Boosting Discriminative Visual Representation Learning with Scenario-Agnostic Mixup

Anonymous authors
Paper under double-blind review

Abstract

Mixup is a popular data-dependent augmentation technique for deep neural networks, which contains two sub-tasks, mixup generation, and classification. The community typically confines mixup to supervised learning (SL) and the objective of the generation sub-task is fixed to selected sample pair instead of considering the whole data manifold. To overcome such limitations, we systematically study the mixup generation objective and propose **S**cenario-**A**gnostic **Mix**up for both SL and Self-supervised Learning (SSL) scenarios, named SAMix. Specifically, we hypothesize and verify the objective function of mixup generation as optimizing local smoothness between two mixed classes subject to global discrimination from other classes. Therefore, we propose η-balanced mixup loss for complementary learning of the two sub-objectives. Meanwhile, we parameterize the generation sub-task as a learnable sub-network, Mixer, with mixing attention which avoids trivial solutions and improves transferable abilities. To eliminate the computational cost of online training, we introduce a pre-trained version, SAMix$^{\mathcal{P}}$, that achieves efficient performance in various tasks. Extensive experiments on SL and SSL benchmarks demonstrate that SAMix consistently outperforms leading methods.

1 Introduction

One of the fundamental problems in machine learning is to learn a proper low-dimensional representation efficiently that captures the intrinsic structures of data and facilitates downstream tasks (Bengio et al., 2013; Devlin et al., 2018; Korbar et al., 2018; Chen et al., 2020a). Data mixing, as a means of generating symmetric mixed data and labels, largely improves the efficiency of deep neural networks (DNNs) learning discriminative representation in various scenarios (Zhang et al., 2018; Kim et al., 2020; Lee et al., 2021), especially for Vision Transformers (ViTs) (Dosovitskiy et al., 2021; Touvron et al., 2021). Despite its general application, the policy of the generation process in data mixing requires an *explicit* design. For example, mixed samples are constructed by linear interpolation or random local patch replacement of sample pairs (Zhang et al., 2018; Yun et al., 2019).

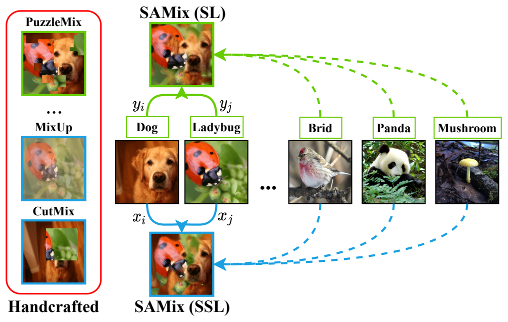

Figure 1: Mixed samples with green boxes rely on labels, while blue boxes do not. The solid line indicates that the local relationship directly influences the mixed sample, and the dashed line indicates that the other class samples serve as global constraints to the current mixed sample. Compared to handcrafted mixup policies that focus only on local sample pairs, SAMix can generate semantic mixed samples by exploiting the local and global information and has no dependency on labels.

Recent proposed optimizable approaches use labels to localize task-relevant targets (*e.g.*, grad-CAM (Selvaraju et al., 2019)) and generate semantic mixed samples such as offline maximizing saliency information of related samples (Kim et al., 2020; 2021; Uddin et al., 2020). These handcrafted mixing policies as shown in the red box of Figure 1. Additionally, (Zhu et al., 2020) learns a mixup generator by supervised adversarial training.

Although optimizable mixup methods have achieved significant gains in supervised learning (SL), it is not applicable in self-supervised learning (SSL). Recent AutoMix (Liu et al., 2022b) provides a new perspective to make mixup policy parameterized and can be trained online. However, the trivial solution is generated without the guidance of labels. Then there is the natural question of **whether we can design a more general and trained mixup policy that can be used for both SL and SSL.** Motivated by this intuition, there are two remaining open problems to be solved: *(i) how to solve the problem of trivial solution; (ii) what is a proper mixup generation objective for both SL and SSL.* Most current works combine mixup with contrastive learning, which directly transfers linear mixup methods into contrastive learning (Lee et al., 2021; Kalantidis et al., 2020; Shen et al., 2021). Although simple, these approaches do not exploit the underlying structure of the data manifold. In this paper, we propose SAMix (Figure 1), which stands for **S**cenario-**A**gnostic **Mix**up, a framework that employs *Mixer* to generate mixed samples adaptively either at *instance-level* or *cluster-level*. To guarantee that task-relevant information can be captured by Mixer, we propose *η-balanced mixup loss* for treating mixup generation and classification differently from a local and global perspective.

Furthermore, **in order to solve the additional computational overhead caused by the optimization process for mixup policy**, we propose a pre-trained setting, SAMix$^{\mathcal{P}}$, which employs a pre-trained Mixer to generate high-quality mixed samples balancing performance and speed. It is worth noticing that SAMix$^{\mathcal{P}}$ can achieve competitive or slightly higher performances like SAMix in classification tasks. Extensive experiments on both SL and SSL demonstrate the effectiveness and transferring abilities of SAMix. Our contributions are summarized as follows:

- We propose mixing attention and non-linear content modeling to design a label-free mixup generator, Mixer, with generalizable and transferable abilities in both SL and SSL.
- We decompose learning objectives (local and global terms) and analyze properties (local smoothness and global discrimination) of mixup generation, then propose η-balanced loss to boost mixup generation.
- Combining the above, a scenario-agnostic mixup training framework that supports online and pre-trained pipelines are designed for both SL and SSL tasks.
- Comprehensive experiments are conducted on nine benchmarks, and our SAMix and SAMix$^{\mathcal{P}}$ achieve state-of-the-art performances.

2 Preliminaries

Given a finite set of i.i.d samples, $X = [x_i]_{i=1}^n \in \mathbb{R}^{D\times n}$, each data $x_i \in \mathbb{R}^D$ is drawn from a mixture of, say C, distributions $\mathcal{D} = \{\mathcal{D}_c\}_{c=1}^C$. Our basic assumption for discriminative representations is that the each component distribution $\mathcal{D}_c$ has relatively low-dimensional intrinsic structures, *i.e.,* the distribution $\mathcal{D}_c$ is constrained on a sub-manifold, say $\mathcal{M}_c$ with dimension $d_c \ll D$. The distribution $\mathcal{D}$ of X is consisted of sub-manifolds, $\mathcal{M} = \cup_{c=1}^C \mathcal{M}_c$. We seek a low-dimensional representation $z_i \in \mathcal{M}$ of x_i by learning a continuous mapping by a network encoder, $f_\theta(x) : x \longmapsto z$ with the parameter $\theta \in \Theta$, which captures intrinsic structures of $\mathcal{M}$ and facilitates the discriminative tasks.

2.1 Discriminative Representation Learning

Parametric training with class supervision. *Some supervised class information* is available to ease the discriminative tasks in practical scenarios. Here, we assume that a one-hot label $y_i \in \mathbb{R}^C$ of each sample x_i can be somehow obtained, $Y = [y_1, y_2, ..., y_n] \in \mathbb{R}^{C\times n}$. We denote the labels generated during training as pseudo labels (PL), while the fixed as ground truth labels (L). Notice that each component $\mathcal{D}_c$ is considered *separated* according to Y in this scenario. Then, a *parametric* classifier, $g_\omega(z) : z \longmapsto p$ with the parameter $\omega \in \Omega$, can be learned to map the representation z_i of each sample to its class label y_i by predicting the probability of p_i being assigned to the c-th class using the softmax criterion, $p_{c|i} = \frac{\exp(w_c^T z_i)}{\sum_{j=1}^C \exp(w_j^T z_i)}$, where w_c is a weight vector for the class c, and $w_c^T z_i$ measures how similar between z_i and the class c. The learning objective is to minimize the cross-entropy loss (CE) between y_i and p_i,

$$\ell^{CE}(y_i, p_i) = -y_i \log p_i. \tag{1}$$

Non-parametric training as instance discrimination. Complementary to the above parametric settings, *non-parametric* approaches are usually adopted in unsupervised scenarios. Due to the *lack of class information*, an instance discriminative task can be designed based on an assumption of *local* *compactness*. We mainly discuss contrastive learning (CL) and take MoCo (He et al., 2020) as an example. Consider a pair of augmented image $(x_i^{\tau_q}, x_i^{\tau_k})$ from the same instance $x_i \in \mathbb{R}^{C\times H\times W}$, the local compactness is introduced by alignment of the encoded representation pair $(z_i^{\tau_q}, z_i^{\tau_k})$ from $f_{\theta,q}$ and the momentum $f_{\theta,k}$, and constrained to the global uniformity by contrasting $z_i^{\tau_q}$ to a dictionary of encoded keys from other images, $\{z_j^{\tau_k}\}_{j=1}^{K}$, where K denotes the length of the dictionary. It can be achieved by the popular non-parametric CL loss with temperature t, called infoNCE (van den Oord et al., 2019):

$$\ell^{NCE}(z_i^{\tau_q}, z_i^{\tau_k}) = -\log \frac{\exp(z_i^{\tau_q} z_i^{\tau_k}/t)}{\sum_{j=1}^{K}\exp(z_i^{\tau_q} z_j^{\tau_k}/t)}. \tag{2}$$

2.2 Mixup for Discriminative Representation

We further consider mixup as a generation task into discriminative representation learning to form a closed-loop framework. Then we have two mutually benefited sub-tasks: (a) *mixed data generation* and (b) *classification*. As for the sub-task (a), we define two functions, $h(\cdot)$ and $v(\cdot)$, to generate mixed samples and labels with a mixing ratio $\lambda \sim Beta(\alpha,\alpha)$. Given the mixed data, (b) defines a mixup training objective to optimize the representation space between instances or classes.

Mixup classification as the main task. Since we aim to seek a good representation to facilitate discriminative tasks, thus the mixed samples should be diverse and well-characterized. The mixed samples with semantic information can be easily obtained by parametric learning, while it becomes a challenge without supervision. Therefore, two types of the mixup classification objective $\mathcal{L}_{\theta,\omega}$ can be defined for *class-level* and *instance-level* mixup training. As for parametric training, given two randomly selected data pairs, (x_i, y_i) and (x_j, y_j), the mixed data is generated as $x_m = h(x_i, x_j, \lambda)$ and $y_m = v(y_i, y_j, \lambda)$. The objective of class-level mixup is similar to Eq. 1 as,

$$\ell^{CE}(y_m, p_m) = \lambda\ell^{CE}(y_m, p_m) + (1-\lambda)\ell^{CE}(y_m, p_m). \tag{3}$$

Notice that we fix $v(\cdot)$ as the linear interpolation in our discussions, *i.e.*, $v(y_i, y_j, \lambda) \triangleq \lambda y_i + (1-\lambda) y_j$. Symmetrically, we denote $h(\cdot)$ as a pixel-wise mixing policy with element-wise product $\odot$ for most input mixup methods (Zhang et al., 2018; Yun et al., 2019; Kim et al., 2020), *i.e.*, $x_m = s_i \odot x_i + s_j \odot x_j$, where $s_i \in \mathbb{R}^{H\times W}$ is a pixel-wise mask and $s_j = 1 - s_i$. Notice that each coordinate $s_{w,h} \in [0, 1]$. We can generate x_m with a pair of randomly selected samples (x_i, x_j) and formulate mixup infoNCE loss for instance-level mixup:

$$\ell^{NCE}(z_m) = \lambda\ell^{NCE}(z_m, z_i) + (1-\lambda)\ell^{NCE}(z_m, z_j), \tag{4}$$

where z_m, z_i and z_j denote the corresponding representations. The major difference with Eq. 3 is that the augmentation view that generates z_m is not from the same view, *i.e.*, z_i and z_j, as the objective function, which is effective in retaining task-relevant information, details in A.5.1.

