i-MIX: A STRATEGY FOR REGULARIZING CONTRASTIVE REPRESENTATION LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Contrastive representation learning has shown to be an effective way of learning representations from unlabeled data. However, much progress has been made in vision domains relying on data augmentations carefully designed using domain knowledge. In this work, we propose i-Mix, a simple yet effective regularization strategy for improving contrastive representation learning in both vision and non-vision domains. We cast contrastive learning as training a non-parametric classifier by assigning a unique virtual class to each data in a batch. Then, data instances are mixed in both the input and virtual label spaces, providing more augmented data during training. In experiments, we demonstrate that i-Mix consistently improves the quality of self-supervised representations across domains, resulting in significant performance gains on downstream tasks. Furthermore, we confirm its regularization effect via extensive ablation studies across model and dataset sizes. The code will be released.

1 INTRODUCTION

Representation learning (Bengio et al., 2013) is a fundamental task in machine learning since the success of machine learning relies on the quality of representation. Self-supervised representation learning (SSL) has been successfully applied in several domains, including image recognition (He et al., 2020; Chen et al., 2020a), natural language processing (Mikolov et al., 2013; Devlin et al., 2018), robotics (Sermanet et al., 2018; Lee et al., 2019), speech recognition (Ravanelli et al., 2020), and video understanding (Korbar et al., 2018; Owens & Efros, 2018). Since no label is available in the unsupervised setting, pretext tasks are proposed to provide self-supervision: for example, context prediction (Doersch et al., 2015), inpainting (Pathak et al., 2016), and contrastive learning (Wu et al., 2018b; Hjelm et al., 2019; He et al., 2020; Chen et al., 2020a). SSL has also been used as an auxiliary task to improve the performance on the main task, such as generative model learning (Chen et al., 2019), semi-supervised learning (Zhai et al., 2019), and improving robustness and uncertainty (Hendrycks et al., 2019).

Recently, contrastive representation learning has gained increasing attention by showing state-of-the-art performance in SSL for large-scale image recognition (He et al., 2020; Chen et al., 2020a), which outperforms its supervised pre-training counterpart (He et al., 2016) on downstream tasks. However, while the concept of contrastive learning is applicable to any domains, the quality of learned representations rely on the domain-specific inductive bias: as anchors and positive samples are obtained from the same data instance, data augmentation introduces semantically meaningful variance for better generalization. To achieve a strong, yet semantically meaningful data augmentation, domain knowledge is required, e.g., color jittering in 2D images or structural information in video understanding. Hence, contrastive representation learning in different domains requires an effort to develop effective data augmentations. Furthermore, while recent works have focused on large-scale settings where millions of unlabeled data is available, it would not be practical in real-world applications. For example, in lithography, acquiring data is very expensive in terms of both time and cost due to the complexity of manufacturing process (Lin et al., 2018; Sim et al., 2019).

Meanwhile, MixUp (Zhang et al., 2018) has shown to be a successful data augmentation for supervised learning in various domains and tasks, including image classification (Zhang et al., 2018), generative model learning (Lucas et al., 2018), and natural language processing (Guo et al., 2019; Guo, 2020).

In this paper, we explore the following natural, yet important question: is the idea of MixUp useful for unsupervised, self-supervised, or contrastive representation learning across different domains?

To this end, we propose *instance Mix (i-Mix)*, a data-driven augmentation for contrastive representation learning effective across different domains. The key idea of i-Mix is to introduce virtual labels in a batch and mix data instances and their corresponding virtual labels in the input and label spaces, respectively. We first introduce the general formulation of i-Mix, and then we show the applicability of i-Mix to state-of-the-art contrastive representation learning methods, SimCLR (Chen et al., 2020a) and MoCo (He et al., 2020), and a method without negative pairs, BYOL (Grill et al., 2020).

Through the experiments, we demonstrate the efficacy of i-Mix in a variety of settings. First, we show the effectiveness of i-Mix by evaluating the discriminative performance of learned representations in multiple domains. Specifically, we adapt i-Mix to the contrastive representation learning methods, advancing state-of-the-art performance across different domains, including image (Krizhevsky & Hinton, 2009; Deng et al., 2009), speech (Warden, 2018), and tabular (Asuncion & Newman, 2007) datasets. Then, we study i-Mix in various conditions, including when 1) the model and training dataset is small or large, 2) domain knowledge is limited, and 3) transfer learning.

