A Simple Data Mixing Prior for Improving Self-Supervised Learning

Sucheng Ren[1] Huiyu Wang[2] Zhengqi Gao[3] Shengfeng He[1*] Alan Yuille[2]
Yuyin Zhou[4] Cihang Xie[4*]

[1] South China University of Technology [2]Johns Hopkins University
[3] Massachusetts Institute of Technology [4]UC Santa Cruz

Abstract

Data mixing (e.g., Mixup, Cutmix, ResizeMix) is an essential component for advancing recognition models. In this paper, we focus on studying its effectiveness in the self-supervised setting. By noticing the mixed images that share the same source images are intrinsically related to each other, we hereby propose SDMP, short for ***S****imple* ***D****ata* ***M****ixing* ***P****rior, to capture this straightforward yet essential prior, and position such mixed images as additional* ***positive pairs*** *to facilitate self-supervised representation learning.*

Our experiments verify that the proposed SDMP enables data mixing to help a set of self-supervised learning frameworks (e.g., MoCo) achieve better accuracy and out-of-distribution robustness. More notably, our SDMP is the first method that successfully leverages data mixing to improve (rather than hurt) the performance of Vision Transformers in the self-supervised setting. Code is publicly available at `https://github.com/OliverRensu/SDMP`.

1. Introduction

Data mixing can effectively improve recognition models. The very first data mixing strategy is introduced in [48], *i.e.*, Mixup, which trains models on convex combinations of pairs of images and their labels. This idea subsequently inspired several follow-ups, including mixing images and cropped patches [46], mixing images and thumbnails [45], and mixing among cropped patches [4, 39].

However, interestingly, data mixing plays little role in the recent surge of self-supervised learning. For instance, while naïvely replacing original images with their mixed counterparts substantially improves Vision Transformers (ViTs) in the supervised setting [40], it cannot improve ViTs under the self-supervised setting. Though many efforts [28, 32, 43] have been made recently by developing more sophisticated training strategies in data mixing, they are exclusively focusing on Convolutional Neural Networks (CNNs). As shown in Table 1 in Section 4, these methods still fail to help (sometimes even hurt) ViTs [17].

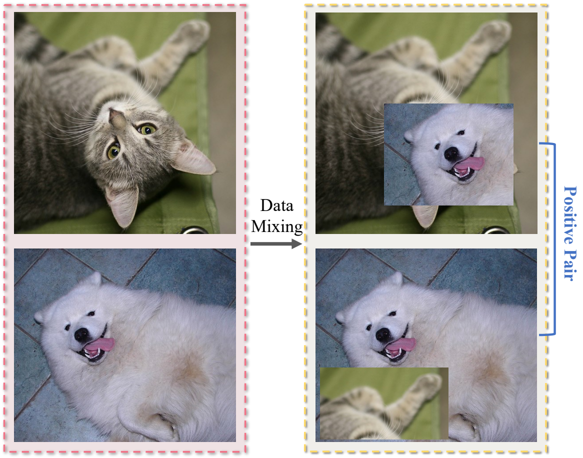


Figure 1. For the mixed images that share the same source (*e.g.*, a cat image and a dog image), they are semantically related and can be treated as additional positive pairs in self-supervised learning.

In this paper, *we aim to develop a generic training strategy in data mixing that can improve the self-supervised representation learning of both CNNs and ViTs*. By taking a closer look at the popular data mixing implementation where an image is mixed with another image *that sampled from the same batch but in the flipped order*[1], we note such created mixed samples are inherently related in pairs (*e.g.*, an example is provided in Figure 1). This indicates that now for one mixed image, there exist three related samples in the same training batch, *i.e.*, a pair of source images and a mixed image created with a different mixing coefficient. This intrinsic relationship qualifies the pair of mixed images to be treated as additional positive samples in self-supervised learning to facilitate representation learning.

[*] Corresponding authors: Shengfeng He (hesfe@scut.edu.cn), Cihang Xie (cixie@ucsc.edu)

[1] The traditional data mixing implementation randomly samples two batches and then mixes them with each other; while this instantiation of data mixing only samples one batch and then mixes pairs of images from the same batch. It generally will not hurt performance, and is very popular in many libraries (*e.g.*, timm [44]), due to its faster computation.

Motivated by the observation above, we hereby propose to leverage this **S**imple **D**ata **M**ixing **P**rior (dubbed *SDMP*) to holistically model the relationship among samples for enhancing self-supervised learning. Different from previous methods [28, 32, 33, 37, 43], SDMP not only considers the relationships between source images and the mixed counterparts, but also encodes the connections between mixed samples in representation learning. We further enhance SDMP's representation learning by semantically weighting the loss to capture the relationships among samples accurately.

Our empirical results verify that the proposed SDMP successfully helps a set of self-supervised learning frameworks gain better accuracy on visual benchmarks and robustness on out-of-distribution samples, for both CNNs and ViTs. More essentially, we stress that our SDMP is the first strategy that enables data mixing to improve self-supervised ViTs. For example, by building upon the latest MoCo v3 [12], while existing training strategies [28,32,43] all hurt the top-1 ImageNet accuracy of ViT-S by 0.2% - 1.6%, SDMP successfully improves ViT-S by 0.6%, attaining 73.8% top-1 ImageNet accuracy. We hope our technical insights and empirical results will be helpful for future works on studying data mixing in self-supervised learning.

2. Related Work

Self-supervised learning. Self-supervised learning aims to let models acquire semantically meaningful representations without human annotations. Traditional pretext tasks include reconstruction by autoencoder [3], colorization [49], rotation prediction [18] or combinations of them [16, 35].

Contrastive learning, which aims to discriminate between different samples, is one of the most successful pretext tasks. Its core idea is to maximize the similarity of positive pairs and minimize the similarity of negative pairs. However, discrimination based methods generally require a large amount of negative pairs, *e.g.*, SimCLR [8] needs a large training batch, MoCo [10, 22] requires a memory bank, and others [1, 5, 47] take a grouping/clustering. Later works [7, 11, 20] successfully remove the need for negative samples, enabling small batch training. In this work, we focus on improving self-supervised learning by having extra positive pairs (generated in data mixing).