Mixup generation as the auxiliary task. Unlike the learning object on the *unmixed* data X in Sec. 2.1, the performance of (b) mixup classification mainly depends on the quality of (a) mixup generation. We thus regard (a) as an auxiliary task to (b) and model $h(\cdot)$ as a sub-network $\mathcal{M}_\phi$ with the parameter $\phi \in \Phi$, called Mixer. Specifically, (a) aims to generate a pixel-wise mask $s \in \mathbb{R}^{H\times W}$ for mixing sample pairs. The mixup mask s_i should directly related to λ and the contents of (x_i, x_j). Practically, our $\mathcal{M}_\phi$ takes l-th layer feature maps $z^l \in \mathbb{R}^{C_l\times H_l\times W_l}$ and λ value as the input, $\mathcal{M}_\phi : x_i, x_j, z_i^l, z_j^l, \lambda \longmapsto x_m$. The generation process of $\mathcal{M}_\phi$ can be trained by a mixup generation loss as $\mathcal{L}_\phi^{gen}$, and a mask loss designed for generated mask s_i denoted as $\mathcal{L}_\phi^{mask}$. Formally, we have the mixup generation loss as $\mathcal{L}_\phi = \mathcal{L}_\phi^{cls} + \mathcal{L}_\phi^{mask}$, and the final learning objective is,

$$\min_{\theta,\omega,\phi} \mathcal{L}_{\theta,\omega} + \mathcal{L}_\phi. \tag{5}$$

Both $\mathcal{L}_{\theta,\omega}$ and $\mathcal{L}_\phi$ can be optimized alternatively in a unified framework using a momentum pipeline with the stop-gradient operation (Grill et al., 2020; Liu et al., 2022b), as shown in Figure 5 (left). **However, in SSL, this framework can easily make the generator fall into a trivial solution.** To solve this problem, we propose a novel mixup loss function and a generator architecture, Mixer. Combining the two can fully exploit the ability of mixup in learning discriminative features.

3 SAMix for Discriminative Representations

3.1 Learning Objective for Mixup Generation

Typically the objective function $\mathcal{L}_\phi$ corresponding to the mixup generation is consistent with the classification in parametric training (*e.g.*, ℓ^{CE}). In this paper, we argue that mixup generation is aimed to *optimize the local term subject to the global term.* The local term focuses on the classes of sample pairs to be mixed, while the global term introduces the constraints of other classes. For example, ℓ^{CE} is the global term whose each class produces an equivalent effect on the final prediction without focusing on the relevant classes of the current sample pair. At the class level, to emphasize the local term, we introduce a parametric binary cross-entropy (pBCE) loss for the generation task. Formally, assuming y_i and y_j belong to the class a and class b, pBCE can be summarized as:

$$\ell_{+}^{CE}(p_m) = -\lambda y_{i,a} \log p_m - (1-\lambda) y_{j,b} \log p_m, \tag{6}$$

where $y_{i,a} = 1$ and $y_{i,b} = 1$ denote the one-hot label for the class a and b. Notice that we use ℓ_+ and ℓ_- to represent the local and global terms, and ℓ_- for the parametric loss refers to ℓ^{CE}. Symmetrically, we have non-parametric binary cross-entropy mixup loss (BCE) for CL:

$$\ell_{+}^{NCE}(z_m) = -\lambda \log p_{m,i} - (1-\lambda) \log p_{m,j}, \tag{7}$$

where $p_{m,i} = \frac{\exp(z_m z_i/t)}{\exp(z_m z_i/t)+\exp(z_m z_j/t)}$ and its ℓ_- refers to ℓ^{NCE}.

Balancing local and global terms. Since both the local and global terms contribute to mixup generation, we analyze the importance of each term in both the SL and SSL tasks to design a balanced learning objective. We first analyze the properties of both terms with two hypothesizes: (i) the local term ℓ_+ *determines* the generation performance, (ii) the global term ℓ_- improves global discrimination but is sensitive to class information. To verify these properties, we design an empirical experiment based on the proposed Mixer on Tiny (see A.5). The main difference between the mixup CE and infoNCE is whether to adopt parametric class centroids. Therefore, we compare the intensity of class information among unlabeled (UL), pseudo labels (PL), and ground truth labels (L). Notice that PL is generated by ODC (Zhan et al., 2020) with the cluster number C. The class supervision can be imported to mixup infoNCE loss by filtering out negative samples with PL or L as (Khosla et al., 2020) denoted as infoNCE (L) and infoNCE (PL). As shown in Figure 2 (left), our hypothesizes are verified in the SL task (as the performance decreases from CE(L) to pBCE(L) and CE(PL) losses), but the opposite result appears in the CL task. The performance increases from InfoNCE(UL) to InfoNCE(L) as the false negative samples are removed (Robinson et al., 2021; Khosla et al., 2020) while trivial solutions occur using BCE(UL) (in Figure 6). Therefore, we propose it is better to explicitly import class information as PL for instance-level mixup to generate "strong" inter-class mixed samples while preserving intra-class compactness.

SAMix with η-balanced generation objectives. Practically, we provide two versions of the learning objective: the mixup CE loss with PL as the class-level version (SAMix-C), and the mixup infoNCE loss as the instance-level version (SAMix-I). Then, we hypothesize that the best performing mixed samples will be close to the sweet spot: achieving λ smoothness locally between two classes or neighborhood systems while globally discriminating from other classes or instances. We propose

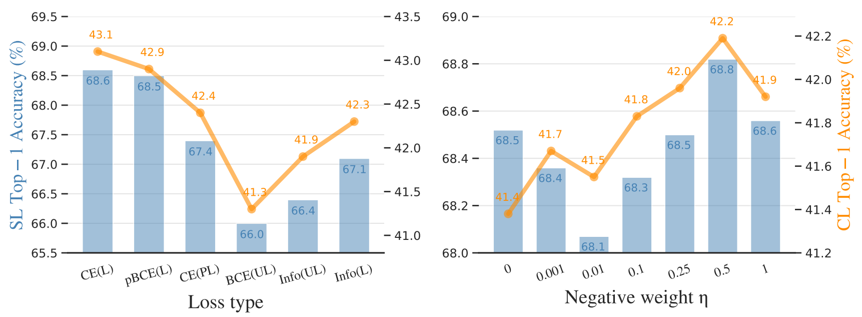

Figure 2: Analysis of the learning objective of Mixer on Tiny ImageNet with ResNet-18. The left figure shows results of various losses on the SL task (left y axis) and the CL task (right y axis). The right one shows the effect of using various negative weights η.

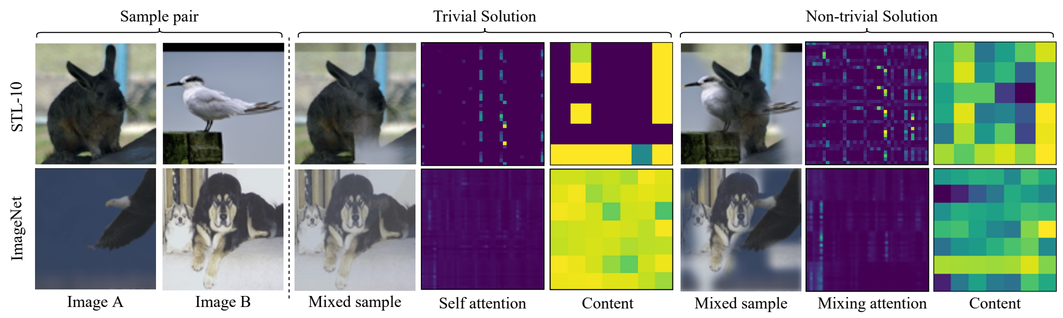

Figure 3: Visualization of trivial (using linear $\mathcal{C}$ and self-attention) and non-trivial solutions (using the proposed non-linear $\mathcal{C}_{NC}$ and mixing attention) for SAMix-I on STL-10 and IN-1k.

an η-*balanced mixup loss* as the objective of mixup generation,

$$\ell_\eta = \ell_+ + \eta \ell_-, \; \eta \in [0, 1]. \tag{8}$$

We analyze the performance of using various η in Eq. 8 on Tiny, as shown in Figure 2 (right), and find that using $\eta = 0.5$ performs best on both the SL and CL tasks. In the end, we provide the learning objective, $\mathcal{L}^{cls}_\phi \triangleq \ell^{CE}_+ + \eta \ell^{CE}_-$, with L for class-level mixup and with PL for SAMix-C, $\mathcal{L}^{cls}_\phi \triangleq \ell^{NCE}_+ + \eta \ell^{NCE}_-$ for SAMix-I (details in A.2).

3.2 De Novo Mixer for Mixup Generation

Although AutoMix proposes the MixBlock that learns adaptive mixup generation policies online, there are three drawbacks in practice: (a) fail to encode the mixing ratio λ on small datasets; (b) fall into trivial solutions when performing CL tasks; (c) the online training pipeline leads to double or more computational costs than MixUp. Our Mixer $\mathcal{M}_\phi$ solves these problems individually.

Adaptive λ encoding and mixing attention. Since a randomly sampled λ should directly guide mixup generation, the predicted mask s should be semantic proportional to λ. The previous typical design regards λ as the prior knowledge and concatenates λ to input feature maps, which might be unable to encode λ properly (the analysis in A.6.1). We propose an *adaptive λ encoding* as,

$$z^l_{i,\lambda} = (1 + \gamma\lambda) z^l_i, \tag{9}$$

where γ is a learnable scalar that constrained to $[0, 1]$. Notice that γ is initialized to 0 during training. As shown in Figure 4, we computes the mixing relationship between two samples using a new *mixing attention*: we concatenate $(z^l_{i,\lambda}, z^l_{j,1-\lambda})$ as the input, $\tilde{z}^l = \text{concat}(z^l_{i,\lambda}, z^l_{j,1-\lambda})$, and compute the attention matrix as,

$$P_{i,j} = \psi\left(\frac{(W_P \tilde{z}^l)^T \otimes W_P \tilde{z}^l}{\mathcal{N}(\tilde{z}^l)}\right), \tag{10}$$

where $\psi(\cdot)$ is the softmax function, $\mathcal{N}(\tilde{z}^l)$ denotes a normalization factor, and $\otimes$ is matrix multiplication. Notice that the mixing attention provides both the cross-attention between $z^l_{i,\lambda}$ and $z^l_{j,\lambda}$ and the self-attention of each feature itself.