Contribution. In summary, our contribution is three-fold:

- We propose i-Mix, a method for regularizing contrastive representation learning, motivated by MixUp (Zhang et al., 2018). We show how to apply i-Mix to state-of-the-art contrastive representation learning methods (Chen et al., 2020a; He et al., 2020; Grill et al., 2020).
- We show that i-Mix consistently improves contrastive representation learning in both vision and non-vision domains. In particular, the discriminative performance of representations learned with i-Mix is on par with fully supervised learning on CIFAR-10/100 (Krizhevsky & Hinton, 2009) and Speech Commands (Warden, 2018).
- We verify the regularization effect of i-Mix in a variety of settings. We empirically observed that i-Mix significantly improves contrastive representation learning when 1) the training dataset size is small, or 2) the domain knowledge for data augmentations is not enough.

2 Related Work

Self-supervised representation learning (SSL) aims at learning representations from unlabeled data by solving a pretext task that is derived from self-supervision. Early works on SSL proposed pretext tasks based on data reconstruction by autoencoding (Bengio et al., 2007), such as context prediction (Doersch et al., 2015) and inpainting (Pathak et al., 2016). Decoder-free SSL has made a huge progress in recent years. Exemplar CNN (Dosovitskiy et al., 2014) learns by classifying individual instances with data augmentations. SSL of visual representation, including colorization (Zhang et al., 2016), solving jigsaw puzzles (Noroozi & Favaro, 2016), counting the number of objects (Noroozi et al., 2017), rotation prediction (Gidaris et al., 2018), next pixel prediction (Oord et al., 2018; Hénaff et al., 2019), and combinations of them (Doersch & Zisserman, 2017; Kim et al., 2018; Noroozi et al., 2018) often leverages image-specific properties to design pretext tasks. Meanwhile, alhough deep clustering (Caron et al., 2018; 2019; Asano et al., 2020) is often distinguished from SSL, it also leverages clustering assignments as self-supervision for representation learning.

Contrastive representation learning has gained lots of attention for SSL (He et al., 2020; Chen et al., 2020a). As opposed to early works on exemplar CNN (Dosovitskiy et al., 2014; 2015), contrastive learning maximizes similarities of positive pairs while minimizes similarities of negative pairs instead of training an instance classifier. As the choice of negative pairs is crucial for the quality of learned representations, recent works have carefully designed them. Memory-based approaches (Wu et al., 2018b; Hjelm et al., 2019; Bachman et al., 2019; Misra & van der Maaten, 2020; Tian et al., 2020a) maintain a memory bank of embedding vectors of instances to keep negative samples, where the memory is updated with embedding vectors extracted from previous batches. In addition, MoCo (He et al., 2020) showed that differentiating the model for anchors and positive/negative samples is effective, where the model for positive/negative samples is updated by the exponential moving average of the model for anchors. On the other hand, recent works (Ye et al., 2019; Misra & van der Maaten, 2020; Chen et al., 2020a; Tian et al., 2020a) showed that learning invariance to different views is important in contrastive representation learning. The views can be generated

through data augmentations carefully designed using domain knowledge (Chen et al., 2020a), splitting input channels (Tian et al., 2020a), or borrowing the idea of other pretext tasks, such as creating jigsaw puzzles or rotating inputs (Misra & van der Maaten, 2020). In particular, SimCLR (Chen et al., 2020a) showed that a simple memory-free approach with a large batch size and strong data augmentations has a comparable performance to memory-based approaches. InfoMin (Tian et al., 2020b) further studied a way to generate good views for contrastive representation learning and achieved state-of-the-art performance by combining prior works. Different from other contrastive representation learning methods, BYOL (Grill et al., 2020) does not require negative pairs, where the proposed pretext task aims at predicting latent representations of one view from another. While prior works have focused on SSL on large-scale visual recognition tasks, our work focuses on contrastive representation learning in both small- and large-scale settings in different domains.