Data mixing. Mixup [48] is the first work on data mixing, which convexly combines data pairs and their corresponding labels to regularize network training, inspiring numerous followups, including Cutout [15], CutMix [46], SaliencyMix [41] and PuzzleMix [29].

Recent works also introduce data mixing to regularize self-supervised learning. Verma *et al*. [43] utilize Mixup to create similar and dissimilar examples by mixing data samples differently, either at the input or hidden-state levels. [30, 32] explore the semi-contrastive encoding with a mixup of negative and positive pairs. Unlike previous works which exclusively focus on CNNs, we are the first to explore data mixing for improving ViTs under the self-supervised setting. We reveal properly modeling the relationships among mixed data (which was largely overlooked) can effectively strengthen self-supervised learning.

Transformers. Transformer [14, 42] is the de-facto standard for natural language processing tasks. Recently, Dosovitskiy *et al*. [17] successfully introduced the pure Transformer architecture for computer vision, attaining competitive recognition performance compared to CNNs on a range of visual benchmarks. Nonetheless, the original ViT training framework strongly demands hundreds of millions of images (*e.g.*, the in-house JFT dataset [38]) in training. Touvron *et al*. [40] relax this learning constraint by incorporating a set of strong regularization techniques into ViT training framework, where data mixing (more specifically, Mixup and CutMix) plays a vital role. In this work, we are particularly interested in exploring the potential of data mixing in helping ViT in the self-supervised setting.

3. Method

3.1. Which Images For Mixing?

Traditionally, data mixing generates mixed data by mixing two randomly sampled images (usually drawn from two different min-batches). While in this work, we follow the mixing strategy applied in the widely used Deep Learning Package timm [44]—we mix the i-th image with another randomly selected j-th image that comes from the same batch. *We by default set j equal to n-i where n is the number of images in this batch, for facilitating implementation*. We refer to this mixing strategy as *intra-batch mixing*. Note this intra-batch mixing strategy naturally enables the mixed images to be related in pairs (as they share the same source images, see an example in Figure 1), which will facilitate our virtual label assignment introduced in Section 3.3.

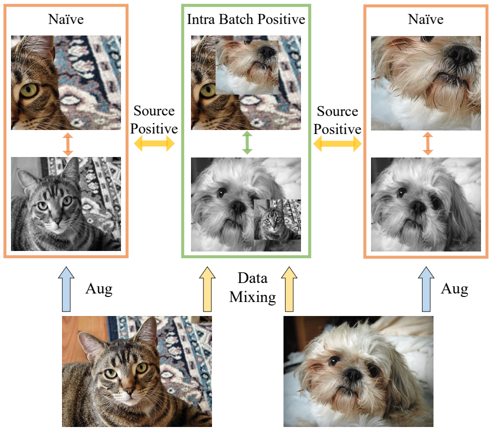
3.2. How To Mix?

To mix images, we mainly consider an element-wise data mixing method, Mixup [48], and two regional data mixing methods, *i.e.*, CutMix [46] and ResizeMix [36].

Mixup. Mixup element-wisely combines two sample while preserving the original content of source data. Specifically, Mixup takes the weight λ_i following a Beta distribution Beta(α, α) and mixes the data as the following:

$$\begin{aligned} x_i^{\text{mix}} &= \lambda_i x_i + (1-\lambda_i) x_{n-i}, \\ x_i'^{\text{mix}} &= \lambda_i x_i' + (1-\lambda_i) x_{n-i}', \end{aligned} \tag{1}$$

where x_i and x_{n-i} indicate the i-th and the $(n-i)$-th image; x_i' and x_{n-i}' are another view of the source images x_i and x_{n-i}, created by data augmentation (*e.g.*, color jittering).



Algorithm 1 Loss computation of our SDMP

```
a, b = aug(x), aug(x) # two different views of input x
lam = Beta(alpha, alpha).sample() # mixing coefficient
a = mix(a, a.flip(), lam)
a = normalize(model(a))
x_one_hot = one_hot(arange(len(x)))
logits1 = matmul(a, normalize(model(b)).T) / t
label1 = lam * x_one_hot + (1-lam) * x_one_hot.flip()
loss1 = CrossEntropyLoss(logits1, label1)
clam = min(lam, 1-lam.flip()) + min(1-lam, lam.flip())
logits2 = matmul(a, normalize(model(mix(b, b.flip(), lam))).T) / t
label2 = 1/(1+clam) * x_one_hot + clam/(1+clam) * x_one_hot.flip()
loss2 = CrossEntropyLoss(logits2, label2)
loss = loss1 + loss2
```

Figure 2. Left panel: The positive pairs considered in self-supervised learning, *i.e.*, the pair of different views (denoted as *naïve*), the pair of the source image and the mixed image (denoted as *source positive*), and the pair of the mixed images (denoted as *intra batch positive*). Right panel: the pseudo code of SDMP in PyTorch.

CutMix. CutMix [46] combines data regionally by cropping and pasting a specific patch from one image to another, *i.e.*, the mixed data comes from one whole image and a local region of another image. Note that both Mixup and CutMix by default are included in ViT's training recipe [40] under the supervised training setting.

ResizeMix. One potential issue of CutMix is the cropped patch and the image itself could be label-irrelevant. To let the cropped patch accurately deliver the corresponding semantics, ResizeMix [36] proposes to take the resized source image as the patch, and mix the data like the following:

$$\begin{aligned} P_i = R(x_i, H_i^p, W_i^p) \quad P_i' &= R(x_i', H_i^p, W_i^p), \\ x_i^{\text{mix}} &= \text{Paste}(P_{n-i}, x_i), \\ x_i'^{\text{mix}} &= \text{Paste}(P_{n-i}', x_i'), \\ \lambda_i &= 1 - \frac{W_{n-i}^p * H_{n-i}^p}{W_i * H_i}, \end{aligned} \tag{2}$$

where $R(\cdot, h, w)$ denotes the image resize function applied with the size of height h and width w; $\text{Paste}(P, x)$ will paste the patch P onto a random location of the image x; H_i, W_i are the height and the width of the i-th image x_i; H_i^p, W_i^p are the randomly sampled patch height and patch width for the i-th image x_i.

