubbed as US3L), as shown in Fig. 1. Finally, we show that temporal consistency of guidance is critical to the success of US-Net training by analyzing the stability of gradient updates of both self-supervised and supervised losses, and we propose three guidelines for the loss design to ensure this consistency in Sec. 3.3.

3.1. Preliminary

In this subsection, we introduce two representative SSL methods SimSiam and SimCLR, as well as (universally) slimmable networks as preliminaries.

1) Self-supervised Losses. Let $\boldsymbol{x}_{i,1}$ and $\boldsymbol{x}_{i,2}$ denote two randomly augmented views from an input image $\boldsymbol{x}_i$. Let f denote an encoder network consisting of a backbone (e.g., ResNet [19]) and a projection MLP head [7].

SimSiam [9] maximizes the similarity between two augmentations of one image. A prediction MLP head [14], denoted as h, transforms the output of one view and matches it to the other view. The output vectors for $\boldsymbol{x}_{i,1}$ are denoted as $\boldsymbol{z}_{i,1} \triangleq f(\boldsymbol{x}_{i,1})$ and $\boldsymbol{p}_{i,1} \triangleq h(f(\boldsymbol{x}_{i,1}))$, and $\boldsymbol{z}_{i,2}$ and $\boldsymbol{p}_{i,2}$ are defined similarly. The negative cosine similarity is defined as $D(\boldsymbol{p}, \boldsymbol{z}) \triangleq -\frac{\boldsymbol{p}}{\|\boldsymbol{p}\|_2} \cdot \frac{\boldsymbol{z}}{\|\boldsymbol{z}\|_2}$ and we assume both $\boldsymbol{z}$ and $\boldsymbol{p}$ have been L_2-normalized for simplicity in subsequent discussions. Let $SG(\cdot)$ denote the stop-gradient operation.

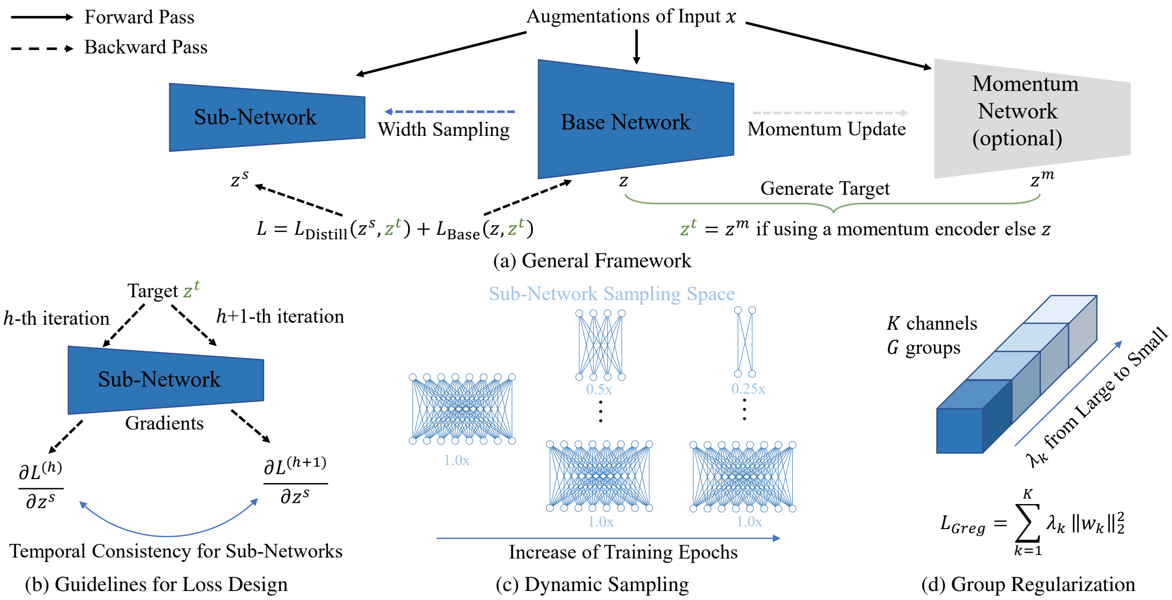


Figure 1. The proposed framework and our method for universally slimmable self-supervised learning.

Then, the loss function in SimSiam is:

$$L_{\text{MSE}} = \sum_i D(\boldsymbol{p}_{i,1}, SG(\boldsymbol{z}_{i,2})) + D(\boldsymbol{p}_{i,2}, SG(\boldsymbol{z}_{i,1})) \,. \quad (1)$$

SimCLR [7] and MoCo [16] contrast with negative samples using InfoNCE [30] loss:

$$L_{\text{NCE}} = -\sum_i \log \frac{e^{\boldsymbol{z}_{i,1}\cdot\boldsymbol{z}_{i,2}}}{e^{\boldsymbol{z}_{i,1}\cdot\boldsymbol{z}_{i,2}} + \sum_{j\neq i, v\in\{1,2\}} e^{\boldsymbol{z}_{i,1}\cdot\boldsymbol{z}_{j,v}}} \,, \quad (2)$$

where we omit the temperature parameter τ for simplicity.

2) Slimmable Networks. Slimmable networks [32] are a class of networks that can be executable at different scales. During training, only the smallest, the largest and a few randomly sampled networks are used to calculate the loss in each iteration, which is known as the *sandwich rule*. Further, inplace distillation [31] is introduced to improve performance, where the knowledge inside the largest network is transferred to sub-networks by using distillation loss.

3.2. The Proposed Method

The following three subsections describe the components of our US3L method (Algorithm 1).

3.2.1 Loss Design

The general framework of our method is depicted in Fig. 1a, in which the loss function is composed of base loss (for the base/largest network) and distillation loss (for sub-networks). By default, we use a momentum encoder to generate targets, InfoNCE as the base loss, and MSE as the distillation loss. Also, we show that we should use an auxiliary distillation head to mitigate the impact of the capacity difference between teacher and student. The overall loss function is

$$L = \underbrace{L_{\text{NCE}}}_{L_{\text{Base}}} \underbrace{- \sum_i \sum_s g(z_i^s) \cdot z_i^m}_{L_{\text{Distill}}} \,, \quad (3)$$

where $g(\cdot)$ is an auxiliary distillation MLP head, z^s and z^m are the output of the sub-network and momentum encoder, respectively. Notice that Eq. (3) is not the only option for the loss design and it can work well as long as it satisfies our guidelines (Fig. 1b), which will be discussed in Sec. 3.3.

3.2.2 Dynamic Sampling

It is worth noting that Yu *et al.* [32] sampled four switches in each iteration throughout training, which is very time-consuming for SSL training. Therefore, we design a dynamic sampling strategy to reduce the training overhead while improving performance (Fig. 1c). First, we argue that it is unnecessary to introduce the training of sub-networks at the beginning. We believe that a good and consistent teacher is essential for the learning of sub-networks [1], so we only need to train the base network at the start. Second, the training of sub-networks should be gradual. Specifically, the width of the smallest sub-network should be gradually reduced. By combining the two sampling strategies described above, we successfully reduce the sampling number s from 4 to 3 (theoretical minimum sampling number) without performance drop (see appendix for detailed analysis and results).

In our implementation, the training process is divided into two stages. In the first stage, only the largest network is trained (i.e., $s = 1$). In the second stage, we sample the largest, the smallest plus a random width ($s = 3$), and the width of the smallest model is gradually reduced. For example, the sampling width range in the second stage begins with [0.75,1.0], then [0.5,1.0], and finally [0.25,1.0].

3.2.3 Group Regularization

Given two channels k_1 and k_2 ($k_1 < k_2$), if k_2 is used in US-Net, then k_1 must also be used. In other words, the earlier channels are used more frequently than the later ones. Therefore, in the training of US-Net, the majority of the weights will be concentrated on the earlier channels to ensure the performance of sub-networks. However, such a weight distribution will limit the base model's capacity and thus affect its performance. To address this problem, we propose group regularization by giving more degrees of freedom (i.e., smaller regularization coefficients) to the later channels (Fig. 1d), so that their weights are more fully utilized. We divide the total K channels into G groups in order, with each group containing $K_G = \lfloor K/G \rfloor$ channels. Then we define:

$$L_{\text{GReg}} = \sum_{k=1}^{K} \lambda_k \|\mathbf{w}_k\|_2^2 \,, \tag{4}$$

$$\lambda_k = \lambda(1 - \lfloor k/K_G \rfloor \alpha) \,, \tag{5}$$

where $\mathbf{w}$ denotes the weight matrix and we set $G = 8$ and $\alpha = 0.05$ throughout this paper. Notice that when $\alpha = 0$, group regularization degenerates into the standard L_2 regularization. We also empirically demonstrate that group regularization is tailored for US-Net in the appendix.