Data augmentation is a technique to increase the diversity of data, especially when training data are not enough for generalization. Since the augmented data must be understood as the original data, data augmentations are carefully designed using the domain knowledge about images (DeVries & Taylor, 2017b; Cubuk et al., 2019a;b; Zhong et al., 2020), speech (Amodei et al., 2016; Park et al., 2019), or natural languages (Zhang et al., 2015; Wei & Zou, 2019).

Some works have studied data augmentation with less domain knowledge: DeVries & Taylor (2017a) proposed a domain-agnostic augmentation strategy by first encoding the dataset and then applying augmentations in the feature space. MixUp (Zhang et al., 2018) is an effective data augmentation strategy in supervised learning, which performs vicinal risk minimization instead of empirical risk minimization, by linearly interpolating input data and their labels on the data and label spaces, respectively. On the other hand, MixUp has also shown its effectiveness in other tasks and non-vision domains, including generative adversarial networks (Lucas et al., 2018), improved robustness and uncertainty (Hendrycks et al., 2020), and sentence classification in natural language processing (Guo, 2020; Guo et al., 2019). Other variations have also been investigated by interpolating in the feature space (Verma et al., 2019) or leveraging domain knowledge (Yun et al., 2019). MixUp would not be directly applicable to some domains, such as point clouds, but its adaptation can be effective (Harris et al., 2020). i-Mix is a kind of data augmentation for better generalization in contrastive representation learning, resulting in better performances on downstream tasks.

3 Approach

In this section, we review MixUp (Zhang et al., 2018) in supervised learning and present i-Mix in contrastive learning (He et al., 2020; Chen et al., 2020a; Grill et al., 2020). Throughout this section, let $\mathcal{X}$ be a data space, $\mathbb{R}^D$ be a D-dimensional embedding space, and a model $f : \mathcal{X} \to \mathbb{R}^D$ be a mapping between them. For conciseness, $f_i = f(x_i)$ and $\tilde{f}_i = f(\tilde{x}_i)$ for $x_i, \tilde{x}_i \in \mathcal{X}$, and model parameters are omitted in loss functions.

3.1 MixUp in Supervised Learning

Suppose an one-hot label $y_i \in \{0,1\}^C$ is assigned to a data x_i, where C is the number of classes. Let a linear classifier predicting the labels consists of weight vectors $\{w_1, \dots, w_C\}$, where $w_c \in \mathbb{R}^D$.[1] Then, the cross-entropy loss for supervised learning is defined as:

$$\ell_{\text{Sup}}(x_i, y_i) = -\sum_{c=1}^{C} y_{i,c} \log \frac{\exp(w_c^\top f_i)}{\sum_{k=1}^{C} \exp(w_k^\top f_i)}. \tag{1}$$

While the cross-entropy loss is widely used for supervised training of deep neural networks, there are several challenges of training with the cross-entropy loss, such as preventing overfitting or networks being overconfident. Several regularization techniques have been proposed to alleviate these issues, including label smoothing (Szegedy et al., 2016), adversarial training (Miyato et al., 2018), and confidence calibration (Lee et al., 2018).

MixUp (Zhang et al., 2018) is an effective regularization with negligible computational overhead. It conducts a linear interpolation of two data instances in both input and label spaces and trains a model

[1] We omit bias terms for presentation clarity.

by minimizing the cross-entropy loss defined on the interpolated data and labels. Specifically, for two labeled data (x_i, y_i), (x_j, y_j), the MixUp loss is defined as follows:

$$\ell_{\text{Sup}}^{\text{MixUp}}\big((x_i, y_i), (x_j, y_j); \lambda\big) = \ell_{\text{Sup}}(\lambda x_i + (1-\lambda)x_j, \lambda y_i + (1-\lambda)y_j), \tag{2}$$

where $\lambda \sim \text{Beta}(\alpha, \alpha)$ is a mixing coefficient from a beta distribution. MixUp is a vicinal risk minimization method (Chapelle et al., 2001) that augments data and their labels in a data-driven manner. Not only improving the generalization on the supervised task, it also improves adversarial robustness (Zhang et al., 2018; Pang et al., 2019) and confidence calibration (Thulasidasan et al., 2019).