Non-linear content modeling. In classical self-attention mechanism, the content sub-module $\mathcal{C}$ is a linear projection, $C_i = W_z \tilde{z}^l$, where W_z denotes a 1×1 convolution. However, we find the training process of this structure is unstable when performing CL tasks with the linear $\mathcal{C}$ in the early period and sometimes trapped in trivial solutions, such as all coordinates on s_i predicted as a constant. As shown in Figure 3, we visualize C_i and $P_{i,j}$ of trivial and non-trivial results, and find that the trivial s_i is usually caused by a constant C_i. We hypothesize that trivial solutions happen earlier in the

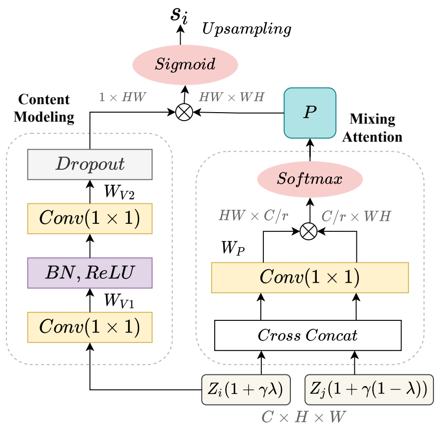

Figure 4: The network architecture of the proposed Mixer for mixup generation.

linear $\mathcal{C}$ than in $P_{i,j}$, because it might be unstable to project high-dimensional features to 1-dim linearly. Hence, we design a *non-linear content modeling* sub-module $\mathcal{C}_{NC}$ that contains two 1×1 convolution layers with a batch normalization layer and a ReLU layer in between, as shown in Figure 4. To increase the robustness and randomness of mixup training, we add a Dropout layer with a dropout ratio of 0.1 in $\mathcal{C}_{NC}$. Formally, Mixer $\mathcal{M}_\phi$ can be written as,

$$s_i = U\Bigg(\sigma\Bigg(\psi\Big(\frac{(W_P \tilde{z}^l)^T \otimes W_P \tilde{z}^l}{\mathcal{N}(\tilde{z}^l)}\Big) \otimes \mathcal{C}_{NC}(z^l_{i,\lambda})\Bigg)\Bigg). \tag{11}$$

Pre-trained pipeline *v.s.* online pipeline. Despite mixup methods optimized online (Liu et al., 2022b) outperforming hand-crafted variants by a large margin, their extra computational cost is unbearable, especially on large-scale datasets. On large-scale benchmarks, we notice that the mixed samples generated by SAMix in the early and late training periods or using different CNN encoders vary little. So we hypothesize that we can replace the online training Mixer with a pre-trained one in SAMix, similar to knowledge distillation (Dabouei et al., 2021). From this perspective, we propose the pre-trained pipeline of SAMix, denoted as SAMix$^\mathcal{P}$, in Figure 5. We can conclude: (i) SAMix$^\mathcal{P}$ pre-trained on large-scale datasets achieves similar or better performance as its online training version on current or relevant datasets with less computational cost. (ii) SAMix$^\mathcal{P}$ with light or median CNN encoders (*e.g.*, ResNet-18) yields better performance than heavy ones (*e.g.*, ResNet-101). (iii) SAMix$^\mathcal{P}$ has better transferring abilities than AutoMix$^\mathcal{P}$. (iv) Online training pipeline is still irreplaceable on small datasets (*e.g.*, CIFAR and CUB).

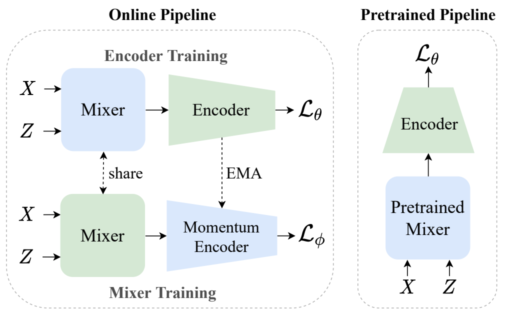

Figure 5: Training framework comparison of the online pipeline (left) in AutoMix and the proposed pre-trained pipeline (right). X and Z denote the input sample and corresponding feature maps from Momentum Encoder. Blue modules are not updated by back-propagation. The online pipeline optimizes Mixer and Encoder alternatively, while the pre-trained pipeline adopts pre-trained Mixer on large datasets.

Prior knowledge of mixup. Moreover, we summarize some commonly adopted prior knowledge (Kim et al., 2020; Dabouei et al., 2021) for mixup as two aspects: (a) adjusting the mean of s_i correlated with λ, and (b) balancing the smoothness of local image patches while maintaining discrimination of x_m. Based on them, we introduce λ *adjusting* and modifying the mask loss $\mathcal{L}^{mask}_\phi$ (details in A.2).

3.3 Discussion and Visualization of SAMix

To show the influence of local and global constraints on mixup generation, we visualize mixed samples generated from Mixer on various scenarios in Figure 6.

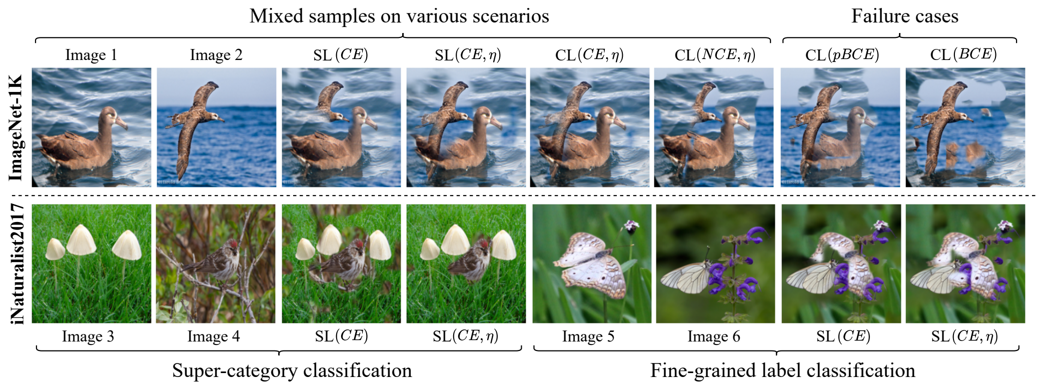

Figure 6: Visualization and comparison of mixed samples from Mixer in various learning scenarios on IN-1k and iNat2017. Note that $\lambda = 0.5$ and $\eta = 0.5$ if the balance coefficient η is included. CL(C) and CL(I) denote using SAMix-C and SAMix-I separately.

Class-level. In the supervised task, global constraint localizes key features by discriminating to other classes, while the local term is prone to preserve more information related to the current two samples and classes. For example, comparing the mixed results with and without η-balanced mixup loss, it was found that pixels of the foreground target were of interest to Mixer. When the global constraint is balanced ($\eta = 0.5$), the foreground target is retained more completely. Importantly, our designed Mixer remains invariant to the background for the more challenging fine-grained classification and preserves discriminative features.

Instance-level. Since no label supervision is available for SSL, the global and local terms are transformed from class to instance. Similar results are shown in the top row, and the only difference is that SAMix-C has a more precise target correspondence compared to SAMix-I via introducing class information by PL, which further indicates the importance of the information of classes. If we only focus on local relationships, Mixer can only generate mixed samples with fixed patterns (the last two results in the top row). These failure cases imply the importance of global constraints.

4 EXPERIMENTS

We first evaluate SAMix for supervised learning (SL) in Sec. 4.1 and self-supervised learning (SSL) in Sec. 4.2, and then perform ablation studies in Sec. 4.3. Nine benchmarks are used for evaluation: CIFAR-100 (Krizhevsky et al., 2009), Tiny-ImageNet (Tiny) (Chrabaszcz et al., 2017), ImageNet-1k (IN-1k) (Russakovsky et al., 2015), STL-10 (Coates et al., 2011), CUB-200 (Wah et al., 2011), FGVC-Aircraft (Aircraft) (Maji et al., 2013), iNaturalist2017/2018 (iNat2017/2018) (Horn et al., 2018), and Place205 (Zhou et al., 2014). All experiments are conducted with PyTorch and reported the *mean of 3 trials*. SAMix uses $\alpha = 2$ and the feature layer $l = 3$ while SAMix$^{\mathcal{P}}$ is pre-trained 100 epochs on IN-1k with ResNet-18. The *median* validation top-1 Acc of the last 10 epochs is recorded.

4.1 EVALUATION ON SUPERVISED IMAGE CLASSIFICATION

CNNs and ViTs are used as backbone networks, including ResNet (R), Wide-ResNet (WRN) (Zagoruyko & Komodakis, 2016), ResNeXt-32x4d (RX) (Xie et al., 2017), and DeiT (Touvron et al., 2021). We use PyTorch training procedures (Paszke et al., 2019) by default: an SGD optimizer with cosine scheduler (Loshchilov & Hutter, 2016). A special case in Table 4: RSB A3 (using a LAMB optimizer (You et al., 2020) for R-50) in timm (Wightman et al., 2021) and DeiT ((using an AdamW optimizer (Loshchilov & Hutter, 2019) for DeiT-S) training recipes are fully adopted on IN-1k. MCE and MBCE denote mixup cross-entropy and mixup binary cross-entropy in RSB A3. For a fair comparison, grid search is performed for hyper-parameters $\alpha \in \{0.1, 0.2, 0.5, 1.0, 2.0, 4.0\}$ of all mixup variants. We follow hyper-parameters in original papers by default. $*$ denotes unpublished

	PyTorch 100ep					PyTorch 300ep			
Methods	R-18	R-34	R-50	R-101	RX-101	R-18	R-34	R-50	R-101
Vanilla	70.04	73.85	76.83	78.18	78.71	71.83	75.29	77.35	78.91
MixUp	69.98	73.97	77.12	78.97	79.98	71.72	75.73	78.44	80.60
CutMix	68.95	73.58	77.17	78.96	80.42	71.01	75.16	78.69	80.59
ManifoldMix	69.98	73.98	77.01	79.02	79.93	71.73	75.44	78.21	80.64
SaliencyMix	69.16	73.56	77.14	79.32	80.27	70.21	75.01	78.46	80.45
FMix*	69.96	74.08	77.19	79.09	80.06	70.30	75.12	78.51	80.20
PuzzleMix	70.12	74.26	77.54	79.43	80.63	71.64	75.84	78.86	80.67
ResizeMix*	69.50	73.88	77.42	79.27	80.55	71.32	75.64	78.91	80.52
AutoMix*	70.50	74.52	77.91	79.87	80.89	72.05	76.10	79.25	80.98
SAMix$^{\mathcal{P}}$	70.83	74.95	78.06	**80.05**	80.98	72.27	76.28	79.39	**81.10**
SAMix	**70.85**	**74.96**	**78.11**	80.02	**81.03**	**72.33**	**76.35**	**79.40**	81.06

Table 1: Top-1 Acc (%) of image classification on IN-1k training 100-epoch and 300-epoch using procedures.

	iNat2017		iNat2018		Places205	
Method	R-50	RX-101	R-50	RX-101	R-18	R-50
Vanilla	60.23	63.70	62.53	66.94	59.63	63.10
MixUp	61.22	66.27	62.69	67.56	59.33	63.01
CutMix	62.34	67.59	63.91	69.75	59.21	63.75
ManifoldMix	61.47	66.08	63.46	69.30	59.46	63.23
SaliencyMix	62.51	67.20	64.27	70.01	59.50	63.33
FMix*	61.90	66.64	63.71	69.46	59.51	63.63
PuzzleMix	62.66	67.72	64.36	70.12	59.62	63.91
ResizeMix*	62.29	66.82	64.12	69.30	59.66	63.88
AutoMix*	63.08	68.03	64.73	70.49	59.74	64.06
SAMix$^{\mathcal{P}}$	**63.38**	68.23	**65.16**	**70.56**	59.82	**64.35**
SAMix	63.32	**68.26**	64.84	70.54	**59.86**	64.27

Table 2: Top-1 Acc (%) of image classification on iNat2017/2018 and Places205.