3.3. What Is the Label?

Given true labels are not available in the self-supervised setting, we next show how to assign virtual labels accordingly. Specifically, we showcase the virtual label assignments for two popular self-supervised learning frameworks.

Case 1: contrastive learning. Contrastive learning positions self-supervised learning as an instance classification task, assigning only one sample (created via data augmentation) as the positive pair and setting all the rest samples as negative. Our SDMP explicitly relaxes this assumption by introducing extra positive pairs. Specifically, we additionally assign the virtual positive labels to the following two groups: 1) the source data and the mixed counterparts; and 2) the pair of mixed data that share the same source data but with different mixing coefficients. This label assignment allows the model to learn to minimize the distance between more than just one pair.

We use the popular MoCo framework as the specific instantiation of contrastive learning. Firstly, to model the relationship between the source data and the mixed counterpart, the mix data x_i^m will be fed into the encoder f as the "query", and the other view of the source images, x_i' and x_{n-i}' (obtained via data augmentation), will be fed into the momentum encoder f_k as the "key". The loss of this part (referred as source loss) can be written as:

$$\begin{aligned} y_i^m = &f(x_i^m) \quad y_i' = f_k(x_i') \quad y_{n-i}' = f_k(x_{n-i}') \\ \mathcal{L}_{\text{MoCo}}^s = &- \lambda_i \log \frac{\exp(< y_i^m, y_i' > /\tau)}{\sum_{j=0}^{n} \exp(< y_i^m, y_j' > /\tau)} \\ &- (1 - \lambda_i) \log \frac{\exp(< y_i^m, y_{n-i}' > /\tau)}{\sum_{j=0}^{n} \exp(< y_i^m, y_j' > /\tau)}, \end{aligned} \tag{3}$$

where τ is the temperature to normalize the output y; recall λ_i is the mixing coefficient for creating the mix data x_i^m.

As mentioned in Section 3.1, we follow the intra-batch mixing strategy in timm [44] to mix one batch of data and its reversed-order counterpart accordingly. Note such mixed images are naturally related in pairs, *e.g.*, x_i^m and x_{n-i}^m are related because both of them are created by mixing x_i and x_{n-i}. In addition, by considering data augmentation, we propose to set x_i^m, $x_i'^m$ and $x_{n-i}'^m$ as the positive samples. Therefore the loss here (referred to as mixing loss), which aims to learn from the relationship between mix data, could be written as:

$$
\begin{aligned}
\lambda_i^c =& \min(\lambda_i, 1-\lambda_{n-i}) + \min(1-\lambda_i, \lambda_{n-i}), \\
\mathcal{L}_{\text{MoCo}}^m =& -\frac{1}{1+\lambda_i^c} \log \frac{\exp(< y_i^m, y_i'^m > /\tau)}{\sum_{j=0}^{n} \exp(< y_i^m, y_j'^m > /\tau)} \\
& - \frac{\lambda_i^c}{1+\lambda_i^c} \log \frac{\exp(< y_i^m, y_{n-i}'^m > /\tau)}{\sum_{j=0}^{n} \exp(< y_i^m, y_j'^m > /\tau)},
\end{aligned} \tag{4}
$$

where the coefficient λ^c aims to capture the shared semantics between two mixed data.

Note that Eq. (4) can either be used as an extra view, or replace one of the symmetric view in MoCo. We hereby take the replace version to describe the total loss of our SDMP in MoCo:

$$
\mathcal{L} = \sum_{i=0}^{n} \Big[\mathcal{L}_{\text{MoCo}}^s(x_i, x_i') + \mathcal{L}_{\text{MoCo}}^m(x_i'^m, x_i, x_{n-i}, x_i^m) \Big]. \tag{5}
$$

Case 2: knowledge distillation. Recent works [7, 11, 20] show negative samples could be "safely" dropped in self-supervised learning. We hereby focus on studying the knowledge distillation framework introduced in DINO [7], where the student model (with the parameter θ_s) is asked to match the output of the teacher model (with the parameter θ_t). The corresponding distillation loss is:

$$
H(P_t(x), P_s(x)) = -P_t(x) \log P_s(x), \tag{6}
$$

where $P_s(x)$ and $P_t(x)$ denote the output distribution (*i.e.*, after a softmax function) of the student model and the teacher model, respectively.

In our data mixing case, there are two teachers that the student needs to distill from. The first one is distilling from the source teacher, which takes the source data as input:

$$
\mathcal{L}_{\text{DINO}}^s = H(\lambda_i P_t(x_i') + (1-\lambda_i) P_t(x_{n-i}'), P_s(x_i^m)). \tag{7}
$$

With Eq. (7), we can minimize the distance between the mixed data and the two corresponding source data. Another teacher that needs to be distilled is the mix teacher, which take the mixed data as input:

$$
\begin{aligned}
\mathcal{L}_{\text{DINO}}^m =& \frac{1}{1+\lambda_i^c} H(P_t(x_i'^m), P_s(x_i^m)) \\
& + \frac{\lambda_i^c}{1+\lambda_i^c} H(P_t(x_{n-i}'^m), P_s(x_i^m)).
\end{aligned} \tag{8}
$$

With Eq. (8), the student can learn the output distribution of the mixed data from the mix teacher. Note that both the source teacher and the mix teacher share the same network parameter θ_t, which is updated by using the exponential moving average of the student parameter θ_s.

To sum up, the total loss of our SDMP in DINO is:

$$
\mathcal{L} = \mathcal{L}_{\text{DINO}} + \mathcal{L}_{\text{DINO}}^m + \mathcal{L}_{\text{DINO}}^s, \tag{9}
$$

where $\mathcal{L}_{\text{DINO}}$ is the original DINO loss that applied to the teacher-student pairs without data mixing.