3.2 i-MIX IN CONTRASTIVE LEARNING

We introduce *instance mix (i-Mix)*, a data-driven augmentation strategy for contrastive representation learning to improve the generalization of learned representations. Intuitively, instead of mixing class labels, i-Mix interpolates their *virtual* labels, which indicates their identity in a batch.

Let $\mathcal{B} = \{(x_i, \tilde{x}_i)\}_{i=1}^N$ be a batch of data pairs, where N is the batch size, $x_i, \tilde{x}_i \in \mathcal{X}$ are two views of the same data, which are usually generated by different augmentations. For each anchor x_i, we call $\tilde{x}_i$ and $\tilde{x}_{j \neq i}$ positive and negative samples, respectively.[2] Then, the model f learns to maximize similarities of positive pairs (instances from the same data) while minimize similarities of negative pairs (instances from different data) in the embedding space. The output of f is L2-normalized, which has shown to be effective (Wu et al., 2018a; He et al., 2020; Chen et al., 2020a). Let $v_i \in \{0, 1\}^N$ be the virtual label of x_i and $\tilde{x}_i$ in a batch $\mathcal{B}$, where $v_{i,i} = 1$ and $v_{i,j \neq i} = 0$. For a general sample-wise contrastive loss with virtual labels $\ell(x_i, v_i)$, the i-Mix loss is defined as follows:

$$\ell^{i\text{-Mix}}\big((x_i, v_i), (x_j, v_j); \mathcal{B}, \lambda\big) = \ell(\text{Mix}(x_i, x_j; \lambda), \lambda v_i + (1-\lambda)v_j; \mathcal{B}), \tag{3}$$

where $\lambda \sim \text{Beta}(\alpha, \alpha)$ is a mixing coefficient and Mix is a mixing operator, which can be adapted depending on target domains: for example, $\text{MixUp}(x_i, x_j; \lambda) = \lambda x_i + (1-\lambda)x_j$ (Zhang et al., 2018) when data values are continuous, and $\text{CutMix}(x_i, x_j; \lambda) = M_\lambda \odot x_i + (1-M_\lambda) \odot x_j$ (Yun et al., 2019) when data values have a spatial correlation, where M_λ is a binary mask filtering out a region whose relative area is $(1-\lambda)$, and $\odot$ is an element-wise multiplication. In the following, we show how to apply i-Mix to contrastive representation learning methods. We use the MixUp operator for i-Mix formulations and experiments, unless otherwise specified.

SimCLR (Chen et al., 2020a) is a simple contrastive representation learning method without a memory bank, where each anchor has one positive sample and $(2N-2)$ negative samples. Let $x_{N+i} = \tilde{x}_i$ for conciseness. Then, the $(2N-1)$-way discrimination loss is written as follows:

$$\ell_{\text{SimCLR}}(x_i; \mathcal{B}) = -\log \frac{\exp\big(s(f_i, f_{(N+i) \bmod 2N})/\tau\big)}{\sum_{k=1, k\neq i}^{2N} \exp\big(s(f_i, f_k)/\tau\big)}, \tag{4}$$

where τ is a temperature scaling parameter and $s(f, \tilde{f}) = (f^\top \tilde{f})/\|f\|\|\tilde{f}\|$ is the inner product of two L2-normalized vectors. In this formulation, i-Mix is not directly applicable because virtual labels are defined differently for each anchor.[3] To resolve this issue, we simplify the formulation of SimCLR by excluding anchors from negative samples. Then, with virtual labels, the N-way discrimination loss is written as follows:

$$\ell_{\text{N-pair}}(x_i, v_i; \mathcal{B}) = -\sum_{n=1}^{N} v_{i,n} \log \frac{\exp\big(s(f_i, \tilde{f}_n)/\tau\big)}{\sum_{k=1}^{N} \exp\big(s(f_i, \tilde{f}_k)/\tau\big)}, \tag{5}$$

where we call it the **N-pair** contrastive loss, as the formulation is similar to the N-pair loss in the context of metric learning (Sohn, 2016).[4] For two data instances (x_i, v_i), (x_j, v_j) and a batch of data pairs $\mathcal{B} = \{(x_i, \tilde{x}_i)\}_{i=1}^N$, the i-Mix loss is defined as follows:

$$\ell_{\text{N-pair}}^{i\text{-Mix}}\big((x_i, v_i), (x_j, v_j); \mathcal{B}, \lambda\big) = \ell_{\text{N-pair}}(\lambda x_i + (1-\lambda)x_j, \lambda v_i + (1-\lambda)v_j; \mathcal{B}). \tag{6}$$