	CIFAR-100			Tiny		CUB-200		FGVC-Aircraft	
Method	R-18	RX-50	WRN-28-8	R-18	RX-50	R-18	RX-50	R-18	RX-50
Vanilla	78.04	81.09	81.63	61.68	65.04	77.68	83.01	80.23	85.10
MixUp	79.12	82.10	82.82	63.86	66.36	78.39	84.58	79.52	85.18
CutMix	78.17	81.67	84.45	65.53	66.47	78.40	85.68	78.84	84.55
ManifoldMix	80.35	82.88	83.24	64.15	67.30	79.76	86.38	80.68	86.60
SaliencyMix	79.12	81.53	84.35	64.60	66.55	77.95	83.29	80.02	84.31
FMix*	79.69	81.90	84.21	63.47	65.08	77.28	84.06	79.36	84.85
PuzzleMix	80.43	82.57	85.02	65.81	66.92	78.63	84.51	80.76	86.23
ResizeMix*	80.01	81.82	84.87	63.74	65.87	78.50	84.77	78.10	84.08
AutoMix*	82.04	83.64	85.16	67.33	70.72	79.87	86.56	81.37	86.69
SAMix	**82.30**	**84.42**	**85.50**	**68.89**	**72.18**	**81.11**	**86.83**	**82.15**	**86.80**

Table 3: Top-1 Acc (%) of supervised image classification on CIFAR-100, Tiny, CUB-200 and Aircraft.

	R-50 (A3)		DeiT-S
Methods	MCE	MBCE	MCE
MixUp+CutMix	76.49	78.08	79.80
MixUp	76.01	77.66	79.65
CutMix	76.47	77.62	79.78
SaliencyMix	76.85	77.93	79.32
FMix*	76.09	77.76	79.41
PuzzleMix	77.27	78.02	79.84
ResizeMix*	76.90	77.85	79.93
TransMix‡	-	-	80.70
AutoMix*	77.45	78.34	80.75
SAMix$^{\mathcal{P}}$	77.80	**78.73**	80.87
SAMix	**78.33**	78.45	**80.94**

Table 4: Top-1 Acc (%) of image classification on IN-1k.

work on *arxiv*. † and ‡ denote reproduced results by official codes and originally reported results, the rest are reproduced (see A.3 and A.4).

Comparison and discussion Table 3 shows results on small-scale and fine-grained classification tasks, SAMix consistently improves classification performances over the previous best algorithm AutoMix with improved Mixer. Notice that SAMix significantly improved the performance of CUB-200 and Aircraft by 1.24% and 0.78% based on ResNet-18, and continued to expand its dominance on Tiny by bringing 1.23% and 1.40% improvement on ResNet-18 and ResNeXt-50. As for the large-scale classification task, we benchmark popular mixup methods in Table 1, 4 and 2, SAMix and SAMix$^{\mathcal{P}}$ outperform all existing methods on IN-1k, iNat2017/2018 and Places205. Especially, SAMix$^{\mathcal{P}}$ yields similar or better performances than SAMix with less computational cost.

4.2 Evaluation on Self-supervised Learning

Then, we evaluate SAMix on SSL tasks pre-training on STL-10, Tiny, and IN-1k. All comparing methods are based on MoCo.V2 except SwAV (Caron et al., 2020). We adopt all hyper-parameter configurations from MoCo.V2 unless otherwise stated. We compared SAMix in two dimensions in CL: (i) compare with other mixup variants, based on our proposed cross-view pipeline, and the predefined cluster information is given (denotes by C) or not, as shown in Table 5. (ii) longitudinal comparison with CL methods that utilize *input space* and *input+latent* space mixup strategies, including MoCHi (Kalantidis et al., 2020), i-Mix (Lee et al., 2021), Un-Mix (Shen et al., 2021), and WBSIM (Chu et al., 2020), as shown in Table 6. Notice that $\star$ denotes our modified methods (PuzzleMix* uses PL and Inter-Intra* combines inter-class CutMix with intra-class MixUp.

Linear Classification Following the linear classification protocol proposed in MoCo, we train a linear classifier on top of frozen backbone features with the supervised train set. We train 100 epochs using SGD with a batch size of 256. The initialized learning rate is set to 0.1 for Tiny and STL-10 while 30 for IN-1k, and decay by 0.1 at epochs 30 and 60.

Comparison and discussion As shown in Table 5, SAMix-I outperforms all the linear mixup methods by a large margin, while SAMix-C surpasses the saliency-based PuzzleMix when PL is available. And SAMix-I has both global and local properties through infoNCE and BCE losses. Meanwhile, Table 6 demonstrates that both SAMix-I and SAMix-C surpass other CL methods combined with the predefined mixup. Overall, SAMix-C yields the best performance in CL takes which indicates it provides task-relevant information with the help of PL.

CL baseline	Methods	R18		R50	
		400ep	800ep	400ep	800ep
MoCo.V2	-	81.50	85.64	84.89	89.68
	MixUp	84.51	87.93	88.24	92.20
	ManifoldMix	84.17	87.70	88.06	91.65
	CutMix	84.28	87.60	87.51	90.81
MoCo.V2	SaliencyMix	84.33	87.27	87.35	90.77
	FMix*	84.43	87.68	88.14	91.56
	ResizeMix*	83.88	87.25	86.88	90.83
	SAMix-I	**85.44**	**88.58**	**88.87**	**92.41**
SwAV† (C)	-	81.10	85.56	84.35	88.79
MoCo.V2 (C)	Inter-Intra*	84.89	87.85	88.33	92.24
MoCo.V2 (C)	PuzzleMix*	84.98	88.07	88.40	91.98
MoCo.V2 (C)	**SAMix-C**	**85.60**	**88.63**	**88.91**	**92.45**

Table 5: Top-1 Acc (%) of linear classification pre-trained on STL-10.

CL method	Methodd	Tiny		IN-1k	
		R18	R50	R18	R50
MoCo.V2	-	38.29	42.08	52.85	67.66
MoCo.V2	MixUp	41.24	46.61	53.03	68.07
MoCo.V2	CutMix	41.62	46.24	52.98	68.28
MoCo.V2	SaliencyMix	41.14	46.13	53.06	68.31
MoCo.V2 (C)	PuzzleMix*	41.86	46.72	53.46	68.48
MoCHi†	*input+latent*	41.78	46.55	53.12	68.01
i-Mix†	*input+latent*	41.61	46.57	53.09	68.10
UnMix‡	*input+latent*	-	-	-	68.60
WBSIM‡	*input*	-	-	-	68.40
MoCo.V2	**SAMix-I**	41.97	47.23	53.75	68.76
MoCo.V2	**SAMix-I$^{\mathcal{P}}$**	43.57	**48.10**	53.72	68.82
MoCo.V2 (C)	**SAMix-C**	**43.68**	47.51	**53.93**	**68.86**

Table 6: Top-1 Acc (%) of linear classification pre-trained on Tiny and IN-1k.

Downstream Tasks Following the protocol in MoCo, we evaluate transferable abilities of the learned representation of comparing methods to object detection task on PASCAL VOC (Everingham et al., 2010) and COCO (Lin et al., 2014) in Detectron2 (Wu et al., 2019). We fine-tune Faster R-CNN (Ren et al., 2015) with pre-trained models on VOC *trainval07+12* and evaluate on the VOC *test2007* set. Similarly, Mask R-CNN (He et al., 2017) is fine-tuned (2× schedule) on the COCO *train2017* and evaluated on the COCO *val2017*. SAMix still achieves comparable performance with state-of-the-art mixup methods. More results are detailed in A.5.5.

4.3 Ablation Study

We conduct ablation studies in four aspects: (i) **Mixer**: Table 8 verifies the effectiveness of each proposed module in both SL and CL tasks on Tiny. The first three modules enable Mixer to model

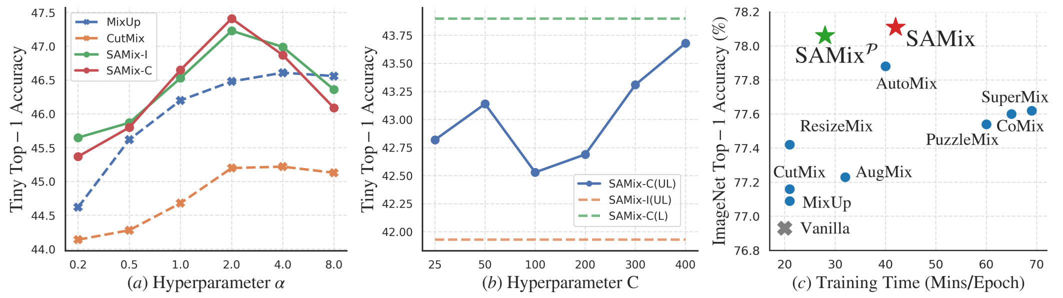

Figure 7: (a) Hyper-parameter α for mixup. (b) The cluster number C for SAMix-C in CL tasks on Tiny. (c) Top-1 accuracy *v.s.* training time on IN-1k based on ResNet-50 with 100 epochs.

the non-linear mixup relationship, while the next two modules enhance Mixer, especially in CL tasks. (ii) **Learning objectives**: We analyze the effectiveness of proposed ℓ_η with other losses, as shown in Table 7. Using ℓ_η for the mixup CE and infoNCE consistently improves the performance both for the CL task on STL-10 and Tiny. (iii) **Time complexity analysis**: Figure 7 (c) shows computational analysis conducted on the SL task on IN-1k using PyTorch 100-epoch settings. It shows that the overall accuracy *v.s.* time efficiency of SAMix and SAMix$^{\mathcal{P}}$ are superior to other methods. (iv) **Hyper-parameters**: Figure 7 (a) and (b) show ablation results of the hyper-parameter α and the clustering number C for SAMix-C. We empirically choose α=2.0 and $C = 200$ as default.

method	STL-10	Tiny
BCE	85.25	41.28
infoNCE	85.36	41.85
infoNCE ($\eta = 0.5$)	**85.44**	**41.97**
CE (PL)	85.56	42.36
CE (PL)+infoNCE	85.41	42.12
CE (PL, $\eta = 0.5$)	**85.60**	**42.53**

Table 7: Ablation of the proposed learning objectives in the SSL task on STL-10 and Tiny.

module	SL	CL
Mixing attention	67.17	40.58
+Adaptive λ	67.95	41.82
+Non-linear content	68.46	42.45
+$\mathcal{L}_{mask}$	68.57	42.68
+λ adjusting	68.61	43.14
+ℓ_η ($\eta = 0.5$)	**68.82**	**43.68**

Table 8: Ablation of proposed modules in Mixer and the η-balanced loss on Tiny.

5 Related Work

Class-level Mixup There are four types sample mixing policies for class-level mixup: linear mixup of input space (Zhang et al., 2018; Yun et al., 2019; Hendrycks et al., 2019; Harris et al., 2020; Qin et al., 2020) and latent space (Verma et al., 2019; Faramarzi et al., 2020), saliency-based (Uddin et al., 2020; Kim et al., 2020; 2021), generation-based (Zhu et al., 2020; Venkataramanan et al., 2022), and learning mixup generation and classification end-to-end (Liu et al., 2022b; Dabouei et al., 2021). SAMix belongs to the fourth type and learns both class- and instance-level mixup relationships, and its pre-trained SAMix$^{\mathcal{P}}$ eliminates high time-consuming problems of this type of method. Additionally, some researchers (Park et al., 2022; Chen et al., 2021; Liu et al., 2022a) improve class mixing policies upon linear mixup. See A.7 for details.