3.3. Three Guidelines for Loss Design

The special feature of US-Net training is the introduction of sub-network training (i.e., L_{Distill}), so the training stability of the sub-networks is very important. In this paper, we find that the key is to ensure the temporal consistency of guidance for sub-networks. One image has different views in two adjacent iterations, which will produce different outputs because the model has not converged and is unstable. We hope that the gradients generated by different views of *the same image* will also be close between iterations (i.e., robust to changes and provide consistent guidance to sub-networks).

In the context of SSL, (1) and (2) can also be adapted for distillation and we use z^s and z^t to denote the output of the sub and base network, respectively. The losses are:

$$L_{\text{MSE-distill}} = -z_i^s \cdot z_i^t \,, \tag{6}$$

$$L_{\text{NCE-distill}} = -z_i^s \cdot z_i^t + \log \sum_k e^{z_i^s \cdot z_k^t} \,. \tag{7}$$

Algorithm 1 The proposed US3L method

Require: Define width range $R = [R_{\text{min}}, R_{\text{max}}]$x, for example, $R_{\text{min}} = 0.25$, $R_{\text{max}} = 1.0$.
for $h = 1, \dots, T_{iters}$ **do**
  Define period length $T_p = \lfloor T_{iters}/4 \rfloor$.
  Clear gradients, *optimizer.zero_grad()*.
  Run base network $z_1 = M(x_1)$, $z_2 = M(x_2)$.
  Compute base loss, $loss = L_{\text{Base}}(z_1, z_2) + L_{\text{GReg}}$.
  Detach label, $z_1^t = z_1.detach()$, $z_2^t = z_2.detach()$.
  if $t \leq T_p$ **then**
    Continue
  end if
  Dynamic adjust range, $R_{\text{min}} = 1 - 0.25\lfloor t/T_p \rfloor$.
  Randomly sample a width from R as *width samples*.
  Add the smallest width R_{min} to *width samples*.
  for *width* in *width samples* **do**
    Execute sub-network M' at width, and distillation head g, $z_1^s = g(M'(x_1))$, $z_2^s = g(M'(x_2))$.
    $loss \mathrel{+}= L_{\text{Distill}}(z_1^s, z_2^t) + L_{\text{Distill}}(z_2^s, z_1^t)$.
  end for
  Accumulate gradients, *loss.backward()*.
  Update weights, *optimizer.step()*.
end for

Lemma 3.1 *MSE is not robust to changes in the output, whereas NCE is stabilized by distances from other samples.*

Proof. For (6), the derivative can be derived as follows:

$$\frac{\partial L}{\partial z_i^s} = -z_i^t \,. \tag{8}$$

For (7), the derivative can be derived as follows:

$$\frac{\partial L}{\partial z_i^s} = -z_i^t + \sum_j \frac{e^{z_i^s \cdot z_j^t}}{\sum_k e^{z_i^s \cdot z_k^t}} z_j^t \triangleq -z_i^t + \sum_j P_j z_j^t \,. \tag{9}$$

Hence, we see that for MSE, the gradient only depends on the output z^t. This output will be very unstable in different iterations due to factors such as rapid model updates and image augmentations, resulting in temporal gradient instability. In contrast, NCE loss is also stabilized by the distance from other samples (corresponding to the extra $\sum_j P_j z_j^t$ term). □

To further illustrate Lemma 3.1, consider the following example (Fig. 2b,c). Assume that due to image augmentations, all outputs are transformed by the same rotation matrix w^θ from the h-th to the h+1-th iteration (Fig. 2a). The gradient difference between iterations for MSE is:

$$\frac{\partial L^{(h+1)}}{\partial z_i^s} - \frac{\partial L^{(h)}}{\partial z_i^s} = (I - w^\theta) z_i^t \,, \tag{10}$$

where I denotes the identity matrix. For InfoNCE, we have:

$$\frac{\partial L^{(h+1)}}{\partial z_i^s} - \frac{\partial L^{(h)}}{\partial z_i^s} = (I - w^\theta)(z_i^t - \sum_j P_j z_j^t) \,. \tag{11}$$

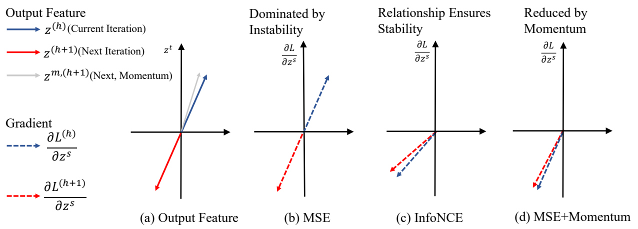


Figure 2. Illustration of feature changes and corresponding gradient changes under various settings. Best viewed in color.

If the output of the student is already aligned with the teacher, then we have $P_i \approx 1$ and thus it can be verified that $\|\frac{\partial L_{\text{NCE}}^{(h+1)}}{\partial z_i^s} - \frac{\partial L_{\text{NCE}}^{(h)}}{\partial z_i^s}\|_2 < \|\frac{\partial L_{\text{MSE}}^{(h+1)}}{\partial z_i^s} - \frac{\partial L_{\text{MSE}}^{(h)}}{\partial z_i^s}\|_2$. That is, NCE loss achieves better temporal stability than MSE for the learning of sub-networks.

Lemma 3.2 *Supervised cross entropy (CE) is also stabilized by relative distance and temporal consistency is preserved.*

Proof. Let y_i denote the label for x_i, C denote the number of classes, $\mathbf{w} \in \mathbb{R}^{d\times C}$ denote the weight matrix of the classification head (d is the feature dimension). Then:

$$L_{\text{CE}} = -\log \frac{e^{\mathbf{w}_{y_i}^T \boldsymbol{z}_i}}{\sum_{j=1}^{C} e^{\mathbf{w}_j^T \boldsymbol{z}_i}} = -\mathbf{w}_{y_i}^T \boldsymbol{z}_i + \log \sum_{j=1}^{C} e^{\mathbf{w}_j^T \boldsymbol{z}_i} .$$

The derivative is as follows:

$$\frac{\partial L_{\text{CE}}}{\partial \boldsymbol{z}_i} = -\mathbf{w}_{y_i} + \sum_{j=1}^{C} \frac{e^{\mathbf{w}_j^T \boldsymbol{z}_i}}{\sum_{k=1}^{C} e^{\mathbf{w}_k^T \boldsymbol{z}_i}} \mathbf{w}_j \triangleq -\mathbf{w}_{y_i} + \sum_j P_j \mathbf{w}_j .$$

The gradient is not only related to the target class weight $\mathbf{w}_{y_i}$ and the analysis is then similar to the NCE above. □

From Lemma 3.2 we can understand the huge difference between MSE-based SimSiam and CE-based supervised classification in Table 1. The key is that inconsistent outputs can make the temporal gradient updates for MSE very unstable.

Lemma 3.3 *A momentum teacher will better preserve temporal consistency by producing slowly updating outputs.* [1]

From the above analyses, we can summarize three guidelines below to ensure the temporal consistency of guidance:

1. The base loss is based on the relative distance to produce temporal consistent outputs of the base network.
2. The distillation loss is based on the relative distance to produce temporal consistent guidance for sub-networks.
3. A momentum teacher is used to produce stable guidance for sub-networks.

Experimental results in Sec. 4.5 further verify the effectiveness of our proposed three guidelines, and we will empirically find that **at least one of the three guidelines needs to be satisfied** to make it work for the US-Net framework.

[1] It is a known fact that a momentum teacher reduces the degree of output change and helps the model with more stable training [14].