[2] Some literature (He et al., 2020; Chen et al., 2020b) refers to them as query and positive/negative keys.

[3] We present the application of i-Mix to the original SimCLR formulation in Appendix A.

[4] InfoNCE (Oord et al., 2018) is a similar loss inspired by the idea of noise-contrastive estimation (Gutmann & Hyvärinen, 2010).

Algorithm 1 Loss computation for i-Mix on N-pair contrastive learning in PyTorch-like style.

```
a, b = aug(x), aug(x) # two different views of input x
lam = Beta(alpha, alpha).sample() # mixing coefficient
randidx = randperm(len(x))
a = lam * a + (1-lam) * a[randidx]
logits = matmul(normalize(model(a)), normalize(model(b)).T) / t
loss = lam * CrossEntropyLoss(logits, arange(len(x))) + \
       (1-lam) * CrossEntropyLoss(logits, randidx)
```

Algorithm 1 provides the pseudocode of i-Mix on N-pair contrastive learning for one iteration.[5]

Pair relations in contrastive loss. To use contrastive loss for representation learning, one needs to properly define a pair relation $\{(x_i, \tilde{x}_i)\}_{i=1}^N$. For contrastive representation learning, where semantic class labels are not provided, the pair relation would be defined in that 1) a positive pair, x_i and $\tilde{x}_i$, are different views of the same data and 2) a negative pair, x_i and $\tilde{x}_{j\neq i}$, are different data instances. For supervised representation learning, x_i and $\tilde{x}_i$ are two data instances from the same class, while x_i and $\tilde{x}_{j\neq i}$ are from different classes. Note that two augmented versions of the same data also belong to the same class, so they can also be considered as a positive pair. i-Mix is not limited to self-supervised contrastive representation learning, but it can also be used as a regularization method for supervised contrastive representation learning (Khosla et al., 2020) or deep metric learning (Sohn, 2016; Movshovitz-Attias et al., 2017).

MoCo (He et al., 2020). In contrastive representation learning, the number of negative samples affects the quality of learned representations (Arora et al., 2019). Because SimCLR mines negative samples in the current batch, having a large batch size is crucial, which often requires a lot of computational resources (Chen et al., 2020a). For efficient training, recent works have maintained a memory bank $\mathcal{M} = \{\mu_k\}_{k=1}^K$, which is a queue of previously extracted embedding vectors, where K is the size of the memory bank (Wu et al., 2018b; He et al., 2020; Tian et al., 2020a;b). In addition, MoCo introduces an exponential moving average (EMA) model to extract positive and negative embedding vectors, whose parameters are updated as $\theta_{f^{\text{EMA}}} \leftarrow m\theta_{f^{\text{EMA}}} + (1-m)\theta_f$, where $m \in [0, 1)$ is a momentum coefficient and θ is model parameters. The loss is written as follows:

$$\ell_{\text{MoCo}}(x_i; \mathcal{B}, \mathcal{M}) = -\log \frac{\exp\left(s(f_i, \tilde{f}_i^{\text{EMA}})/\tau\right)}{\exp\left(s(f_i, \tilde{f}_i^{\text{EMA}})/\tau\right) + \sum_{k=1}^K \exp\left(s(f_i, \mu_k)/\tau\right)}. \tag{7}$$