Instance-level Mixup A complementary method to learn better instance-level representation is to apply mixup in SSL scenarios (Lee et al., 2021). However, most approaches are limited to using linear mixup variants, such as applying MixUp and CutMix in the input or latent space mixup (Kalantidis et al., 2020; Lee et al., 2021; Chu et al., 2020) for SSL without ground-truth labels. SAMix improves SSL tasks by learning mixup policies online.

6 Limitations and Conlusions

We first study objectives for mixup generation as local-emphasized and global-constrained terms for learning adaptive mixup policy at both class- and instance-level. SAMix provides a unified framework with both online and pre-trained pipelines to boost discriminative representation learning based on improved Mixer. As a limitation, the Mixer only takes two samples as input and conflicts when the task-relevant information is overlapping. In the future, more than two samples or conflict-aware Mixer is another promising avenue for future research.

References

Mohamed Ishmael Belghazi, Aristide Baratin, Sai Rajeswar, Sherjil Ozair, Yoshua Bengio, Aaron Courville, and R Devon Hjelm. Mine: Mutual information neural estimation. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2018.

Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1798–1828, 2013.

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

Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

Jie-Neng Chen, Shuyang Sun, Ju He, Philip Torr, Alan Yuille, and Song Bai. Transmix: Attend to mix for vision transformers, 2021.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2020a.

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

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

Patryk Chrabaszcz, Ilya Loshchilov, and Frank Hutter. A downsampled variant of imagenet as an alternative to the cifar datasets. *arXiv preprint arXiv:1707.08819*, 2017.

Xiangxiang Chu, Xiaohang Zhan, and Xiaolin Wei. Beyond single instance multi-view unsupervised representation learning. *arXiv preprint arXiv:2011.13356*, 2020.

Adam Coates, Andrew Ng, and Honglak Lee. An analysis of single-layer networks in unsupervised feature learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pp. 215–223. JMLR Workshop and Conference Proceedings, 2011.

Ali Dabouei, Sobhan Soleymani, Fariborz Taherkhani, and Nasser M Nasrabadi. Supermix: Supervising the mixing data augmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 13794–13803, 2021.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

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 *ICLR*, 2021.

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.

Mojtaba Faramarzi, Mohammad Amini, Akilesh Badrinaaraayanan, Vikas Verma, and Sarath Chandar. Patchup: A regularization technique for convolutional neural networks. *arXiv preprint arXiv:2006.07794*, 2020.

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 *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

Ethan Harris, Antonia Marcu, Matthew Painter, Mahesan Niranjan, and Adam Prügel-Bennett Jonathon Hare. Fmix: Enhancing mixed sample data augmentation. *arXiv preprint arXiv:2002.12047*, 2(3):4, 2020.

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

Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2017.

Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9729–9738, 2020.

Dan Hendrycks, Norman Mu, Ekin D Cubuk, Barret Zoph, Justin Gilmer, and Balaji Lakshminarayanan. Augmix: A simple data processing method to improve robustness and uncertainty. *arXiv preprint arXiv:1912.02781*, 2019.

Grant Van Horn, Oisin Mac Aodha, Yang Song, Yin Cui, Chen Sun, Alex Shepard, Hartwig Adam, Pietro Perona, and Serge Belongie. The inaturalist species classification and detection dataset. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

Yannis Kalantidis, Mert Bulent Sariyildiz, Noe Pion, Philippe Weinzaepfel, and Diane Larlus. Hard negative mixing for contrastive learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. Supervised contrastive learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

Jang-Hyun Kim, Wonho Choo, and Hyun Oh Song. Puzzle mix: Exploiting saliency and local statistics for optimal mixup. In *International Conference on Machine Learning*, pp. 5275–5285. PMLR, 2020.

Jang-Hyun Kim, Wonho Choo, Hosan Jeong, and Hyun Oh Song. Co-mixup: Saliency guided joint mixup with supermodular diversity. *arXiv preprint arXiv:2102.03065*, 2021.

Bruno Korbar, Du Tran, and Lorenzo Torresani. Cooperative learning of audio and video models from self-supervised synchronization. *arXiv preprint arXiv:1807.00230*, 2018.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pp. 1097–1105, 2012.

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 *International Conference on Learning Representations (ICLR)*, 2021.

Siyuan Li, Zicheng Liu, Zedong Wang, Di Wu, and Stan Z. Li. OpenMixup: Open mixup toolbox and benchmark for visual representation learning. https://github.com/Westlake-AI/openmixup, 2022.

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 *Proceedings of the European Conference on Computer Vision (ECCV)*, 2014.

Zicheng Liu, Siyuan Li, Ge Wang, Cheng Tan, Lirong Wu, and Stan Z. Li. Decoupled mixup for data-efficient learning, 2022a.

Zicheng Liu, Siyuan Li, Di Wu, Zhiyuan Chen, Lirong Wu, Jianzhu Guo, and Stan Z Li. Automix: Unveiling the power of mixup. In *Proceedings of the European Conference on Computer Vision (ECCV)*. Springer, 2022b.

Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983*, 2016.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations (ICLR)*, 2019.

Subhransu Maji, Esa Rahtu, Juho Kannala, Matthew Blaschko, and Andrea Vedaldi. Fine-grained visual classification of aircraft. *arXiv preprint arXiv:1306.5151*, 2013.

Joonhyung Park, June Yong Yang, Jinwoo Shin, Sung Ju Hwang, and Eunho Yang. Saliency grafting: Innocuous attribution-guided mixup with calibrated label mixing. In *AAAI Conference on Artificial Intelligence*, 2022.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

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.

Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *arXiv preprint arXiv:1506.01497*, 2015.

Joshua David Robinson, Ching-Yao Chuang, Suvrit Sra, and Stefanie Jegelka. Contrastive learning with hard negative samples. In *International Conference on Learning Representations*, 2021.

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

Ramprasaath R. Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. *arXiv preprint arXiv:1610.02391*, 2019.

Zhiqiang Shen, Zechun Liu, Zhuang Liu, Marios Savvides, Trevor Darrell, and Eric Xing. Un-mix: Rethinking image mixtures for unsupervised visual representation learning. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, 2021.

Yonglong Tian, Dilip Krishnan, and Phillip Isola. Contrastive multiview coding. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020a.

Yonglong Tian, Chen Sun, Ben Poole, Dilip Krishnan, Cordelia Schmid, and Phillip Isola. What makes for good views for contrastive learning? In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020b.

Hugo Touvron, M. Cord, M. Douze, F. Massa, Alexandre Sablayrolles, and H. Jégou. Training data-efficient image transformers & distillation through attention. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2021.

Yao-Hung Hubert Tsai, Yue Wu, Ruslan Salakhutdinov, and Louis-Philippe Morency. Self-supervised learning from a multi-view perspective. In *International Conference on Learning Representations (ICLR)*, 2021.

AFM Uddin, Mst Monira, Wheemyung Shin, TaeChoong Chung, Sung-Ho Bae, et al. Saliencymix: A saliency guided data augmentation strategy for better regularization. *arXiv preprint arXiv:2006.01791*, 2020.

Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2019.

Shashanka Venkataramanan, Ewa Kijak, Laurent Amsaleg, and Yannis Avrithis. Alignmixup: Improving representations by interpolating aligned features. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

Vikas Verma, Alex Lamb, Christopher Beckham, Amir Najafi, Ioannis Mitliagkas, David Lopez-Paz, and Yoshua Bengio. Manifold mixup: Better representations by interpolating hidden states. In *International Conference on Machine Learning*, pp. 6438–6447. PMLR, 2019.

Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. The caltech-ucsd birds-200-2011 dataset. California Institute of Technology, 2011.

Ross Wightman, Hugo Touvron, and Hervé Jégou. Resnet strikes back: An improved training procedure in timm. *arXiv preprint arXiv:2110.00476*, 2021.

Yuxin Wu, Alexander Kirillov, Francisco Massa, Wan-Yen Lo, and Ross Girshick. Detectron2. `https://github.com/facebookresearch/detectron2`, 2019.

Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1492–1500, 2017.

Yang You, Jing Li, Sashank Reddi, Jonathan Hseu, Sanjiv Kumar, Srinadh Bhojanapalli, Xiaodan Song, James Demmel, Kurt Keutzer, and Cho-Jui Hsieh. Large batch optimization for deep learning: Training BERT in 76 minutes. In *International Conference on Learning Representations (ICLR)*, 2020.

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 *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 6023–6032, 2019.

Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. In *Proceedings of the British Machine Vision Conference*, 2016.

Xiaohang Zhan, Jiahao Xie, Ziwei Liu, Yew Soon Ong, and Chen Change Loy. Online deep clustering for unsupervised representation learning. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. In *International Conference on Learning Representations (ICLR)*, 2018.

Bolei Zhou, Agata Lapedriza, Jianxiong Xiao, Antonio Torralba, and Aude Oliva. Learning deep features for scene recognition using places database. In *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 487–495, 2014.

Jianchao Zhu, Liangliang Shi, Junchi Yan, and Hongyuan Zha. Automix: Mixup networks for sample interpolation via cooperative barycenter learning. In *European Conference on Computer Vision*. Springer, 2020.

A Appendix

We first provide dataset settings in A.1 and implementation details for supervised (SL) and self-supervised learning (SSL) tasks in A.2, A.3 and A.4. Then, we detailed experiment settings and analysis for Sec. 3 in A.5. Moreover, we visualize mixed samples in A.6, and provide detailed related work in A.7.

A.1 Basic Settings

Reproduction details. We use OpenMixup Li et al. (2022) implemented in PyTorch (Paszke et al., 2019) as our code-base for both supervised image classification and contrastive learning (CL) tasks. Except results marked by † and ‡, we reproduce most experiment results of compared methods, including Mixup (Zhang et al., 2018), CutMix (Yun et al., 2019), ManifoldMix (Verma et al., 2019), SaliencyMix (Uddin et al., 2020), FMix (Harris et al., 2020), and ResizeMix (Qin et al., 2020).

Dataset information. We briefly introduce image datasets used in Sec. 4: (1) CIFAR-100 (Krizhevsky et al., 2009) contains 50k training images and 10K test images of 100 classes. (2) ImageNet-1k (IN-1k) (Krizhevsky et al., 2012) contains 1.28 million training images and 50k validation images of 1000 classes. (3) Tiny-ImageNet (Tiny) (Chrabaszcz et al., 2017) is a rescaled version of ImageNet-1k, which has 100k training images and 10k validation images of 200 classes. (4) STL-10 (Coates et al., 2011) benckmark is designed for semi- or unsupervised learning, which consists of 5k labeled training images for 10 classes 100k unlabelled training images, and a test set of 8k images. (5) CUB-200-2011 (CUB) (Wah et al., 2011) contains over 11.8k images from 200 wild bird species for fine-grained classification. (6) FGVC-Aircraft (Aircraft) (Maji et al., 2013) contains 10k images of 100 classes of aircraft. (7) iNaturalist2017 (iNat2017) (Horn et al., 2018) is a large-scale fine-grained classification benchmark consisting of 579.2k images for training and 96k images for validation from over 5k different wild species. (8) PASCAL VOC (Everingham et al., 2010) is a classical objection detection and segmentation dataset containing 16.5k images for 20 classes. (9) COCO (Lin et al., 2014) is an objection detection and segmentation benchmark containing 118k scenic images with many objects for 80 classes.