4. Experimental Results

We introduce the implementation details in Sec. 4.1. We experiment with CNNs in Sec. 4.2 and ViTs in Sec. 4.3 on CIFAR-100 [21] and CIFAR-10 [21], respectively. Then, we experiment on ImageNet [28] (IN) in Sec. 4.4 and we evaluate the transfer performance of ImageNet pretrained models on downstream recognition, object detection, and instance segmentation benchmarks. Finally, we investigate the effects of different components in our method in Sec. 4.5.

4.1. Implementation Details

Datasets. The main experiments are conducted on three benchmark datasets, i.e., CIFAR-10, CIFAR-100 and ImageNet. We also conduct transfer experiments on 5 recognition benchmarks as well as 2 detection benchmarks Pascal VOC 07&12 [12] and COCO2017 [24].

Backbones. In addition to the commonly used ResNet-50 [19], we also adopt 2 smaller networks, i.e., ResNet-18 and MobileNetv2 [29]. Moreover, we evaluate our method on vision transformers [11]. Sometimes we abbreviate ResNet-18/50 to R-18/50 and MobileNetv2 to MBv2.

Training details. We use SGD for pretraining, with a batch size of 512 and a base lr=0.5. The learning rate has a cosine decay schedule. The weight decay is 0.0001 and the SGD momentum is 0.9. We pretrain for 400 epochs on CIFAR-100 and 100 epochs on ImageNet unless otherwise specified.

4.2. Experiments on CIFAR

We compare our method with state-of-the-art SSL methods BYOL [14], SimSiam [9] and SimCLR [7], and *individually* pretrain for them to obtain models at different channel widths. Notice that it is not a fair comparison with our method because they must pretrain different models for different widths (i.e., need 9 pretrained models for 9 widths), whereas ours is *trained only once with one copy of weights*. To better illustrate the effectiveness of our method, we also compare it with one strong baseline SEED [13]. In addition to training each model separately, SEED also requires an additional pretrained teacher model. The original implementation of SEED uses InfoNCE-based distillation loss and we add one more variant SEED-MSE (use MSE distillation loss). We also directly adapt different SSL methods to US-Net [31] for comparison. We report the linear evaluation accuracy for pretrained models of all methods under the same setting.

As shown in Table 2, our method achieves higher accuracy consistently than baseline methods. Take R-18 as an example, our method achieves **2.2%** and **6.1%** higher accuracy than BYOL at widths of 1x and 0.25x, respectively, with less training cost. When compared with the strong baseline SEED, our method even performs better in most cases, with much less training cost and additional dependencies. Also notice that the SEED-MSE variant performs even better than

Table 2. Main results on CIFAR-100. '-' denotes the model collapses. n denotes the number of sub-models and $n = 9$ in this table. T denotes the cost of training one model on CIFAR-100 for 400 epochs and we do not consider the effect of model size here, because models of different sizes are encountered in each method. The best two results are **bolded** and <u>underlined</u>, respectively.

Backbone	Method	Once Training	Pretrained Teacher	Training Cost	Linear Accuracy (%)								
					1.0x	0.9x	0.8x	0.7x	0.6x	0.5x	0.4x	0.3x	0.25x
ResNet-18	SimCLR [7]	×	×	nT	66.5	65.4	64.7	63.7	62.6	61.0	59.0	56.1	53.6
	SimSiam [9]	×	×	nT	66.5	65.4	64.6	63.5	62.6	60.0	58.3	54.9	52.4
	BYOL [14]	×	×	nT	66.8	66.0	65.6	65.3	63.0	62.1	59.5	56.0	54.3
	SEED [13]	×	BYOL R-50	nT	67.3	66.6	65.8	65.2	64.8	63.5	62.2	60.1	58.5
	SEED-MSE	×	BYOL R-18	nT	67.5	67.2	66.5	66.0	65.9	64.8	<u>64.0</u>	<u>62.4</u>	60.1
	SEED-MSE	×	BYOL R-50	nT	67.5	66.8	66.7	66.0	65.4	<u>64.9</u>	63.6	61.3	60.1
	US [31]+SimCLR	✓	×	$4T$	65.5	64.9	63.8	63.6	62.7	61.8	60.2	58.2	57.4
	US [31]+SimSiam	✓	×	$4T$	57.5	57.4	57.3	57.0	56.3	55.4	54.5	53.1	52.4
	Ours	✓	×	$\mathbf{2.5T}$	<u>69.0</u>	<u>68.2</u>	<u>68.0</u>	<u>66.9</u>	<u>66.1</u>	64.7	62.6	60.9	<u>60.4</u>
	Ours (800ep)	✓	×	$5T$	**70.1**	**69.3**	**69.0**	**68.7**	**67.3**	**66.4**	**64.2**	**63.1**	**62.3**
ResNet-50	BYOL [14]	×	×	nT	67.0	66.7	66.5	66.3	66.0	64.9	63.8	62.1	61.2
	SEED [13]	×	BYOL R-50	nT	70.3	69.8	69.6	69.4	69.0	68.2	67.2	65.6	65.1
	SEED-MSE	×	BYOL R-50	nT	69.4	69.0	68.5	69.1	68.4	68.1	67.3	66.9	66.4
	US [31]+SimCLR	✓	×	$4T$	70.1	69.9	69.7	69.3	68.7	68.2	67.5	66.0	65.5
	US [31]+SimSiam	✓	×	$4T$	54.7	54.6	54.7	54.7	54.7	54.8	54.6	54.3	54.0
	Ours	✓	×	$\mathbf{2.5T}$	<u>72.6</u>	<u>72.0</u>	<u>71.5</u>	<u>71.2</u>	<u>70.6</u>	<u>70.2</u>	<u>68.6</u>	<u>67.7</u>	<u>67.4</u>
	Ours (800ep)	✓	×	$5T$	**73.0**	**72.5**	**71.9**	**71.6**	**71.1**	**70.8**	**69.1**	**68.0**	**67.6**
MobileNetv2	BYOL [14]	×	×	nT	61.2	60.7	60.5	60.2	59.9	58.7	57.3	54.6	51.9
	SEED-MSE	×	BYOL R-50	nT	**68.6**	**68.9**	**67.6**	**67.3**	**67.4**	**66.3**	**65.5**	**64.0**	**62.6**
	SEED-MSE	×	BYOL MBv2	nT	63.8	63.5	63.8	<u>63.6</u>	<u>63.6</u>	<u>63.3</u>	<u>62.7</u>	<u>62.1</u>	<u>59.8</u>
	US [31]+SimCLR	✓	×	$4T$	56.2	56.0	55.3	55.0	54.8	54.3	54.0	53.2	52.2
	US [31]+SimSiam	✓	×	$4T$	-	-	-	-	-	-	-	-	-
	Ours	✓	×	$\mathbf{2.5T}$	<u>65.7</u>	<u>65.1</u>	<u>64.2</u>	<u>63.6</u>	63.4	62.2	61.5	60.7	59.3

the original SEED, especially for small subnets, which is consistent with our analysis in Sec. 3.3. That is, when we have a pretrained teacher which can already generate consistent targets, it is sufficient to use MSE for distillation. We can observe similar trends and improvements for R-50. For MobileNetv2, our method outperforms individually trained BYOL and SEED-MSE (when using pretrained MBv2 as the teacher). However, when we use a better teacher (e.g., R-50), our method is inferior to SEED (the gap is within 3 points). Also, our method outperforms the US-Net baseline at all widths for all three backbones (the role of each component will be discussed in Sec. 4.5). Moreover, we can see that our method greatly reduces the training cost. The training time for individually-trained methods is proportional to the number of sub-networks n that need to be used while ours is not affected by n. When compared with the original US-Net, we reduced the expected number of sampling in each iteration from 4 to 2.5 (see appendix for more analyses).

In conclusion, our method outperforms various individually trained SSL algorithms and even achieves comparable accuracy with knowledge distillation, by only training once.