4. Experiment

This section evaluates the effectiveness of SDMP on a set of standard visual benchmarks, including ImageNet [13] and CIFAR-10/100 [31]. Given no prior works successfully enable data mixing to improve self-supervised ViTs, we take ViT-S as the major backbone in experiments. The input patch size of ViT-S is 16×16, therefore resulting in a sequence length of 196 for a 224×224 input image. ViT-S has 12 transformer blocks, and the feature dimension of each block is 384. For data augmentation, we follow the settings in BYOL [20], which includes random resize and crop, color jittering, Gaussian Blurring, and solarization. We use Adam with weight decay as the optimizer [34], and set the learning rate lr following the linear scaling rule [19]: $lr = 0.0005 * \text{batchsize}/256$; our default training batch size is 1024.

We set contrastive learning and knowledge distillation as the default pretext tasks in self-supervised learning, and comprehensively measure the quality of learned representations via linear evaluation, end-to-end finetuning, and semi-supervised learning. In addition, we report model robustness on multiple out-of-distribution benchmarks, including ImageNet-A [27], ImageNet-R [25] and ImageNet-C [26].

4.1. Classification on ImageNet

4.1.1 Linear Evaluation

Linear evaluation [7, 23] accesses the performance of self-supervised learning by freezing all parameters in the backbone network and only training a linear classifier on top of it. Following the setups in the prior work [23], we only take resize, crop and random flipping to augment training images. Due to the variance of feature space in different self-supervised methods, we will adjust the initial learning rate accordingly. Note that most self-supervised ViTs only take the last class token for linear evaluation, while DINO takes the last four class tokens; we follow this setup in our DINO experiments. Moreover, when applying the proposed SDMP to MoCo, we randomly replace one of the symmetric views with the mixed data; when applying the proposed SDMP to DINO, we randomly replace half local views with the mixed data. This replacing strategy ensures SDMP will not bring extra computations to train student models.

Main results. We report the ImageNet linear evaluation results in Table 1. Firstly, we note that the proposed SDMP can bring consistent improvements over MoCo and DINO.

Method	Model	Param.	Epoch	Top-1 (%)
Supervised [40]	ViT-S	21M	300	79.8
BYOL [20]	ViT-S	21M	300	71.4
MoCo v2 [10]	ViT-S	21M	300	72.7
SwAV [6]	ViT-S	21M	300	73.5
MoCo v3 [12]	ViT-S	21M	300	73.2
+ imix* [32]	ViT-S	21M	300	71.6
+ DACL* [43]	ViT-S	21M	300	72.3
+ MoChi* [28]	ViT-S	21M	300	73.0
+ SDMP (ours)	ViT-S	21M	300	73.8
DINO [7]	ViT-S	21M	300	76.0
+ SDMP (ours)	ViT-S	21M	300	76.4
MoCo v3	ViT-B	85M	300	76.7
+ SDMP (ours)	ViT-B	85M	300	77.2
SimCLR [8]	Res50	23M	200	60.6
BYOL [20]	Res50	23M	200	61.9
SwAV [6]	Res50	23M	800	75.3
MoCo v1 [22]	Res50	23M	200	60.6
MoCo v2 [10]	Res50	23M	800	71.1
MoCo v3 [12]	Res50	23M	300	72.8
+ i-mix* [32]	Res50	23M	300	72.8
+ SDMP (ours)	Res50	23M	300	73.5

Table 1. The ImageNet performance of different pre-training methods under the linear evaluation protocol. Our SDMP consistently brings improvements over the baselines for both ViTs and CNNs. * indicate our reproduced results.

For examples, compared to the latest MoCo v3 baseline, SDMP can further boost the top-1 accuracy of ViT-S by 0.6% (*i.e.*, from 73.2% to 73.8%); for DINO, SDMP increases the top-1 accuracy of ViT-S to 76.4% (+0.4%, from 76.0%). In contrast, for existing training strategies that use data mixing, including i-mix [32], DACL [43] and MoChi [28], we note they all fail to help ViT gain stronger performance over the vanilla MoCo v3 baseline, decreasing the top-1 accuracy by 0.2% - 1.6%. Moreover, we verify that SDMP scales well with large ViTs, *i.e.*, it successfully helps ViT-B beat the vanilla MoCo v3 baseline by 0.5% (*i.e.*, from 76.7% to 77.2%).

Lastly, we observe that SDMP also helps CNNs gain better performance in self-supervised learning. For example, with ResNet-50, SDMP effectively improves the MoCo v3 baseline by 0.7% (from 72.8% to 73.5%), while the existing approaches like i-mix hardly bring in extra accuracy gain.

4.1.2 End-to-End Fintuning

We next follow the training recipe in DeiT [40] to finetune pre-trained ViTs. Specifically, these ViT are first pre-trained for 300 epochs in the self-supervised setting, and then finetuned for 100 epochs in the supervised setting. We additionally consider the pure supervised training as a strong baseline for comparison.

Method	Model	Param.	Epoch	Top-1
Supervised	ViT-S	21M	100	75.8
Supervised	ViT-S	21M	300	79.8
MoCo v3	ViT-S	21M	100	78.7
+ SDMP	ViT-S	21M	100	79.1
DINO	ViT-S	21M	100	79.7
+ SDMP	ViT-S	21M	100	80.0

Table 2. End-to-end fintuning on ImageNet. For self-supervised methods, all models here are pre-trained for 300 epochs. The "Epoch" in the table refers to the number of fintuning epochs.

Method	Model	Param.	10%	1%
MoCo v3	ViT-S	21M	66.7	54.4
+ SDMP	ViT-S	21M	67.4	55.5
DINO	ViT-S	21M	67.2	55.6
+ SDMP	ViT-S	21M	68.0	56.3

Table 3. Semi-supervised learning on ImageNet with 10% and 1% labeled data. All methods are pre-trained for 300 epochs.