A.2 Implementation of SAMix

Online training pipeline. We provide detailed implementation of SAMix in SL tasks. As shown in Figure 5 (left), we adopt the momentum pipeline (Grill et al., 2020; **?**) to optimize $\mathcal{L}_{\theta,\omega}$ for mixup classification and $\mathcal{L}_{\phi}$ for mixup generation in Eq. 5 in an end-to-end manner:

$$\theta_q^t, \omega_q^t \leftarrow \underset{\theta,\omega}{\operatorname{argmin}}\, \mathcal{L}_{\theta_q^{t-1},\omega_q^{t-1}}, \tag{12}$$

$$\phi^t \leftarrow \underset{\phi}{\operatorname{argmin}}\, \mathcal{L}_{\theta_k^t,\omega_k^t} + \mathcal{L}_{\phi^{t-1}}, \tag{13}$$

where t is the iteration step, θ_q, ω_q and θ_k, ω_k denote the parameters of online and momentum networks, respectively. The parameters in the momentum networks are an exponential moving average of the online networks with a momentum decay coefficient m, taking θ_k as an example,

$$\theta_k^t \leftarrow m\theta_k^{t-1} + (1-m)\theta_q^t. \tag{14}$$

The SAMix training process is summarized as four steps: (1) using the momentum encoder to generate the feature maps Z^l for Mixer $\mathcal{M}_\phi$; (2) generating X_{mix}^q and X_{mix}^k by Mixer for the online networks and Mixer; (3) training the online networks by Eq. 12 and the Mixer by Eq. 13 separately; (4) updating the momentum networks by Eq. 14.

Prior knowledge of mixup. As we discussed in Sec. 3.2, we introduce some prior knowledge to the Mixer from two aspects: (a) To adjust the mean of s_i correlated with λ, we introduce a mask loss that aligns the mean of s_i to λ, $\ell_\mu = \beta \max(|\lambda - \mu_i| - \epsilon, 0)$, where $\mu_i = \frac{1}{HW}\sum_{h,w} s_{i,h,w}$ is the mean and $\epsilon = 0.1$ as a margin. Meanwhile, we propose a test time λ *adjusting* method. Assuming $\mu_i < \lambda$, we adjust each coordinate on s_i as $\hat{s}_i = \frac{\mu_i}{\lambda} s_i$, and $\hat{s}_j = 1 - \hat{s}_i$. (b) To balance the smoothness of local image patches and the discrimination (e.g., variance) of x_m, we adopt a bilinear upsampling as $U(\cdot)$ for smoother masks and propose a variance loss to encourage the sparsity of learned masks, $\ell_\sigma = \frac{1}{WH}\sum_{w,h}(\mu_i - s_{w,h})^2$. We summarize the mask loss as, $\mathcal{L}_\phi^{mask} = \beta(\ell_\mu + \ell_\sigma)$, where β is a balancing weight. β is initialized to 0.1 and linearly decrease to 0 during training.

A.3 Supervised Image Classification

Hyper-parameter settings. As for hyper-parameters of SAMix, we follow the basic setting in AutoMix for both SL and SSL tasks: SAMix adopts $\alpha = 2$, the feature layer $l = 3$, the bilinear upsampling, and the weight $\beta = 0.1$ which linearly decays to 0. We use $\eta = 0.5$ for small-scale datasets (CIFAR-100, Tiny, CUB and Aircraft) and $\eta = 0.1$ for large-scale datasets (IN-1k and iNat2017). As for other methods, PuzzleMix (Kim et al., 2020), Co-Mixup (Kim et al., 2021), and AugMix (Hendrycks et al., 2019) are reproduced by their official implementations with $\alpha = 1, 2, 1$ for all datasets. As for mixup methods reproduced by us, we provide dataset-specific hyper-parameter settings as follows. For CIFAR-100, Mixup and ResizeMix use $\alpha = 1$, and CutMix, FMix and SaliencyMix use $\alpha = 0.2$, and ManifoldMix uses $\alpha = 2$. For Tiny, IN-1k, and iNat2017 datasets, ManifoldMix uses $\alpha = 0.2$, and the rest methods adopt $\alpha = 1$ for median and large backbones (e.g., ResNet-50). Specially, all these methods use $\alpha = 0.2$ (only) for ResNet-18. For small-scale fine-grained datasets (CUB-200 and Aircraft), SaliencyMix and FMix use $\alpha = 0.2$, and ManifoldMix uses $\alpha = 0.5$, while the rest use $\alpha = 1$.

A.4 Contrastive Learning

Implementation of SAMix-C and SAMix-I. As for SSL tasks, we adopt the cross-view objective, $\ell^{NCE}(z_i^{\tau_q}, z_i^{\tau_k}) + \ell^{NCE}(z_m)$, where $z_i = z_i^{\tau_k}$ and $z_j = z_j^{\tau_k}$, for instance-level mixup classification in all methods (except for † and ‡ marked methods). We provide two variants, SAMix-C and SAMix-I, which use different learning objectives of mixup classification. The basic network structures (an encoder f_θ and a projector g_ω) are adopted as MoCo.V2 (Chen et al., 2020b). Similar to SAMix in SL tasks, SAMix-C employ a parametric cluster classification head g_ψ^C for online clustering (Caron et al., 2018; Zhan et al., 2020) to provide pesudo labels (PL) to caculate $\mathcal{L}_\phi^{cls}$. It takes feature vectors from the momentum encoder as the input (optimized by Eq. 13) and has no impact on the mixup classification objective for the online networks. Meanwhile, SAMix-I employs the instance-level classification loss for both $\mathcal{L}_{\theta,\omega}$ and $\mathcal{L}_\phi^{cls}$. Moreover, we use the proposed η-balanced mixup loss $\mathcal{L}_\phi^{cls}$ for both SAMix-C and SAMix-I with $\eta = 0.5$ and the objective $\mathcal{L}_\phi$ for Mixer.

Hyper-parameter settings. Except for SwAV (Caron et al., 2020), all compared CL methods use MoCo.V2 pre-training settings, which adopts ResNet (He et al., 2016) as the encoder f_θ with two-layer MLP projector g_ω and is optimized by SGD optimizer and Cosine scheduler with the initial learning rate of 0.03 and the batch size of 256. The length of the momentum dictionary is 65536 for IN-1k and 16384 for STL-10 and Tiny datasets. The data augmentation strategy is based on IN-1k in MoCo.v2 as following: Geometric augmentation is `RandomResizedCrop` with the scale in $[0.2, 1.0]$ and `RandomHorizontalFlip`. Color augmentation is `ColorJitter` with $\{$brightness, contrast, saturation, hue$\}$ strength of $\{0.4, 0.4, 0.4, 0.1\}$ with a probability of 0.8, and `RandomGrayscale` with a probability of 0.2. Blurring augmentation is using square Gaussian kernel of size 23×23 with a std uniformly sampled in $[0.1, 2.0]$. We use 224×224 resolutions for IN-1k and 96×96 resolutions for STL-10 and Tiny datasets.

Evaluation protocols. We evaluate the SSL representation with a linear classification protocol proposed in MoCo (He et al., 2020), which trains a linear classifier on top of the frozen representation on the training set. The linear classifier is trained 100 epochs by an SGD optimizer with the SGD momentum of 0.9 and the weight decay of 0. We set the initial learning rate of 30 for IN-1k as MoCo, and 0.1 for STL-10 and Tiny. The learning rate decays by 0.1 at epochs 60 and 80. Moreover, we adopt object detection task to evaluate transfer learning abilities following MoCo, which uses the 4-th layer feature maps of ResNet (ResNet-C4) to fine-tune Faster R-CNN (Ren et al., 2015) with 24k iterations on the *trainval07+12* set and Mask R-CNN (He et al., 2017) with $2\times$ schedule on the *train2017* set.

A.5 Empirical Experiments

A.5.1 Cross-view Training Pipeline for Instance-level Mixup

We first analyze the learning objective of instance-level mixup classification for contrastive learning (CL) in Sec. 2.2. As shown in Figure 8 (a), there are two possible objectives for instance-level mixup

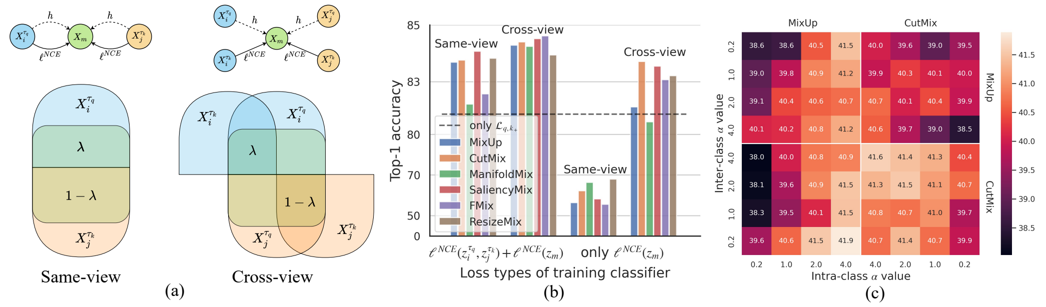

Figure 8: (a) Graphical models and information diagrams of *same-view* and *cross-view* training pipeline for instance-level mixup. Taking the *cross-view* as an example, $x_m = h(x_i^{\tau_2}, x_j^{\tau_2}, \lambda)$, the λ region denotes the corresponding information partition for $\lambda I(z_m^{\tau_q}, z_i^{\tau_k})$ and the $1-\lambda$ region for $(1-\lambda)I(z_m^{\tau_q}, z_j^{\tau_k})$. (b) Linear evaluation (top-1 accuracy on STL-10) of whether to use the cross-view pipeline and to combine the original infoNCE loss with the mixup infoNCE loss. (c) A heat map of linear evaluation (top-1 accuracy on Tiny) represents the effects of using MixUp and CutMix as the inter-class (y-axis) and intra-class mixup (x-axis) using various α.

defined in Eq. 4: the *same-view*, $\max_{\theta,\omega} \lambda I(z_m^{\tau_q}, z_i^{\tau_q}) + (1-\lambda)I(z_m^{\tau_q}, z_j^{\tau_q})$, and *cross-view* objective, $\max_{\theta,\omega} \lambda I(z_m^{\tau_q}, z_i^{\tau_k}) + (1-\lambda)I(z_m^{\tau_q}, z_j^{\tau_k})$. We hypothesize that the cross-view objective yields better CL performance than the same-view because the mutual information between two augmented view should be reduced while keeping task-relevant information (Tian et al., 2020b; Tsai et al., 2021). To verify this hypothesis, we design an experiment of various mixup methods with $\alpha = 1$ on STL-10 with ResNet-18. As shown in Figure 8 (b), we compare using the same-view or cross-view pipelines combined with using $\ell^{NCE}(z_i^{\tau_q}, z_i^{\tau_k}) + \ell^{NCE}(z_m)$ or only using $\ell^{NCE}(z_m)$. We can conclude: (i) Degenerated solutions occur when using the same-view pipeline while using the cross-view pipeline outperforms the CL baseline. It is mainly caused by degenerated mixed samples which contain parts of the same view of two source images. Therefore, we propose the cross-view pipeline for the instance-level mixup, where z_i and z_j in Eq. 4 are representations of $x_i^{\tau_k}$ and $x_j^{\tau_k}$. (ii) Combining both the original and mixup infoNCE loss, $\ell^{NCE}(z_i^{\tau_q}, z_i^{\tau_k}) + \ell^{NCE}(z_m)$, surpasses only using one of them, which indicates that mixup enables f_θ to learn relationship between local neighborhood systems.