4.3. Experiments with Vision Transformer

To further demonstrate the efficacy of our method, we evaluate our method on vision transformers [11] (ViTs) and adopt the popular MoCov3 [10] as our baseline method. We employ the official code and experiment on CIFAR-10. To the best of our knowledge, we are the first to study slimmable self-supervised ViTs and we directly reduce the embedding dimension in all layers. As shown in Table 3, our method significantly outperforms individually trained MoCov3. Also, our method surpasses US+MoCov3 (adapt MoCov3 to US-Net) at all widths despite using less training cost. In conclusion, the results show that our method still works even for complex architectures such as vision transformers.

Table 3. Linear evaluation results for ViT on CIFAR-10.

Backbone	Method	Once Training	Linear Accuracy (%)			
			1.0x	0.75x	0.5x	0.25x
ViT-Tiny	MoCov3 [10]	×	82.6	79.5	75.8	68.0
	US+MoCov3	✓	79.8	79.4	77.6	76.4
	Ours	✓	**86.0**	**84.7**	**83.3**	**80.2**
ViT-Small	MoCov3 [10]	×	88.0	86.8	83.0	75.5
	US+MoCov3	✓	88.2	87.5	86.3	84.9
	Ours	✓	**90.3**	**89.7**	**88.7**	**85.5**

4.4. ImageNet and Transferring Experiments

In this subsection, we do unsupervised pretraining on the large-scale ImageNet training set without using labels. The linear evaluation results on ImageNet are shown in Table 4. Also, we evaluate the transfer ability of the learned representations on ImageNet later. We train one model for our method and 4 separate models for BYOL, each of which is trained for 100 epochs. As shown in Table 4, our US3L achieves higher accuracy than BYOL at all widths and our advantages become greater as the width shrinks.

We investigate the downstream object detection performance on Pascal VOC07&12 in Fig. 3 and COCO2017 in Table 5. The detector is Faster R-CNN [27] for VOC and Mask R-CNN [18] for COCO (both with FPN [23] back-

Table 4. Linear evaluation results on ImageNet.

Backbone	Method	Once Training	Linear Accuracy (%) 1.0x	0.75x	0.5x	0.25x
ResNet-18	BYOL	×	54.0	53.7	47.4	34.9
	US+BYOL	✓	55.9	53.1	48.0	40.6
	Ours	✓	**56.9**	**54.5**	**48.7**	**40.7**
ResNet-50	BYOL	×	68.1	66.3	61.2	50.9
	US+BYOL	✓	64.7	64.3	62.6	57.1
	Ours	✓	**68.4**	**66.7**	**63.4**	**57.7**

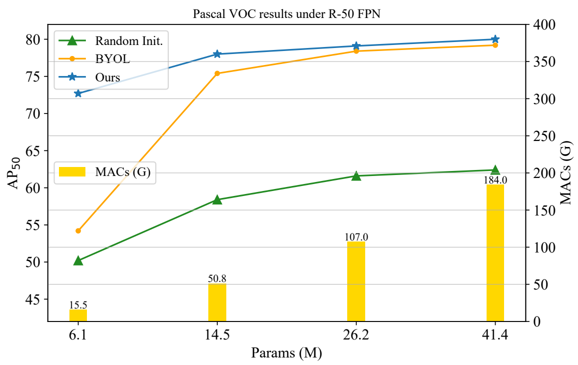


Figure 3. Transfer results on Pascal VOC 07&12 under R50-FPN.

Table 5. Transfer results on COCO2017 object detection& instance segmentation under R-50 FPN.

Width	Method	AP^{bb}	AP^{bb}_{50}	AP^{bb}_{75}	AP^{mk}	AP^{mk}_{50}	AP^{mk}_{75}
1.0x	BYOL	37.9	57.8	40.9	33.2	54.3	35.0
	Ours	**38.3**	**58.0**	**41.2**	**33.6**	**54.6**	**35.3**
0.75x	BYOL	35.7	55.3	38.6	32.4	52.4	34.5
	Ours	**36.2**	**55.8**	**39.0**	**32.8**	**52.9**	**35.0**
0.5x	BYOL	32.6	51.5	35.2	29.9	48.7	31.7
	Ours	**33.5**	**52.7**	**35.8**	**30.6**	**49.8**	**32.3**
0.25x	BYOL	26.0	43.5	27.0	24.2	40.8	25.3
	Ours	**27.5**	**45.0**	**29.3**	**25.5**	**42.4**	**26.9**

bone), following [3]. Fig. 3 shows that our method outperforms BYOL on Pascal VOC at width of 1.0x. Also, as we decrease the width, our advantages over the baseline counterpart BYOL will be further expanded: up to **+2.6** and **+18.5** AP_{50} at width of 0.5x and 0.25x, respectively. We can reach similar conclusions on COCO2017 from Table 5. Although our method achieves comparable accuracy to BYOL at 1.0x on COCO, we achieve **+0.9** and **+1.5** AP^{bb} points higher at 0.5x and 0.25x, respectively. Note that our improvements on COCO are not as large as that on VOC. It is because the amount of training data in COCO is large enough to close the gap between different pretrained models, as noted in [17].

We also transfer the ImageNet learned representations to 5 downstream recognition benchmarks in Table 6. As seen, our method improves a lot on all recognition benchmarks (except for pets) under linear evaluation.

4.5. Ablation Studies

In this subsection, we demonstrate the effectiveness of the proposed three guidelines as well as the proposed strategies.

Table 6. Transfer results on recognition benchmarks under linear evaluation. 'C-10/100' denotes 'CIFAR-10/100'.

Net	Width	Params	MACS	Method	Linear Accuracy (%) C-10	C-100	Flowers	Pets	Dtd
R-50	1.0x	22.56M	4.11G	BYOL	**87.1**	60.6	81.0	**80.9**	70.7
				Ours	**87.1**	**61.5**	**90.6**	79.4	**72.6**
	0.75x	14.77M	2.34G	BYOL	83.6	52.8	82.9	74.2	66.3
				Ours	**84.4**	**56.9**	**89.7**	**78.0**	**71.1**
	0.5x	6.92M	1.06G	BYOL	80.6	52.0	74.8	75.0	65.9
				Ours	**81.6**	**52.8**	**88.1**	**76.8**	**68.8**
	0.25x	1.99M	0.28G	BYOL	75.9	46.2	75.4	64.7	61.4
				Ours	**78.9**	**49.9**	**84.4**	**74.0**	**64.9**

4.5.1 Effectiveness of The Three Guidelines

We investigate various combinations of training loss, distillation head and momentum target in Table 7. The 'Base Loss' column represents the loss function for the base (i.e., largest) model. The 'Distill Loss' column represents the distillation loss for sub-networks ('×' means sub-networks use the same loss as the base network without using distillation). The 'Auxiliary Distill Head' column indicates whether to use an additional head for distillation (i.e., $g(\cdot)$ in (3)). The 'Momentum Target' column indicates whether to maintain a momentum encoder of the base network, 'Base/Sub Network' column represents whether the base/sub networks use the output of the momentum encoder as the target. We have the following observations from Table 7:

- As aforementioned, the model will collapse if we use individual MSE loss for sub-networks in case 1 (i.e., SimSiam). This phenomenon is *completely different from the supervised case*, where each sub-network can be trained with cross entropy loss alone to achieve good results. We conjecture that the base and sub networks directly imitate their respective targets when using MSE, which will bring instability to the entire model, as analyzed in Sec. 3.3.
- Following US-Net, we use inplace distillation to guide sub-networks in case 2 and it solves the collapse problem. It is because now all sub-networks are aligned to the same target, which ensures cross-subnet consistency. Nevertheless, there is still a large gap compared with individually trained self-supervised networks (nearly 10 points).
- If we continue to use a momentum teacher, we will find consistent improvements in Table 7 (e.g., case 3 vs. case 2). When comparing case 7 and case 6, we find that it is better to align all networks to the same target, which confirms our analysis of guidance consistency again. Consistency should not only exist between iterations, but also across sub-networks (all sub-networks should use the same target).
- **Experimental results are in full agreement with the proposed three guidelines**. Lemma 3.1 states that InfoNCE can obtain a more stable gradient than MSE. Notice that the loss function consists of the base and distillation loss. (i) When the base loss uses MSE, the output of the base network will be unstable, so the sub-networks need to use InfoNCE loss for distillation to deal with this instability (case

Table 7. Ablation studies of the loss design under ResNet-18 on CIFAR-100. '-' denotes the model collapses.