We present the comparisons in Table 2. Firstly, compared to the vanilla self-supervised training baselines, SDMP brings consistent improvements. For example, with ViT-S, it beats the MoCo v3 baseline by 0.4% and the DINO baseline by 0.3%. More interestingly, when comparing to the strong supervised training baselines, we note SDMP a) substantially outperforms the 100-epoch supervised training baseline; and b) can closely match, or even outperforms, the strong 300-epoch supervised training baseline.

4.1.3 Semi-Supervised Learning

We now follow the semi-supervised learning protocol in [9], where the pre-trained will be finetuned with only a small portion of ImageNet, *e.g.*, 1% data or 10% data. The results are reported in Table 3. Firstly, our SDMP can consistently outperform the MoCo v3 baseline and the DINO baseline in both the 1% data setting and the 10% data setting. We note the performance gap between SDMP and the baseline tends to become larger if less data is used for finetuning. For example, with MoCo v3, the performance gap is 0.4% with 100% data (the end-to-end finetuning setting in Section 4.1.2), 0.7% with 10% data, and 1.1% with 1% data. This result suggests that SDMP can help self-supervised learning generalize better in the small labeled-data regime.

4.1.4 Robustness on Out-of-Distribution Datasets

We hereby evaluate model robustness on out-of-distribution data. Specifically, we test the performance on perturbed versions of ImageNet, *i.e.*, natural adversarial examples (ImageNet-A [27]), semantic shifts (ImageNet-R [25]), and common image corruption (ImageNet-C [26]).

Method	ImageNet (%)	A (%)	R (%)	C (%)
MoCo v3	78.7	18.1	42.1	52.9
+ SDMP	79.1	18.9	42.8	53.4
DINO	79.7	20.0	44.9	54.7
+ SDMP	80.0	21.1	45.3	55.0

Table 4. Performance on ImageNet and out-of-distribution datasets. "A", "R", "C" refer to ImageNet-A [27], ImageNet-R [25], and ImageNet-C [26], respectively. Note that when measuring performance on ImageNet-C, we directly use top-1 accuracy (the higher the better) as the evaluation metric, rather than using "mCE" (the lower the better) as originally defined in [26].

As shown in Table 4, our SDMP consistently improves the performance of the MoCo v3 baseline and the DINO baseline on these out-of-distribution tests. Notably, this robustness improvement could be much more substantial than the performance gain in the standard ImageNet benchmark. For instance, with DINO, our SDMP "merely" improve the ImageNet end-to-end finetuning result by 0.3%, whereas its gain on ImageNet-A is 1.1%. These results suggest that SDMP can be an ideal addition to existing self-supervised learning frameworks for enhancing model robustness.

4.2. CIFAR-10/100 Results

In addition to ImageNet, we study the popular CIFAR-10 and CIFAR-100 datasets [31] to verify the generalization of SDMP. Given it is extremely non-trivial to directly pre-train ViTs on the small CIFAR-10/100 dataset, we hereby exclusively focus on studying CNN architectures, more specifically, the ResNet family [24]. We will study ViTs + SDMP on the other small dataset in Section 4.3.5.

Experiment setup. CIFAR-10/100 contain 32×32 small size images with 10/100 classes, respectively. Both datasets contain 50000 training images and 10000 testing images. We select ResNet-50 and ResNet-101 as the main CNN architectures for pre-training. We slightly modify the ResNet architecture to make it more suitable for the CIFAR-10/100 training: for the first convolution layer, we change the kernel size from 7×7 to 3×3 and reduce the stride from 2 to 1; we also remove the max pooling layer that was originally placed right after the first convolution layer. We select the latest MoCo v3 as our self-supervised learning framework. In addition, we re-implement i-mix in MoCo v3 as a strong baseline for comparison.

CIFAR-10. The linear evaluation results on CIFAR-10 are shown in the fourth column of Table 5. Firstly, compared to the vanilla MoCo v3 baseline, we note applying data mixing always yields better CIFAR-10 performance. Secondly, though both approaches enhance the MoCo v3 baseline, we note that SDMP consistently attains higher performance than i-mix. For example, by pre-training ResNet-50 for 200 epochs, i-mix improves the vanilla baseline by 2.1% while SDMP yields a large performance improvement of 3.0%.

Method	Model	Epoch	CIFAR10	CIFAR100
MoCo v3	Res50	200	86.5	63.2
+ i-mix	Res50	200	88.6	66.1
+ SDMP	Res50	200	89.5	68.2
MoCo v3	Res50	2000	93.7	69.0
+ i-mix	Res50	2000	95.4	77.3
+ SDMP	Res50	2000	95.8	78.7
MoCo v3	Res101	200	86.4	63.3
+ i-mix	Res101	200	89.4	67.7
+ SDMP	Res101	200	90.0	69.7
MoCo v3	Res101	2000	93.8	68.5
+ i-mix	Res101	2000	95.8	78.4
+ SDMP	Res101	2000	95.8	80.0

Table 5. The CIFAR-10/100 performance of different pre-training methods under the linear evaluation protocol. We note SDMP consistently yields the best performance among all settings.

CIFAR-100. We report the linear evaluation results on CIFAR-100 in the last column of Table 5. Similar to the results in CIFAR-10, applying data mixing also helps MoCo v3 in CIFAR-100. We note the performance improvement is particularly substantial when you train with a larger model for longer epochs, *e.g.*, by pre-training ResNet-101 for 2000 epochs, both i-mix and SDMP can outperform the vanilla MoCo v3 by at least ~10.0%. We conjecture this is because CIFAR-100 is a relatively smaller dataset (*e.g.*, only 500 training samples per class); therefore, data augmentation is strongly demanded in training to help larger models generalize better. Next, we note SDMP consistently yields better performance than i-mix. For example, for both ResNet-50 and ResNet-101, SDMP beats i-mix by ~2% in the 200-epoch training, and by ~1.5% in the 2000-epoch training.