A.5.2 Analysis of Instance-level Mixup

As we discussed in Sec. A.5.1, we propose the cross-view training pipeline for instance-level mixup classification. We then discuss inter- and intra-class proprieties of instance-level mixup. As shown in Figure 8 (c), we adopt inter-cluster and intra-cluster mixup from {Mixup, CutMix} with $\alpha \in \{0.2, 1, 2, 4\}$ to verify that instance-level mixup should treat inter- and intra-class mixup differently. Empirically, mixed samples provided by Mixup preserve global information of both source samples (smoother) while samples generated by CutMix preserve local patches (more discriminative). And we introduce pseudo labels (PL) to indicate different clusters by clustering method ODC (Zhan et al., 2020) with the class (cluster) number C. Based on experiment results, we can conclude that inter-class mixup requires *discriminative* mixed samples with *strong* intensity while the intra-class needs *smooth* samples with *low* intensity. Moreover, we provide two cluster-based instance-level mixup methods in Table 5 and 6 (denoting by $*$): (a) Inter-Intra*. We use CutMix with $\alpha \geq 2$ as inter-cluster mixup and Mixup with $\alpha = 0.2$ as intra-cluster mixup. (b) PuzzleMix*. We introduce saliency-based mixup methods to SSL tasks by introducing PL and a parametric cluster classifier g_ψ^C after the encoder. This classifier g_ψ^C and encoder f_θ are optimized online like AutoMix and SAMix mentioned in A.2. Based on gradCAM (Selvaraju et al., 2019) calculated from the classifier, PuzzleMix can be adopted on SSL tasks.

A.5.3 Analysis of Mixup Generation Objectives

In Sec. 3.1, we design experiments to analyze various losses for mixup generation in Figure 2 (left) and the proposed η-balanced loss in Figure 2 (right) for both SL and SSL tasks with ResNet-18 on STL-10 and Tiny. Basically, we assume both STL-10 and Tiny datasets have 200 classes on their

100k images. Since STL-10 does not provide ground truth labels (L) for 100k unlabeled data, we introduce PL generated by a supervised pertained classifier on Tiny as the ”ground truth” for its 100k training set. Notice that L denotes ground truth labels and PL denotes pseudo labels generated by ODC (Zhan et al., 2020) with $C = 200$.

As for the SL task, we use the labeled training set for mixup classification (100k on Tiny *v.s.* 5k on STL-10). Notice that SL results are worse than using SSL settings on STL-10, since the SL task only trains a randomly initialized classifier on 5k labeled data. Because the infoNCE and BCE loss require cross-view augmentation (or they will produce trivial solutions), we adopt MoCo.V2 augmentation settings for these two losses when performing the SL task. Compared to CE (L), we corrupt the global term in CE as CE (PL) or directly remove them as pBCE (L) to show that pBCE is vital to optimizing mixed samples. Similarly, we show that the global term is used as the global constrain by comparing BCE (UL) with infoNCE (UL), infoNCE (PL) and infoNCE (L). As for the SSL task, we verify the conclusions drawn from the SL task and conclude that (a) the local term optimizes mixup generation directly, corresponding to the smoothness property, (b) the global term serves as the global constraint corresponding to the discriminative property. Moreover, we verified that using the η-balanced loss as $\mathcal{L}_{\phi}^{cls}$ yields best performace on SL and SSL tasks. Notice that we use $\eta = 0.5$ on small-scale datasets and $\eta = 0.1$ on large-scale datasets for SL tasks, and use $\eta = 0.5$ for all SSL tasks.

A.5.4 Analysis of Mutual Information for Mixup

Since mutual information (MI) as usually adopted to analyze contrastive-based augmentations (Tian et al., 2020a;b), we estimate MI between x_m of various methods and x_i by MINE (Belghazi et al., 2018) with 100k images in 64×64 resolutions on Tiny-ImageNet. We sample $\lambda =$ from 0 to 1 with the step of 0.125 and plot results in Figure 7 (d). Here we see that SAMix-C and SAMix-I with more MI when $\lambda \approx 0.5$ perform better.

A.5.5 Results of Downstream tasks

In Sec. 4.2, we evaluate transferable abilities of the learned representation of self-supervised methods to object detection task on PASCAL VOC (Everingham et al., 2010) and COCO (Lin et al., 2014). In Table 9, the online SAMix-C and the pre-trained SAMix-I$^{\mathcal{P}}$ achieve the best detection performances among the compared methods and significently improves the baseline MoCo.V2 (*e.g.,* SAMix-C gains 0.9% AP and +0.7% AP^b over MoCo.V2). Notice that MoCHi, i-Mix, and UnMix introduce mixup augmentations in both the input and latent spaces, while our proposed SAMix only generates mixed samples in the input space.

		Faster R-CNN			Mask R-CNN		
CL Method	Methods	AP	AP$_{50}$	AP$_{75}$	AP^b	AP$^b_{50}$	AP$^b_{75}$
MoCo.V2	-	56.9	82.2	63.4	40.6	60.1	44.0
MoCo.V2	Mixup	57.4	82.5	64.0	41.0	60.8	44.3
MoCo.V2	CutMix	57.3	82.7	64.1	41.1	60.8	44.4
MoCo.V2	Inter-Intra*	57.5	82.8	64.2	41.2	60.9	44.4
MoCHi†	$input$+$latent$	57.1	82.7	64.1	41.0	60.8	44.5
i-Mix†	$input$+$latent$	57.5	82.7	64.2	-	-	-
UnMix‡	$input$+$latent$	57.7	83.0	64.3	41.2	60.9	**44.7**
WBSIM‡	$input$	57.4	82.8	64.2	40.7	60.8	44.2
MoCo.V2	**SAMix-I**	57.5	83.1	64.2	41.2	61.0	44.5
MoCo.V2	**SAMix-I$^{\mathcal{P}}$**	**57.8**	**83.2**	64.3	**41.3**	**61.1**	44.6
MoCo.V2	**SAMix-C**	57.7	83.1	**64.4**	**41.3**	**61.1**	**44.7**

Table 9: Transferring to object detection with Faster R-CNN on VOC and Mask R-CNN on COCO.

A.6 Visualization of SAMix

A.6.1 Mixing Attention and Content in Mixer

In Sec. 3.2, we discuss the trivial solutions of Mixer, which are usually occurred in SSL tasks. Given the sample pair (x_i, x_j) and $\lambda = 0.5$, we visualize the content C_i and $P_{i,j}$ to compare the trivial

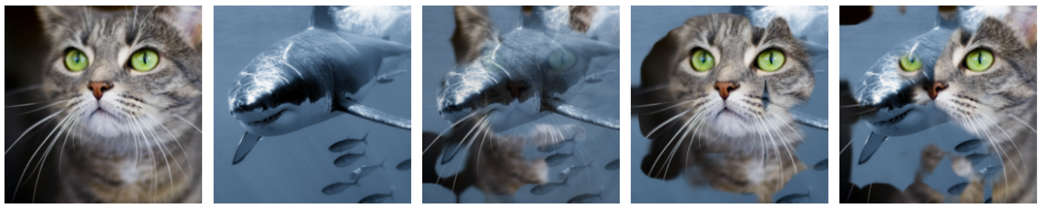

Figure 9: Visualization of loss effect. Both infoNCE and BCE loss have different emphases: infoNCE shows the similar effect of supervised fine-grained classification, focusing on fragmented and essential features, while BCE focuses on object completeness.

and non-trivial results in the SSL task on STL-10, as shown in Figure 3. As we can see, both C_i and $P_{i,j}$ from the trivial solution has extremely large or small scale values while C_i generated by C_{NCL} containing more balanced values. Since the attention weight $P_{i,j}$ is normalized by softmax, we hypothesize that C_i more likely causes trivial solutions. To verify our hypothesis, we freeze W_P in the original MB and compare the original linear content projection W_z with the non-linear content modeling. The results confirm that the non-linear module can prevent large-scale values on C_i and eliminate the trivial solutions.

A.6.2 Effects of Mixup Generation Loss

In addition to Sec.3.3, we further provide visualization of mixed samples using the infoNCE (Eq. 4), BCE (Eq. 7), and η-balanced infoNCE loss (Eq. 8) for Mixer. As shown in Figure 9, we find that mixed samples using infoNCE mixup loss prefer instance-specific and fine-grained features. On the contrary, mixed samples of the BCE loss seem only to consider discrimination between two corresponding neighborhood systems. It is more inclined to maintain the continuity of the whole object relative to infoNCE. Thus, combining both the characteristics, the η-balanced infoNCE loss yields mixed samples that retain both instance-specific features and global discrimination.

A.6.3 Visualization of Mixed Samples in SAMix

SAMix in various scenarios. In addition to Sec. 3.3, we visualize the mixed samples of SAMix in various scenarios to show the relationship between mixed samples and class (cluster) information. Since IN-1k contains some samples in CUB and Aircraft, we choose the overlapped samples to visualize SAMix trained for the fine-grained SL task (CUB and Aircraft) and SSL tasks (SAMix-I and SAMix-C). As shown in Figure 10, mixed samples reflect the granularity of class information adopted in mixup training. Specifically, we find that mixed samples using infoNCE mixup loss (Eq.4) is more closely to the fine-grained SL because they both have many fine-grained centroids.

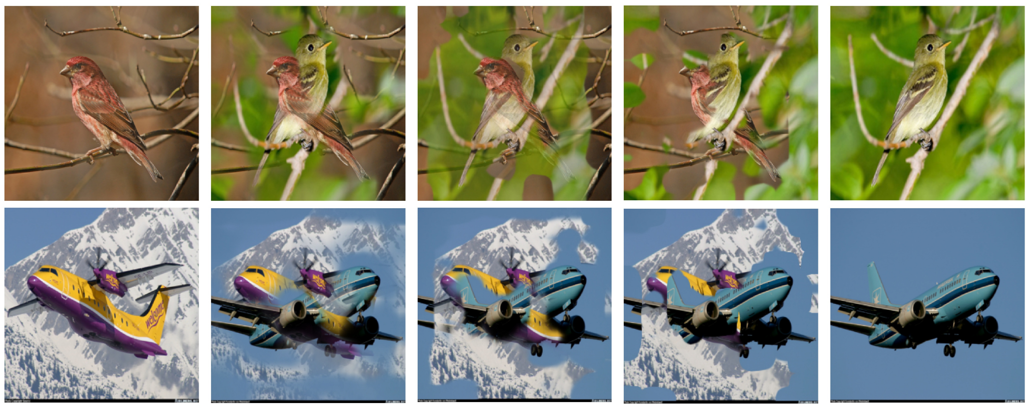

Figure 10: Visualization of SAMix in various scenarios on CUB and Aircraft. Given images A and B, the middle three mixed samples are generated by SAMix with $\lambda = 0.5$ trained in the fine-grained SL task and the SSL tasks (SAMix-C and SAMix-I).

Comparison with PuzzleMix in SL tasks. To highlight the accurate mixup relationship modeling in SAMix compared to PuzzleMix (standing for saliency-based methods), we visualize the results of mixed samples from these two methods in the supervised case in Figure 11. There is three main difference: (a) bilinear upsampling strategy in SAMix makes the mixed samples more smooth in local patches. (b) adaptive λ encoding and mixing attention enhance the correspondence between mixed samples and λ value. (c) η-balanced mixup loss enables SAMix to balance global discriminative and fine-grained features.