Base Loss	Case	Distill Loss	Auxiliary Distill Head	Momentum Target Base Network	Momentum Target Sub Network	Linear Accuracy (%) 1.0x	0.9x	0.8x	0.7x	0.6x	0.5x	0.4x	0.3x	0.25x
MSE	1	×	×	×	×	-	-	-	-	-	-	-	-	-
	2	MSE	×	×	×	57.5	57.4	57.3	57.0	56.3	55.4	54.5	53.1	52.4
	3	MSE	×	✓	✓	64.7	64.7	64.5	64.3	**63.9**	62.6	**61.3**	59.7	**59.3**
	4	MSE	✓	✓	✓	**65.4**	**65.0**	**64.8**	**64.5**	63.8	**62.7**	61.1	**59.8**	58.9
	5	InfoNCE	×	×	×	62.3	62.3	62.3	62.2	61.8	60.6	58.9	57.6	57.2
	6	InfoNCE	×	×	✓	63.7	63.8	63.7	63.6	63.1	62.0	60.6	59.3	58.2
	7	InfoNCE	×	✓	✓	65.0	65.0	65.1	**65.0**	64.5	62.7	61.3	59.8	59.2
	8	InfoNCE	✓	✓	✓	**65.5**	**65.5**	**65.6**	**65.0**	**64.6**	**63.2**	**61.6**	**60.2**	**59.7**
InfoNCE	9	×	×	×	×	64.8	64.0	63.2	62.0	60.8	59.8	57.4	55.1	54.2
	10	MSE	×	×	×	65.0	64.4	63.1	62.3	61.9	60.3	58.3	57.1	56.6
	11	MSE	×	×	✓	65.8	65.0	64.4	63.4	62.7	61.8	59.8	58.5	57.6
	12	MSE	×	✓	✓	66.9	66.3	65.7	64.9	63.8	62.9	61.6	59.5	59.1
	13	MSE	✓	✓	✓	**67.7**	**67.2**	**66.5**	**66.0**	**65.1**	**64.3**	**62.5**	**60.5**	**59.6**
	14	InfoNCE	×	×	×	65.5	64.9	63.8	63.6	62.7	61.8	60.2	58.2	57.4
	15	InfoNCE	×	×	✓	64.7	64.5	64.0	63.6	62.3	61.4	59.8	58.4	57.9
	16	InfoNCE	×	✓	✓	66.0	65.4	64.8	64.3	63.8	62.4	61.1	59.8	58.7
	17	InfoNCE	✓	✓	✓	**67.4**	**66.0**	**66.1**	**65.6**	**64.7**	**64.0**	**62.2**	**60.2**	**59.5**

Table 8. Ablation studies of our strategies on CIFAR-100.

Backbone	Dynamic Sampling	Group Reg.	Linear Accuracy (%) 1.0x	0.8x	0.6x	0.5x	0.3x	0.25x
R-18	×	×	67.7	66.5	65.1	64.3	60.5	59.6
	✓	×	68.6	67.2	65.5	64.6	60.7	59.9
	×	✓	68.6	67.3	65.5	64.4	**60.9**	60.1
	✓	✓	**69.0**	**68.0**	**66.1**	**64.7**	**60.9**	**60.4**
R-50	×	×	71.0	70.6	70.0	69.1	67.2	66.8
	✓	×	71.8	71.1	70.2	69.3	67.3	67.2
	×	✓	71.9	71.1	70.0	69.6	**67.7**	**67.5**
	✓	✓	**72.6**	**71.5**	**70.6**	**70.2**	**67.7**	67.4
MBv2	×	×	62.9	62.0	61.5	60.4	59.6	58.7
	✓	×	64.7	63.3	62.3	61.7	**60.7**	59.2
	×	✓	64.0	63.2	62.1	61.4	60.2	59.0
	✓	✓	**65.7**	**64.2**	**63.4**	**62.2**	**60.7**	**59.3**

5 v.s. case 2). (ii) When the base loss uses InfoNCE, the base network can achieve better temporal stability. Hence, the sub-networks already get stable targets from the teacher and using MSE or InfoNCE for distillation (case 10&14) can achieve good results. In short, in the absence of a momentum teacher, at least one of base loss and distillation loss should use InfoNCE to ensure stability.

- The use of an auxiliary distillation head will result in consistent improvements when we compare the last two rows of each block in Table 7 (e.g., case 8 vs. case 7).

4.5.2 Effectiveness of Our Strategies

We study the effect of our proposed strategies in Table 8 and the baseline is the best practice in Table 7 (i.e., case 13). We can have the following conclusions from Table 8:

- Our dynamic sampling strategy can improve the accuracy of the base network as well as sub-networks significantly. Although our total number of iterations is less (without training sub-networks at the start), we can still guarantee the performance of small sub-networks. It is because now we get a better and more stable teacher and then distillation speeds up the convergence of sub-networks.

Table 9. Intersection of the sandwich rule and dynamic sampling on CIFAR-100 under ResNet-18.

Sandwich Rule	Dynamic Sampling	Linear Accuracy (%) 1.0x	0.8x	0.6x	0.5x	0.3x	0.25x
×	×	65.1	65.0	64.7	63.2	59.4	56.4
×	✓	67.4	**67.3**	**65.9**	**64.7**	59.9	58.7
✓	×	67.7	66.5	65.1	64.3	60.5	59.6
✓	✓	**68.6**	**67.3**	65.5	64.4	**60.9**	**60.1**

- Our group regularization can also significantly improve the accuracy of the largest model while improving the accuracy of sub-networks. The experimental results verify our analysis in Sec. 3.2.3 that our grouping strategy can alleviate the problem of limited model capacity in US-Net.

We also study the intersection of the sandwich rule and dynamic sampling in Table 9. As seen, our dynamic sampling strategy can be used alone or combined with the sandwich rule, which brings improved performance for both scenarios. The results further demonstrate the universality and effectiveness of our dynamic sampling.

5. Conclusions

In this paper, we proposed a method called US3L for training universally slimmable self-supervised models. We provided theoretical analyses about the loss design and proposed three guidelines to ensure temporal consistency for US-Net training. Moreover, we proposed dynamic sampling and group regularization to solve the problems of inefficient training and limited model capacity. Experiments on various benchmarks and architectures (both CNNs and ViTs) show that our method significantly outperforms various baselines. When transferring to various downstream tasks, our models exhibit significant advantages at different widths, with only once training and one copy of weights. In the future, we will try to compress in dimensions such as depth and kernel size and explores combinations with NAS methods.

References

[1] Lucas Beyer, Xiaohua Zhai, Amelie Royer, Larisa Markeeva, Rohan Anil, and Alexander Kolesnikov. Knowledge distillation: A good teacher is patient and consistent. In *The IEEE Conference on Computer Vision and Pattern Recognition*, pages 10925–10934, 2022. 3

[2] Han Cai, Chuang Gan, Tianzhe Wang, Zhekai Zhang, and Song Han. Once-for-all: Train one network and specialize it for efficient deployment. In *The International Conference on Learning Representations*, pages 1–14, 2020. 1, 2

[3] Yun-Hao Cao, Peiqin Sun, Yechang Huang, Jianxin Wu, and Shuchang Zhou. Synergistic self-supervised and quantization learning. In *The European Conference on Computer Vision*, volume 13690 of *LNCS*, page 587–604. Springer, 2022. 2, 7

[4] Mathilde Caron, Piotr Bojanowski, Armand Joulin, and Matthijs Douze. Deep clustering for unsupervised learning of visual features. In *The European Conference on Computer Vision*, volume 11218 of *LNCS*, pages 132–149. Springer, 2018. 2

[5] Qing Chang, Junran Peng, Lingxie Xie, Jiajun Sun, Haoran Yin, Qi Tian, and Zhaoxiang Zhang. DATA: Domain-aware and task-aware self-supervised learning. In *The IEEE Conference on Computer Vision and Pattern Recognition*, pages 9841–9850, 2022. 2