The results above suggest that the proposed SDMP can effectively generalize to CIFAR-10 and CIFAR-100. Moreover, we note the proposed SDMP can consistently outperforms i-mix [32] in all settings, corroborating our observation on ImageNet (in Section 4.1.1) that encoding the relationship between mixed data is essential for enhancing self-supervised learning.

4.3. Ablation Study

4.3.1 On the importance of λ

We sample λ from Beta distributions as the coefficient when mixing two images. We treat this λ as the prior in the loss during pre-training. In this part, we ablate how different setups of λ affect model performance.

Method	λ_i	λ_i^c	Linear	Finetuning
MoCo v3	None	None	73.2	78.7
+ SDMP	Static	Rand.	71.5	78.0
+ SDMP	Rand.	Static	72.5	78.4
+ SDMP	Rand.	Rand.	73.8	79.1
DINO	None	None	76.0	79.7
+ SDMP	Static	Rand.	75.5	79.4
+ SDMP	Rand.	Static	76.1	79.9
+ SDMP	Rand.	Rand.	76.4	80.0

Table 6. The ImageNet performance of different setups of λ. "None" refers to no data mixing is applied. "Rand." is the default setup in SDMP, which calculate λ_i or λ_i^c based on the randomly sampled λ. "Static" refers to the setting that λ_i or λ_i^c in the loss is a pre-defined constant and irrelevant to λ.

Static weight. There are two parts in the training loss related to the data mixing coefficient λ: the source loss (*e.g.*, Eq. (3) and Eq. (7)) and the mixing loss (*e.g.*, Eq. (4) and Eq. (8)). To check whether λ is a useful prior, we manually opt out the prior when computing the loss. Specifically, we keep the randomly sampled λ in data mixing; whereas for the loss computation, we could 1) set $\lambda_i = 0.5$ for ablating the source loss (*e.g.*, now both λ_i and $1 - \lambda_i$ equal to 0.5), or 2) set $\lambda_i^c = 1$ for ablating the mixing loss (*e.g.*, now both $\frac{1}{1+\lambda_i^c}$ and $\frac{\lambda_i^c}{1+\lambda_i^c}$ equal to 0.5). We referred to this weight assigning strategy as "static weight"; for the original weight assigning strategy introduced in Section 3.3, we refer to it as "random weight".

We report the ImageNet evaluation results in Table 6. Firstly, when applying "static weight" in the source loss (*i.e.*, $\lambda_i = 0.5$), we note the performance of both DINO and MoCo v3 substantially drops in both the linear evaluation and the end-to-end finetuning. Moreover, we note the resulted models even perform much worse than the vanilla MoCo v3 or DINO baselines. But meanwhile, one interesting observation is that, compared to MoCo v3, DINO can more robustly cope with "static weight". For example, by changing from "random weight" to "static weight", the linear evaluation accuracy largely drops by 2.3% in MoCo v3 (from 73.8% to 71.5%), while such drop in DINO is only 0.9% (from 76.4% to 75.5%).

Next, we ablate the effect of "static weight" in the mixing loss (*i.e.*, $\lambda_i^c = 1$). For MoCo v3, applying "static weight" always hurts performance. While for DINO, we find that "static weight" and "random weight" lead to similar accuracy, *e.g.*, 79.9% *vs.* 80.0% when finetuning the whole model. We conjecture this phenomenon should be attributed to the intra-batch similarity, as the intra-batch sample pair we built shares the same source data but only with different mixing coefficients, therefore by even using a constant λ_i^c can somewhat capture the overlapped semantics.

Method	λ_i	λ_i^c	Linear	Finetuning
MoCo v3	None	None	73.2	78.7
+ SDMP	Shared	Shared	72.3	78.2
+ SDMP	Ind.	Ind.	73.8	79.1
DINO	None	None	76.0	79.7
+ SDMP	Shared	Shared	74.5	79.0
+ SDMP	Ind.	Ind.	76.4	80.0

Table 7. The ImageNet performance of using independent (per-sample) λ (denoted as "Ind.") or shared (per-batch) λ across the whole training batch (denoted as "Shared").

Per-sample weight vs. per-batch weight. By default, we assign each sample with a randomly sampled mixing coefficient λ. An alternative strategy is to assign a shared mixing coefficient for the entire training batch, namely, setting $\lambda_1 = \lambda_2 = ... = \lambda_n$ for the training batch with n samples. The results are reported in Table 7. We note that taking a shared mixing coefficient leads to a performance decrease, *e.g.*, the accuracy drops by 1.5% in the linear evaluation and 0.9% in end-to-end finetuning for MoCo v3. We conjecture this is because the same mixing pattern is shared across the whole training batch, therefore weakening the training regularization brought by data mixing. These results suggest that assigning the per-sample λ is more effective than assigning the per-batch λ for mixing the data.

4.3.2 Data Mixing Strategies

For each training batch, the default setup in SDMP is to select a data mixing strategy from the set {Mixup, Cutmix and ResizeMix} uniformly at random. To ablate the effects of applying different data mixing strategies, we then compare our default setup with two additional settings: 1) applying exclusively with the element-wise data mixing Mixup; and 2) applying exclusively with the regional data mixing Resizemix. We report the results in Table 8. We note that: 1) our SDMP consistently outperforms the vanilla MoCo v3 or DINO baselines, even if only one data mixing strategy is applied; 2) our SDMP achieves the best results when {Mixup, Cutmix and ResizeMix} are all used.

Method	Model	Mixing	Epoch	Top-1 (%)
MoCo v3	ViT-S	None	100	64.7
+ SDMP	ViT-S	Mixup	100	65.1
+ SDMP	ViT-S	Resizemix	100	65.4
+ SDMP	ViT-S	All	100	65.5
DINO	ViT-S	None	100	73.8
+ SDMP	ViT-S	Mixup	100	74.3
+ SDMP	ViT-S	Resizemix	100	74.4
+ SDMP	ViT-S	All	100	74.4

Table 8. Ablations of different data mixing strategies.