Comparison of SAMix-I and SAMix-C in SSL tasks. As shown in Figure 12, we provide more mixed samples of SAMix-I and SAMix-C in the SSL tasks to show that introducing class information by PL can help Mixer generate mixed samples that retain both the fine-grained features (instance discrimination) and whole targets.

A.7 DETAILED RELATED WORK

Contrastive Learning. CL amplifies the potential of SSL by achieving significant improvements on classification (Chen et al., 2020a; He et al., 2020; Caron et al., 2020; Chen et al., 2020c), which maximizes similarities of positive pairs while minimizing similarities of negative pairs. To provide a global view of CL, MoCo (He et al., 2020) proposes a memory-based framework with a large number of negative samples and model differentiation using the exponential moving average. SimCLR (Chen et al., 2020a) demonstrates a simple memory-free approach with large batch size and strong data augmentations that is also competitive in performance to memory-based methods. Unlike other CL approaches, BYOL (Grill et al., 2020) does not require negative pairs or a large batch size for the proposed pretext task, which tries to estimate latent representations from the same instance.

Mixup. MixUp (Zhang et al., 2018), convex interpolations of any two samples and their unique one-hot labels, were presented as the first mixing-based data augmentation approach for regularising the training of networks. ManifoldMix (Verma et al., 2019) and PatchUp (Faramarzi et al., 2020) expand it to the hidden space. CutMix (Yun et al., 2019) suggests a mixing strategy based on the patch of the image, *i.e.*, randomly replacing a local rectangular section in images. Based on CutMix, ResizeMix (Qin et al., 2020) inserts a whole image into a local rectangular area of another image after scaling down. FMix (Harris et al., 2020) converts the image to Fourier space (spectrum domain) to create binary masks. To generate more semantic virtual samples, offline optimization algorithms are introduced for the saliency regions. SaliencyMix (Uddin et al., 2020) obtains the saliency using a universal saliency detector. With optimization transportation, PuzzleMix (Kim et al., 2020) and Co-Mixup (Kim et al., 2021) present more precise methods for finding appropriate mixup masks based on saliency statistics. SuperMix (Dabouei et al., 2021) combines mixup with knowledge distillation, which learns a pixel-wise sample mixing policy via a teacher-student framework to distill class knowledge. Differing from previous methods, AutoMix (**?**) can learn the mixup generation by a sub-network end-to-end which generated mixed samples via feature maps and the mixing ratio.

Mixup for contrastive learning. A complementary method for better instance-level representation learning is to use mixup on CL (Kalantidis et al., 2020; Shen et al., 2021). When used in collaboration with CE loss, Mixup and its several variants provide highly efficient data augmentation for SL by establishing a relationship between samples. Without a ground-truth label, the most of approaches are limited to linear mixup methods. For example, Un-mix (Shen et al., 2021) attempts to use MixUp in the input space for self-supervised learning, whereas the developers of MoChi (Kalantidis et al., 2020) propose mixing the negative sample in the embedding space to increase the number of hard negatives but at the expense of classification accuracy. i-Mix (Lee et al., 2021) and BSIM (Chu et al., 2020) demonstrated how to regularize contrastive learning by mixing instances in the input or latent spaces. We introduce an automatic mixup for SSL tasks, which adaptively learns the instance relationship based on inter- and intra-cluster properties online.

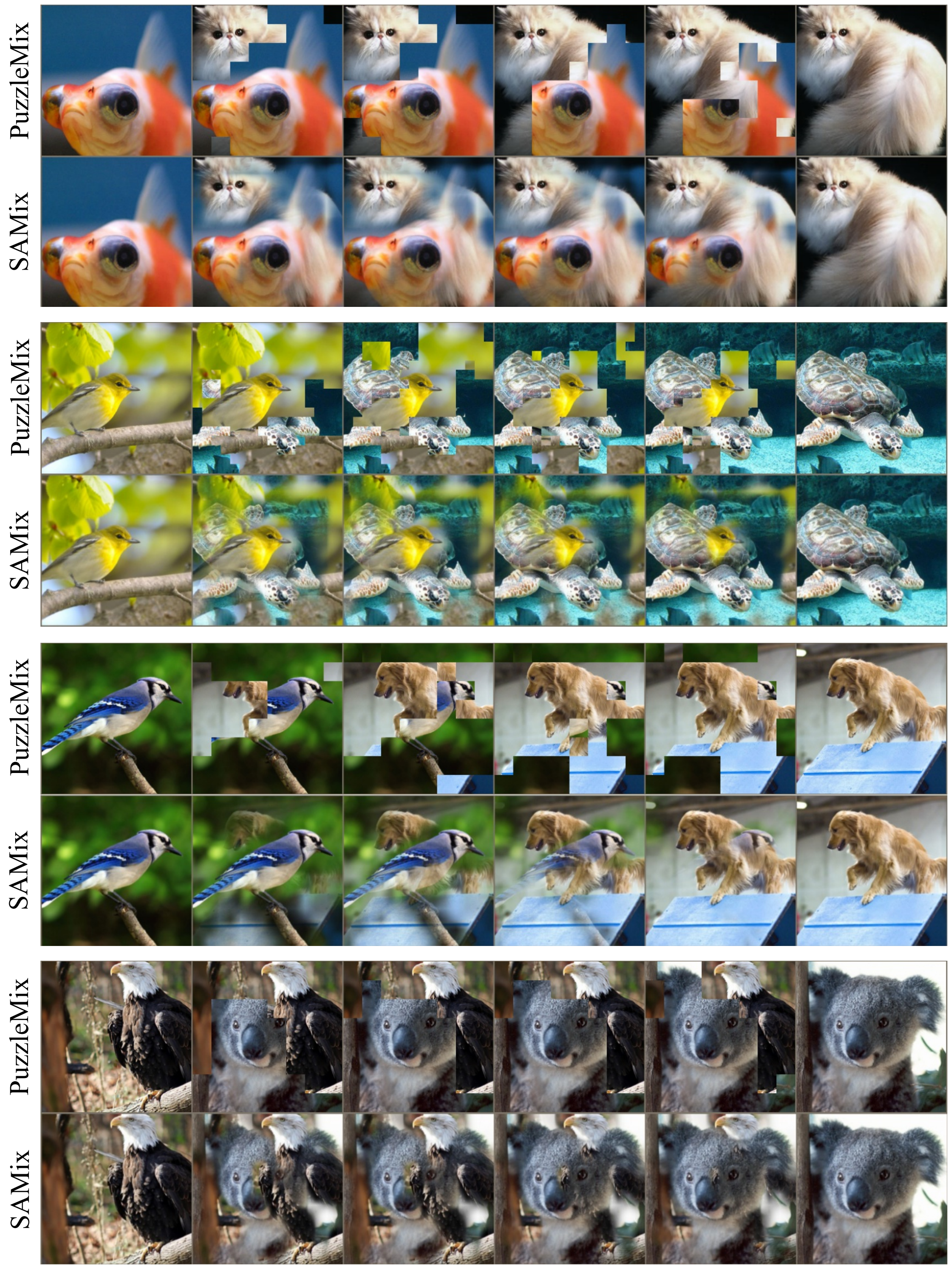

Figure 11: Visualization of PuzzleMix *v.s.* SAMix for SL tasks on IN-1k. In every four rows, the upper and lower two rows represent mixed samples generated by PuzzleMix and SAMix, respectively. λ value changes from left ($\lambda = 0$) to right ($\lambda = 1$) by an equal step.

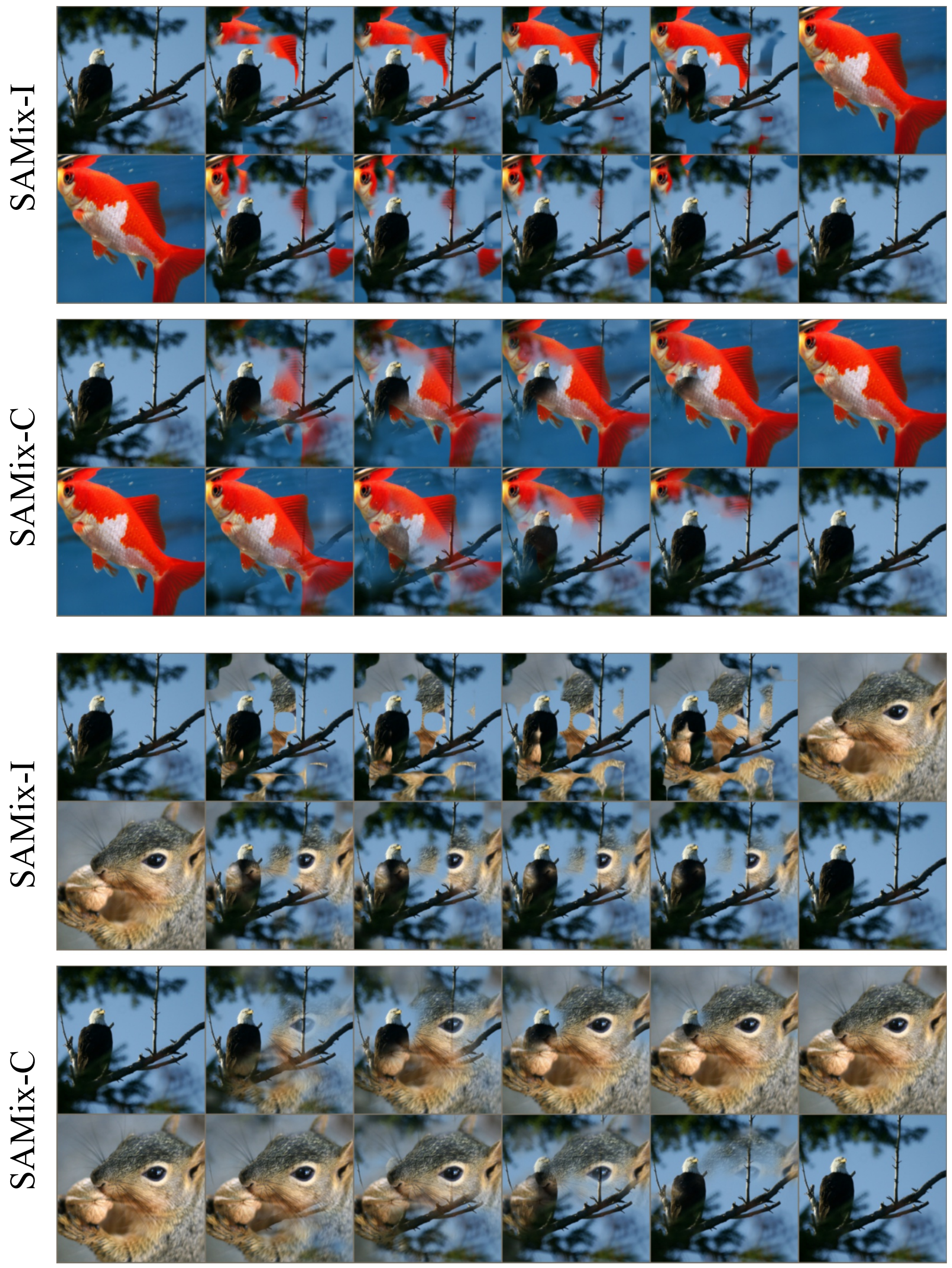

Figure 12: Visualization of SAMix-I *v.s.* SAMix-C for SSL tasks on IN-1k. In every four rows, the upper and lower two rows represent mixed samples generated by SAMix-I and SAMix-C, respectively. λ value changes from left ($\lambda = 0$) to right ($\lambda = 1$) by an equal step.