[6] Arnav Chavan, Zhiqiang Shen, Zhuang Liu, Zechun Liu, Kwang-Ting Cheng, and Eric Xing. Vision transformer slimming: Multi-dimension searching in continuous optimization space. In *The IEEE Conference on Computer Vision and Pattern Recognition*, pages 4931–4941, 2022. 2

[7] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *The International Conference on Machine Learning*, pages 1597–1607, 2020. 1, 2, 3, 5, 6

[8] Xinlei Chen, Haoqi Fan, Ross Girshick, and Kaiming He. Improved baselines with momentum contrastive learning. *arXiv preprint arXiv:2003.04297*, 2020. 1

[9] Xinlei Chen and Kaiming He. Exploring simple siamese representation learning. In *The IEEE Conference on Computer Vision and Pattern Recognition*, pages 15750–15758, 2021. 1, 2, 5, 6, 11

[10] Xinlei Chen, Saining Xie, and Kaiming He. An empirical study of training self-supervised vision transformers. In *The IEEE International Conference on Computer Vision*, pages 9640–9649, 2021. 6

[11] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, pages 1–12, 2021. 2, 5, 6

[12] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (VOC) challenge. *International Journal of Computer Vision*, 88(2):303–338, 2010. 5

[13] Zhiyuan Fang, Jianfeng Wang, Lijuan Wang, Lei Zhang, Yezhou Yang, and Zicheng Liu. SEED: Self-supervised distillation for visual representation. In *The International Conference on Learning Representations*, pages 1–12, 2021. 2, 5, 6

[14] Jean-Bastien Grill, Florian Strub, Florent Altche, Corentin Tallec, Pierre H.Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Daniel Guo, Mohammand Gheshlaghi Azar, Bial Piot, Koray Kavukcuoglu, Remi Munos, and Michal Valko. Boostrap your own latent: A new approach to self-supervised learning. In *Advances in neural information processing systems*, pages 21271–21284, 2020. 1, 2, 5, 6

[15] Song Han, Huizi Mao, and William J. Dally. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. In *The International Conference on Learning Representations*, pages 1–14, 2016. 1

[16] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *The IEEE Conference on Computer Vision and Pattern Recognition*, pages 9729–9738, 2020. 2, 3

[17] Kaiming He, Ross Girshick, and Piotr Dollár. Rethinking imagenet pre-training. In *The IEEE International Conference on Computer Vision*, pages 4918–4927, 2019. 7

[18] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask R-CNN. In *The IEEE International Conference on Computer Vision*, pages 2961–2969, 2017. 6

[19] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *The IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016. 2, 5

[20] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015. 1

[21] Alex Krizhevsky and Geoffrey E. Hinton. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009. 5

[22] Changlin Li, Guangrun Wang, Bing Wang, Xiaodan Liang, Zhihui Li, and Xiaojun Chang. Dynamic slimmable network. In *The IEEE Conference on Computer Vision and Pattern Recognition*, pages 8607–8617, 2021. 1

[23] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *The IEEE Conference on Computer Vision and Pattern Recognition*, pages 2177–2125, 2017. 6

[24] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft COCO: Common objects in context. In *The European Conference on Computer Vision*, volume 8693 of *LNCS*, pages 740–755. Springer, 2014. 5

[25] Jian-Hao Luo, Jianxin Wu, and Weiyao Lin. Thinet: A filter level pruning method for deep neural network compression. In *The IEEE International Conference on Computer Vision*, pages 5058–5066, 2017. 1

[26] Mehdi Noroozi and Paolo Favaro. Unsupervised learning of visual representations by solving jigsaw puzzles. In *The European Conference on Computer Vision*, volume 9910 of *LNCS*, pages 69–84. Springer, 2016. 2

[27] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015. 6

[28] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252, 2015. 5

[29] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. MobileNetV2: Inverted residuals and linear bottlenecks. In *The IEEE Conference on Computer Vision and Pattern Recognition*, pages 4510–4520, 2018. 1, 5

[30] Aarin van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018. 2, 3

[31] Jiahui Yu and Thomas Huang. Universally slimmable networks and improved training techniques. In *The IEEE International Conference on Computer Vision*, pages 1803–1811, 2019. 1, 2, 3, 5, 6, 12

[32] Jiahui Yu, Linjie Yang, Ning Xu, Jianchao Yang, and Thomas Huang. Slimmable Neural Networks. In *The International Conference on Learning Representations*, pages 1–12, 2019. 1, 2, 3

[33] Shuchang Zhou, Yuxin Wu, Zekun Ni, Xinyu Zhou, He Wen, and Yuheng Zou. Dorefa-net: Training low bitwidth convolutional neural networks with low bitwidth gradients. *arXiv preprint arXiv:1606.06160*, 2016. 1

A. Training Details

Datasets. The statistics of the recognition benchmarks used in our paper are shown in Table 10.

Table 10. Statistics of the recognition benchmarks used in the paper.

Datasets	# Category	# Training	# Testing
CIFAR-10	10	50,000	10,000
CIFAR-100	100	50,000	10,000
Flowers	102	2,040	6,149
Pets	37	3,680	3,669
DTD	47	3,760	1,880

Training details for SSL methods. Training details for SimCLR, BYOL, SimSiam and our US3L on CIFAR-100 are shown in Table 11.

Table 11. Training details for SimCLR, BYOL, SimSiam and our US3L on CIFAR-100 in Table 2, Table 7 and Table 8. τ denotes the temperature parameter, and m denotes the momentum coefficient for the momentum network.

Method	Settings								
	bs	lr	wd	epochs	optimizer	lr sche.	τ	m	dim
SimSiam	512	0.1	5e-4	400	SGD	cosine	-	-	2048
SimCLR	512	0.5	1e-4	400	SGD	cosine	0.5	-	2048
BYOL	512	0.1	5e-4	400	SGD	cosine	-	0.99	2048
Ours	512	0.5	1e-4	400	SGD	cosine	0.5	0.99	2048

Training details for linear evaluation and fine-tuning. For ImageNet linear evaluation, we follow the same settings in [9]. For linear evaluation on other datasets, we train for 100 epochs with lr initialized to 30.0, which is divided by 10 at the 60-th and 80-th epoch.

Source codes. We promise that all codes will be made publicly available upon acceptance of the paper.

B. More Results

B.1. Transfer Results for ResNet-18

We plot the downstream object detection performance on Pascal VOC07&12 for ResNet-18 FPN in Fig. 4. Moreover, we present the transfer results on downstream recognition benchmarks for ResNet-18 in Table 12. The results show that our method is also effective for ResNet-18 when transferring to downstream object detection and recognition tasks.

B.2. Ablation Studies of Dynamic Sampling

In this subsection, we conduct ablation studies of our dynamic sampling strategy and show how we successfully reduced the sampling number s from 4 to 3 by using dynamic sampling while improving accuracy. We also compute the expected total forward number for T iterations. Take our dynamic sampling as an example, we train the largest model only in the first $\frac{T}{4}$ iterations and sample three sub-networks

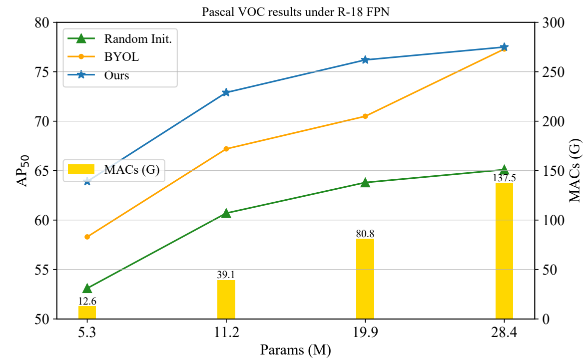


Figure 4. Transfer results on Pascal VOC 07&12 under R18-FPN.

Table 12. Transfer results on recognition benchmarks under linear evaluation. 'C-10/100' denotes 'CIFAR-10/100'.