4.3.3 Extra View Version *vs*. Replace Version in MoCo

The MoCo v3 framework default sees two differently augmented views of the same input. To additionally incorporate data mixing, the mixed data can either be used to replace one of the existing views or form an extra view. To ensure a fair comparison among different strategies, we set *the total training epoch to be the same as the vanilla MoCo v3 baseline for the replace version, but reduce the total training epoch by 1/3 for the extra view version*. This is because, compared to the vanilla MoCo v3 baseline and the replace version, the extra view version requires three views (instead of two views) from the same training sample. Table 9 reports the corresponding ImageNet performance. We can observe that both the replace version and the extra view version outperform the vanilla MoCo v3 baseline.

Method	Model	Epoch	Top-1 (%)
MoCo v3	ViT-S	150	66.7
+ SDMP (Replace)	ViT-S	150	67.4
+ SDMP (Extra)	ViT-S	100	67.5

Table 9. Comparisons among the extra view version, the replace version and the vanilla MoCo v3 baseline on ImageNet.

4.3.4 Local Views in DINO

One of the most important contributions in DINO is encouraging "local-to-global" correspondences [7], *i.e.*, in addition to two global views (at resolution 224^2 covering a large image region), DINO additionally introduces several local views (at resolution 96^2 covering only a small image region) in training. We hereby investigate the importance of the number of local views on SDMP. Specifically, when applying SDMP to DINO, we always replace half of the local views with mixed data; for the extreme case that no local views exist, we replace one of the global views with our mixed data. The ImageNet linear evaluation results are reported in Table 10. We note SDMP cannot work well, and sometimes even underperforms the vanilla DINO baseline, when no local views or only a few local views exist. Interestingly, by increasing the number of local views, SDMP begins to bridge such a performance gap, eventually outperforming the vanilla DINO baseline. For example, by training with eight local views, SDMP beats the vanilla DINO baseline by 0.4% (74.4% *vs*. 74.0%). These results suggest that having enough local views are essential for ensuring the improvements brought by data mixing.

Method	Global Clean	Global Mixed	Local Clean	Local Mixed	Top-1 (%)
DINO	2	✗	✗	✗	67.8
+ SDMP	1	1	✗	✗	64.1
DINO	2	✗	2	✗	71.5
+ SDMP	2	✗	1	1	70.9
DINO	2	✗	6	✗	73.8
+ SDMP	2	✗	3	3	74.0
DINO	2	✗	8	✗	74.0
+ SDMP	2	✗	4	4	74.4

Table 10. Ablating the effects of local views on ImageNet accuracy. All models are pre-trained for 100 epochs. Global/Local Clean denotes no data mixing is applied to global/local views. Global/Local Mixed denotes global/local views with data mixing.

4.3.5 Generalization on Small Datasets with ViTs

Section 4.2 demonstrates ResNet + SDMP can generalize well to small-scale datasets. We hereby verify if this conclusion holds for ViT + SDMP. We choose the relatively small dataset, *i.e.*, ImageNet-100, for this purpose. As shown in Table 11, SDMP consistently improves MoCo v3 (from 79.1% to 81.8%) and DINO (from 82.0% to 83.2%), showing its effectiveness at different data scale regime with ViTs.

Method	Model	Param.	Epoch	Linear
Supervised	ViT-S	21M	300	88.0
MoCo v3	ViT-S	21M	300	79.1
+ SDMP	ViT-S	21M	300	81.8
DINO	ViT-S	21M	300	82.0
+ SDMP	ViT-S	21M	300	83.2

Table 11. Linear evaluation on the small-scale ImageNet-100.

5. Conclusion

In this paper, we develop a generic training strategy for enabling data mixing to effectively help self-supervised training, especially with Vision Transformers. By following the intra-batch data mixing strategy in timm [44], we propose SDMP to capture the intrinsic relationships between mixed data in a precise manner. Experiments show that our method brings consistent improvements, and is compatible with various self-supervised learning frameworks, architectures, and datasets.

Discussion & Limitation This work mines the intra-batch relationship between mixed samples to help self-supervised learning. Future work could examine how to integrate our method into the recent self-supervised masked image modeling methods [2, 21, 50]. In addition, due to computational limitations, our experiments are mainly built upon the small-sized ViT (*i.e.*, ViT-S); future works could verify the effectiveness of our method at a larger scale.

Acknowledgement: This work is supported by a gift from Open Philanthropy, ONR N00014-21-1-2812, and Google Cloud Research Credits program.

References

[1] Yuki Markus Asano, Christian Rupprecht, and Andrea Vedaldi. Self-labelling via simultaneous clustering and representation learning. In *ICLR*, 2020. 2

[2] Hangbo Bao, Li Dong, and Furu Wei. Beit: Bert pre-training of image transformers. In *ICLR*, 2022. 8

[3] Yoshua Bengio, Pascal Lamblin, Dan Popovici, and Hugo Larochelle. Greedy layer-wise training of deep networks. In *NeurIPS*, 2007. 2

[4] Alexey Bochkovskiy, Chien-Yao Wang, and Hong-Yuan Mark Liao. Yolov4: Optimal speed and accuracy of object detection. *arXiv preprint arXiv:2004.10934*, 2020. 1

[5] Mathilde Caron, Piotr Bojanowski, Armand Joulin, and Matthijs Douze. Deep clustering for unsupervised learning of visual features. In *ECCV*, 2018. 2

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

[7] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *ICCV*, 2021. 2, 4, 5, 8

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

[9] Ting Chen, Simon Kornblith, Kevin Swersky, Mohammad Norouzi, and Geoffrey E Hinton. Big self-supervised models are strong semi-supervised learners. In *NeurIPS*, 2020. 5

[10] Xinlei Chen, Haoqi Fan, Ross Girshick, and Kaiming He. Improved baselines with momentum contrastive learning. *arXiv preprint arXiv:2003.04297*, 2020. 2, 5

[11] Xinlei Chen and Kaiming He. Exploring simple siamese representation learning. In *CVPR*, 2021. 2, 4