The memory bank $\mathcal{M}$ is then updated with $\{\tilde{f}_i^{\text{EMA}}\}$ in the first-in first-out order. In this $(K+1)$-way discrimination loss, data pairs are independent to each other, such that i-Mix is not directly applicable because virtual labels are defined differently for each anchor. To overcome this issue, we include the positive samples of other anchors as negative samples, similar to the N-pair contrastive loss in Eq. (5). Let $\tilde{v}_i \in \{0, 1\}^{N+K}$ be a virtual label indicating the positive sample of each anchor, where $\tilde{v}_{i,i} = 1$ and $\tilde{v}_{i,j\neq i} = 0$. Then, the $(N+K)$-way discrimination loss is written as follows:

$$\ell_{\text{MoCo}}(x_i, \tilde{v}_i; \mathcal{B}, \mathcal{M}) = -\sum_{n=1}^N \tilde{v}_{i,n} \log \frac{\exp\left(s(f_i, \tilde{f}_n^{\text{EMA}})/\tau\right)}{\sum_{k=1}^N \exp\left(s(f_i, \tilde{f}_k^{\text{EMA}})/\tau\right) + \sum_{k=1}^K \exp\left(s(f_i, \mu_k)/\tau\right)}. \tag{8}$$

As virtual labels are bounded in the same set in this formulation, i-Mix is directly applicable: for two data instances $(x_i, \tilde{v}_i)$, $(x_j, \tilde{v}_j)$, a batch of data pairs $\mathcal{B} = \{(x_i, \tilde{x}_i)\}_{i=1}^N$, and the memory bank $\mathcal{M}$, the i-Mix loss is defined as follows:

$$\ell_{\text{MoCo}}^{i\text{-Mix}}\big((x_i, \tilde{v}_i), (x_j, \tilde{v}_j); \mathcal{B}, \mathcal{M}, \lambda\big) = \ell_{\text{MoCo}}(\lambda x_i + (1-\lambda)x_j, \lambda\tilde{v}_i + (1-\lambda)\tilde{v}_j; \mathcal{B}, \mathcal{M}). \tag{9}$$

BYOL. Different from other contrastive representation learning methods, BYOL is a self-supervised representation learning method without contrasting negative pairs. For two views of the same data

[5]For losses linear with respect to labels (e.g., the cross-entropy loss), they are equivalent to $\lambda\ell(\lambda x_i + (1-\lambda)x_j, v_i) + (1-\lambda)\ell(\lambda x_i + (1-\lambda)x_j, v_j)$, i.e., optimization to the mixed label is equivalent to joint optimization to original labels. The proof for losses in contrastive learning methods is provided in Appendix B.

$x_i, \tilde{x}_i \in \mathcal{X}$, the model f learns to predict a view embedded with the EMA model $\tilde{f}_i^{\text{EMA}}$ from its embedding f_i. Specifically, an additional prediction layer g is introduced, such that the difference between $g(f_i)$ and $\tilde{f}_i^{\text{EMA}}$ is learned to be minimized. The BYOL loss is written as follows:

$$\ell_{\text{BYOL}}(x_i, \tilde{x}_i) = \left\| g(f_i)/\|g(f_i)\| - \tilde{f}_i/\|\tilde{f}_i\| \right\|^2 = 2 - 2 \cdot s(g(f_i), \tilde{f}_i). \tag{10}$$

This formulation can be represented in the form of the general contrastive loss in Eq. (3), as the second view $\tilde{x}_i$ can be accessed from the batch $\mathcal{B}$ with its virtual label v_i. To derive i-Mix in BYOL, let $\tilde{F} = [\tilde{f}_1/\|\tilde{f}_1\|, \ldots, \tilde{f}_N/\|\tilde{f}_N\|] \in \mathbb{R}^{D \times N}$ be the collection of L2-normalized embedding vectors of the second views, such that $\tilde{f}_i/\|\tilde{f}_i\| = \tilde{F} v_i$. Then, the BYOL loss is written as follows:

$$\ell_{\text{BYOL}}(x_i, v_i; \mathcal{B}) = \left\| g(f_i)/\|g(f_i)\| - \tilde{F} v_i \right\|^2 = 2 - 2 \cdot s(g(f_i), \tilde{F} v_i). \tag{11}$$