Net	Width	Params	MACS	Method	Linear Accuracy (%)				
					C-10	C-100	Flowers	Pets	Dtd
R-18	1.0x	11.69M	1.82G	BYOL	76.6	48.6	83.0	71.1	64.8
				Ours	**77.9**	**52.6**	**84.9**	**71.2**	**65.2**
	0.75x	6.68M	1.05G	BYOL	76.4	48.1	82.6	**71.0**	**63.7**
				Ours	**76.7**	**49.0**	**83.5**	**71.0**	**63.7**
	0.5x	3.06M	0.49G	BYOL	74.6	46.9	81.4	**67.6**	61.7
				Ours	**75.2**	**47.4**	**82.0**	67.3	**62.6**
	0.25x	0.83M	0.14G	BYOL	67.0	41.2	75.5	57.6	56.4
				Ours	**67.8**	**41.4**	**77.0**	**60.6**	**56.6**

in the last $\frac{3T}{4}$ iterations, hence the expected total forward number is:

$$1 \times \frac{T}{4} + 3 \times \frac{3T}{4} = 2.5T\,. \tag{12}$$

As shown in Table 13, our dynamic sampling strategy achieves the best accuracy-efficiency trade-off. Notice that we also investigate the two components in our dynamic sampling strategy separately and we can clearly see that 'max first' reduces the training overhead (case 4) and 'gradually reduce' improves the accuracy (case 3).

B.3. Hyper-parameter Studies of G and α

In this subsection, we study the choice of hyper-parameters G and α in our group regularization (Eq. (5)) in Table 14. Notice that when $\alpha = 0$, group regularization is equivalent to the standard L_2 normalization (case 1). We used $G = 8$ and $\alpha = 0.05$ in the paper. We also present the max decay fraction $G \times \alpha$ (i.e., the decay rate for the last group). Table 14 shows that we can achieve the best results when $G \times \alpha = 40\%$ (case 1 $\sim$ case 5). Then we keep $G \times \alpha = 40\%$ and change G to 4 or 16. In terms of hyper-parameter G, we can see that $G = 8$ (case 3) outperforms $G = 4$ (case 6) and accuracy is saturated and will not continue to increase beyond 8 ($G = 16$, case 7). It is worth noting that if we set α to a negative value which goes against our motivation (case 9), we will no longer see performance gains for large sub-networks as before. The results here further validate the effectiveness of our method and the

Table 13. Ablation studies of the sampling strategy under ResNet-18 on CIFAR-100. T denotes the total number of iterations. 'Max first' denotes whether to train the largest network only in early epochs. 'Gradually reduce' denotes whether to gradually reduce the width of the smallest network.

Case	Sampling number s	Expected forward number	Dynamic Sampling		Linear Accuracy (%)								
			Max first	Gradually reduce	1.0x	0.9x	0.8x	0.7x	0.6x	0.5x	0.4x	0.3x	0.25x
1	4	$4T$	×	×	68.1	67.4	67.0	66.3	65.3	64.4	62.7	60.8	59.9
2	3	$3T$	×	×	67.7	67.2	66.5	66.0	65.1	64.3	62.5	60.5	59.6
3	3	$3T$	×	✓	68.5	67.9	**67.2**	66.4	65.3	64.5	62.6	**61.0**	**60.2**
4	3	**2.5**T	✓	×	67.8	67.6	66.8	66.2	65.3	64.5	63.0	60.6	59.8
5	3	**2.5**T	✓	✓	**68.6**	**68.1**	**67.2**	**66.6**	**65.5**	**64.6**	**62.8**	60.7	59.9

correctness of our analysis in the paper.

B.4. Group Regularization is Tailored for US-Net

In this subsection, we will demonstrate that our group regularization strategy is tailored for US-Net and our analysis in the paper is valid. We apply group regularization to common SSL methods which train each model individually. As shown in Table 15, the introduction of group regularization does not bring improvements for BYOL and SimCLR, which are individually trained. It shows that the group regularization is tailored for US-Net, and the improvement is due to our unique design rather than factors such as hyper-parameters.

B.5. Our US3L Can Run at Arbitrary Width

Note that we only reported the results of width at [1.0, 0.9, 0.8, 0.7, 0.6, 0.5, 0.4, 0.3, 0.25]x for CIFAR-100 and [1.0, 0.75, 0.5, 0.25]x for ImageNet due to limited space. Actually, the pretrained model of our US3L can run at any width within the predefined width range, by only training once. As a supplement, we present the results of more widths on CIFAR-100 in Table 16 and we can see that our pretrained model can achieve a good accuracy-efficiency trade-off.

B.6. Figures

As a supplement to Table 2 in the paper, we plot the results here to more intuitively see the advantages of our method. We compare with individually trained methods in Fig. 5 and compare with the US-Net baseline in Fig. 6.

B.7. Ablation Studies of Loss Design

We present more ablation results of loss design here in Table 17, as a supplement to Table 7 in the paper. We look more closely at the 'Asymmetric Distill Head' column, which indicates whether to use an additional head for distillation. Notice that there is already an asymmetrical head itself in MSE-based methods like SimSiam and BYOL. So 'Share' refers to sharing the asymmetrical head, and 'New' refers to distillation using a brand new head.

C. More Analysis

Lemma C.1 *$s = 3$ is the theoretical minimum number of samples for US-Net [31].*

Proof. First, from [31] we know sandwich rules: Performances at all widths are bounded by the performance of the model at the smallest and the largest width. In other words, optimizing the lower and upper bounds of performance can implicitly optimize all sub-networks in a US-Net. To optimize for arbitrary widths, we need at least one randomly sampled width per iteration, except for the largest and smallest sub-networks. In conclusion, $s = 3$ is the theoretical minimum number of samples for US-Net. ☐

Table 14. Hyper-parameter studies of G and α in group regularization under ResNet-18 on CIFAR-100.

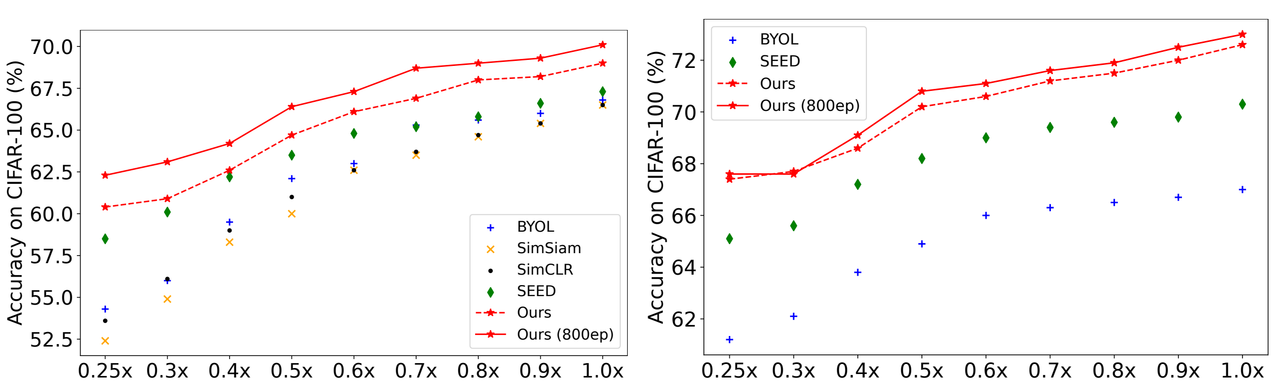
Case	Max decay fraction $G \times \alpha$	Group number G	Decay rate α	Linear Accuracy (%) 1.0x	0.9x	0.8x	0.7x	0.6x	0.5x	0.4x	0.3x	0.25x	Avg.	1.0x diff
1	0%	-	0	67.7	67.2	66.5	66.0	65.1	64.3	62.5	60.5	59.6	64.4	+0.0%
2	20%	8	0.025	68.0	67.4	66.3	66.0	65.2	64.0	62.6	61.0	60.3	64.5	+0.3%
3	40%	8	0.05	68.6	67.8	67.3	66.4	65.5	64.4	63.1	60.9	60.1	**64.9**	**+0.9%**
4	60%	8	0.075	68.2	67.6	66.9	66.1	65.3	64.2	62.7	61.0	60.2	64.7	+0.5%
5	80%	8	0.1	67.7	67.0	66.7	66.4	65.8	64.2	62.8	61.3	60.5	64.7	+0.0%
6	40%	4	0.1	68.0	67.4	66.6	66.1	65.1	63.8	63.0	61.6	60.6	64.7	+0.3%
7	40%	16	0.025	68.7	67.6	67.3	66.5	65.5	64.7	62.8	61.3	60.1	**64.9**	**+1.0%**
8	80%	16	0.05	67.9	67.3	66.7	66.3	65.5	64.2	63.1	61.1	60.5	64.7	+0.2%
9	-40%	8	-0.05	67.4	67.0	66.3	65.9	64.7	64.0	62.8	61.2	60.1	64.4	-0.3%

Table 15. Effect of group regularization in BYOL and SimCLR under ResNet-18 on CIFAR-100. Our group regularization strategy is tailored for US-Net.