[12] Xinlei Chen, Saining Xie, and Kaiming He. An empirical study of training self-supervised vision transformers. In *ICCV*, 2021. 2, 5

[13] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009. 4

[14] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*, 2019. 2

[15] Terrance DeVries and Graham W Taylor. Improved regularization of convolutional neural networks with cutout. *arXiv preprint arXiv:1708.04552*, 2017. 2

[16] Carl Doersch and Andrew Zisserman. Multi-task self-supervised visual learning. In *ICCV*, 2017. 2

[17] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2020. 1, 2

[18] Spyros Gidaris, Praveer Singh, and Nikos Komodakis. Unsupervised representation learning by predicting image rotations. In *ICLR*, 2018. 2

[19] Priya Goyal, Piotr Dollár, Ross Girshick, Pieter Noordhuis, Lukasz Wesolowski, Aapo Kyrola, Andrew Tulloch, Yangqing Jia, and Kaiming He. Accurate, large minibatch sgd: Training imagenet in 1 hour. *arXiv preprint arXiv:1706.02677*, 2017. 4

[20] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre H Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Daniel Guo, Mohammad Gheshlaghi Azar, et al. Bootstrap your own latent: A new approach to self-supervised learning. In *NeurIPS*, 2020. 2, 4, 5

[21] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *CVPR*, 2022. 8

[22] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *CVPR*, 2020. 2, 5

[23] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross B. Girshick. Momentum contrast for unsupervised visual representation learning. In *CVPR*, 2020. 4

[24] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 6

[25] Dan Hendrycks, Steven Basart, Norman Mu, Saurav Kadavath, Frank Wang, Evan Dorundo, Rahul Desai, Tyler Zhu, Samyak Parajuli, Mike Guo, Dawn Song, Jacob Steinhardt, and Justin Gilmer. The many faces of robustness: A critical analysis of out-of-distribution generalization. *ICCV*, 2021. 4, 5, 6

[26] Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. *ICLR*, 2019. 4, 5, 6

[27] Dan Hendrycks, Kevin Zhao, Steven Basart, Jacob Steinhardt, and Dawn Song. Natural adversarial examples. *CVPR*, 2021. 4, 5, 6

[28] Yannis Kalantidis, Mert Bulent Sariyildiz, Noe Pion, Philippe Weinzaepfel, and Diane Larlus. Hard negative mixing for contrastive learning. In *NeurIPS*, 2020. 1, 2, 5

[29] Jang-Hyun Kim, Wonho Choo, and Hyun Oh Song. Puzzle mix: Exploiting saliency and local statistics for optimal mixup. In *ICML*, 2020. 2

[30] Sungnyun Kim, Gihun Lee, Sangmin Bae, and Se-Young Yun. Mixco: Mix-up contrastive learning for visual representation. *arXiv preprint arXiv:2010.06300*, 2020. 2

[31] Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, 2009. 4, 6

[32] Kibok Lee, Yian Zhu, Kihyuk Sohn, Chun-Liang Li, Jinwoo Shin, and Honglak Lee. i-mix: A domain-agnostic strategy for contrastive representation learning. In *ICLR*, 2021. 1, 2, 5, 6

[33] Chunyuan Li, Xiujun Li, Lei Zhang, Baolin Peng, Mingyuan Zhou, and Jianfeng Gao. Self-supervised pre-training with hard examples improves visual representations. *arXiv preprint arXiv:2012.13493*, 2020. 2

[34] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *ICLR*, 2019. 4

[35] Mehdi Noroozi, Ananth Vinjimoor, Paolo Favaro, and Hamed Pirsiavash. Boosting self-supervised learning via knowledge transfer. In *CVPR*, 2018. 2

[36] Jie Qin, Jiemin Fang, Qian Zhang, Wenyu Liu, Xingang Wang, and Xinggang Wang. Resizemix: Mixing data with preserved object information and true labels. *arXiv preprint arXiv:2012.11101*, 2020. 2, 3

[37] Zhiqiang Shen, Zechun Liu, Zhuang Liu, Marios Savvides, Trevor Darrell, and Eric Xing. Un-mix: Rethinking image mixtures for unsupervised visual representation learning. In *AAAI*, 2022. 2

[38] Chen Sun, Abhinav Shrivastava, Saurabh Singh, and Abhinav Gupta. Revisiting unreasonable effectiveness of data in deep learning era. In *ICCV*, 2017. 2

[39] Ryo Takahashi, Takashi Matsubara, and Kuniaki Uehara. Ricap: Random image cropping and patching data augmentation for deep cnns. In *ACML*, 2018. 1

[40] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In *ICML*, 2021. 1, 2, 3, 5

[41] AFM Uddin, Mst Monira, Wheemyung Shin, TaeChoong Chung, Sung-Ho Bae, et al. Saliencymix: A saliency guided data augmentation strategy for better regularization. In *ICLR*, 2021. 2

[42] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, 2017. 2

[43] Vikas Verma, Thang Luong, Kenji Kawaguchi, Hieu Pham, and Quoc Le. Towards domain-agnostic contrastive learning. In *ICML*, 2021. 1, 2, 5

[44] Ross Wightman. Pytorch image models. https://github.com/rwightman/pytorch-image-models, 2019. 1, 2, 3, 8

[45] Tianshu Xie, Xuan Cheng, Xiaomin Wang, Minghui Liu, Jiali Deng, Tao Zhou, and Ming Liu. Cut-thumbnail: A novel data augmentation for convolutional neural network. In *ACM MM*, 2021. 1

[46] Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *ICCV*, 2019. 1, 2, 3

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

[48] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. In *ICLR*, 2018. 1, 2

[49] Richard Zhang, Phillip Isola, and Alexei A Efros. Colorful image colorization. In *ECCV*, 2016. 2

[50] Jinghao Zhou, Chen Wei, Huiyu Wang, Wei Shen, Cihang Xie, Alan Yuille, and Tao Kong. ibot: Image bert pre-training with online tokenizer. In *ICLR*, 2022. 8