Method	Model Type	Once Training	Group Regularization	Linear Accuracy (%) 1.0x	0.9x	0.8x	0.7x	0.6x	0.5x	0.4x	0.3x	0.25x
BYOL	individual	×	×	66.8	66.0	65.6	65.3	63.0	62.1	59.5	56.0	54.3
			✓	66.0	66.2	65.6	64.4	63.4	61.8	59.3	56.1	54.0
SimCLR	individual	×	×	66.5	65.4	64.7	63.7	62.6	61.0	59.0	56.1	53.6
			✓	65.7	65.0	65.0	63.3	62.6	61.0	58.8	56.0	53.4
Ours	US-Net	✓	×	67.7	67.2	66.5	66.0	65.1	64.3	62.5	60.5	59.6
			✓	**68.6**	**67.8**	**67.3**	**66.4**	**65.5**	**64.4**	**63.1**	**60.9**	**60.1**

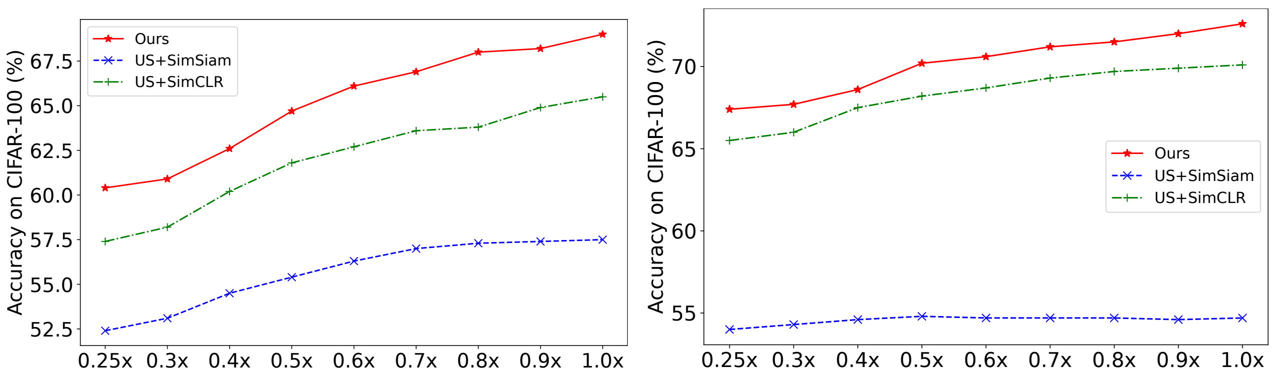
Table 16. Results of our US3L method at different widths under ResNet-18 and ResNet-50 on CIFAR-100. Our US3L can run at arbitrary width and we only reported partial results as a representative in the paper due to limited space.

Method	Backbone	Linear Accuracy (%) 1.0x	0.95x	0.9x	0.85x	0.8x	0.75x	0.7x	0.65x	0.6x	0.55x	0.5x	0.45x	0.4x	0.35x	0.3x	0.275x	0.25x
Ours (800ep)	R-18	70.1	69.6	69.3	69.2	69.0	68.4	68.7	68.0	67.3	66.7	66.4	65.4	64.2	63.6	63.1	63.1	62.3
	R-50	73.0	72.9	72.5	72.1	71.9	71.6	71.6	71.2	71.1	71.0	70.8	69.9	69.1	68.3	68.0	67.8	67.6



(a) ResNet-18 (b) ResNet-50

Figure 5. Comparison with individually trained baselines on CIFAR-100. All scatters are individually trained, whereas our method is trained only once (the red line).



(a) ResNet-18

(b) ResNet-50

Figure 6. Comparison with the original US-Net baseline on CIFAR-100. All are trained only once for 400 epochs.

Table 17. Ablation studies of the loss design under ResNet-18 on CIFAR-100. '-' denotes the model collapses.

Base Loss	Case	Distill Loss	Asymmetric Distill Head	Momentum Target		Linear Accuracy (%)								
				Base model	Sub model	1.0x	0.9x	0.8x	0.7x	0.6x	0.5x	0.4x	0.3x	0.25x
MSE	1	×	×	×	×	-	-	-	-	-	-	-	-	-
	2	MSE	×	×	×	-	-	-	-	-	-	-	-	-
	3	MSE	✓(Share)	×	×	57.5	57.4	57.3	57.0	56.3	55.4	54.5	53.1	52.4
	4	MSE	✓(Share)	×	✓	-	-	-	-	-	-	-	-	-
	5	MSE	×	×	✓	-	-	-	-	-	-	-	-	-
	6	MSE	×	✓	✓	-	-	-	-	-	-	-	-	-
	7	MSE	✓(Share)	✓	✓	64.7	64.7	64.5	64.3	63.9	62.6	61.3	59.7	59.3
	8	MSE	✓(New)	✓	✓	65.4	65.0	64.8	64.5	63.8	62.7	61.1	59.8	58.9
	9	InfoNCE	×	×	×	62.3	62.3	62.3	62.2	61.8	60.6	58.9	57.6	57.2
	10	InfoNCE	✓(Share)	×	×	58.7	58.8	58.8	58.9	58.7	58.4	56.8	55.3	54.3
	11	InfoNCE	✓(New)	×	×	61.5	61.4	61.6	61.6	61.1	60.3	58.7	57.1	56.3
	12	InfoNCE	×	×	✓	63.7	63.8	63.7	63.6	63.1	62.0	60.6	59.3	58.2
	13	InfoNCE	✓(Share)	×	✓	-	-	-	-	-	-	-	-	-
	14	InfoNCE	✓(New)	×	✓	64.5	64.5	64.6	64.5	64.2	63.2	62.1	60.0	59.1
	15	InfoNCE	×	✓	✓	65.0	65.0	65.1	65.0	64.5	62.7	61.3	59.8	59.2
	16	InfoNCE	✓(Share)	✓	✓	65.0	64.9	64.9	64.4	64.1	62.8	61.1	60.0	59.5
	17	InfoNCE	✓(New)	✓	✓	**65.5**	**65.5**	**65.6**	**65.0**	**64.6**	**63.2**	**61.6**	**60.2**	**59.7**
InfoNCE	18	×	×	×	×	64.8	64.0	63.2	62.0	60.8	59.8	57.4	55.1	54.2
	19	MSE	×	×	×	65.0	64.4	63.1	62.3	61.9	60.3	58.3	57.1	56.6
	20	MSE	×	×	✓	65.8	65.0	64.4	63.4	62.7	61.8	59.8	58.5	57.6
	21	MSE	✓	×	✓	66.7	66.0	65.6	64.5	63.3	62.0	60.8	59.3	58.2
	22	MSE	×	✓	✓	66.9	66.3	65.7	64.9	63.8	62.9	61.6	59.5	59.1
	23	MSE	✓	✓	✓	**67.7**	**67.2**	**66.5**	**66.0**	**65.1**	**64.3**	**62.5**	**60.5**	**59.6**
	24	InfoNCE	×	×	×	65.5	64.9	63.8	63.6	62.7	61.8	60.2	58.2	57.4
	25	InfoNCE	×	×	✓	64.7	64.5	64.0	63.6	62.3	61.4	59.8	58.4	57.9
	26	InfoNCE	✓	×	✓	66.1	66.0	65.4	64.4	63.4	62.3	60.8	59.1	58.6
	27	InfoNCE	×	✓	✓	66.0	65.4	64.8	64.3	63.8	62.4	61.1	59.8	58.7
	28	InfoNCE	✓	✓	✓	**67.4**	**66.0**	**66.1**	**65.6**	**64.7**	**64.0**	**62.2**	**60.2**	**59.5**