For two data instances (x_i, v_i), (x_j, v_j) and a batch of data pairs $\mathcal{B} = \{(x_i, \tilde{x}_i)\}_{i=1}^N$, the i-Mix loss is defined as follows:

$$\ell_{\text{BYOL}}^{i\text{-Mix}}\big((x_i, v_i), (x_j, v_j); \mathcal{B}, \lambda\big) = \ell_{\text{BYOL}}(\lambda x_i + (1-\lambda) x_j, \lambda v_i + (1-\lambda) v_j; \mathcal{B}). \tag{12}$$

3.3 InputMix

The contribution of data augmentations to the quality of learned representations is crucial in contrastive representation learning. For the case when the domain knowledge about efficient data augmentations is limited, we propose to apply InputMix together with i-Mix, which mixes input data but not their labels. This method can be viewed as introducing structured noises driven by auxiliary data to the principal data with the largest mixing coefficient λ, and the label of the principal data is assigned to the mixed data. We applied InputMix and i-Mix together on image datasets in Table 3.

4 Experiments

In this section, we demonstrate the effectiveness of i-Mix. In all experiments, we conduct contrastive representation learning on a pretext dataset and evaluate the quality of representations via supervised classification on a downstream dataset. We report the accuracy averaged over up to five runs. In the first stage, a convolutional neural network (CNN) or multilayer perceptron (MLP) followed by the two-layer MLP projection head is trained on an unlabeled dataset. Then, we replace the projection head with a linear classifier and train only the linear classifier on a labeled dataset for downstream task. Except for transfer learning, datasets for the pretext and downstream tasks are the same. For i-Mix, we sample a mixing coefficient $\lambda \sim \text{Beta}(\alpha, \alpha)$ for each data, where $\alpha = 1$ for image and speech datasets and $\alpha = 2$ for tabular datasets unless otherwise stated.[6] Additional details for the experimental settings and more experiments can be found in Appendix C.

4.1 Experimental Setup

Baselines and datasets. We consider 1) N-pair contrastive learning as a memory-free contrastive learning method,[7] 2) MoCo v2 (He et al., 2020; Chen et al., 2020b) [8] as a memory-based contrastive learning method, and 3) BYOL (Grill et al., 2020), which is a self-supervised learning method without negative pairs. We apply i-Mix to these methods and compare their performances. To show the effectiveness of i-Mix across domains, we evaluate the methods on datasets from multiple domains, including image, speech, and tabular datasets.

CIFAR-10/100 (Krizhevsky & Hinton, 2009) consist of 50k training and 10k test images, and ImageNet (Deng et al., 2009) has 1.3M training and 50k validation images, where we use them for evaluation. For some experiments, we use ImageNet-100, a subset with randomly chosen 100 classes out of 1k classes. We apply a set of data augmentations randomly in sequence including random resized cropping, horizontal flipping, color jittering, gray scaling, and Gaussian blurring for

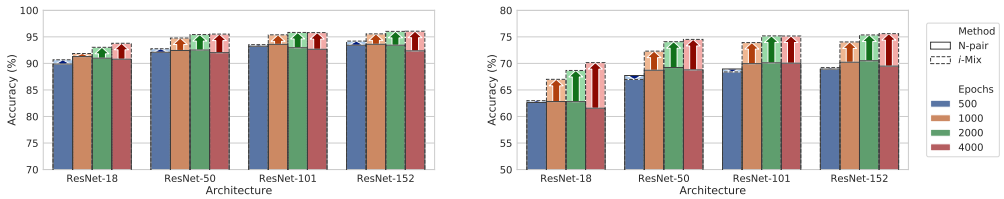
[6] Beta(α, α) is the uniform distribution when $\alpha = 1$, bell-shaped when $\alpha > 1$, and bimodal when $\alpha < 1$.

[7] We use the N-pair formulation in Eq. (5) instead of SimCLR as it is simpler and more efficient to integrate i-Mix. As shown in Appendix C.2, the N-pair formulation results in no worse performance than SimCLR.

[8] MoCo v2 improves the performance of MoCo by cosine learning schedule and more data augmentations.

Domain	Dataset	N-pair	+ i-Mix	MoCo v2	+ i-Mix	BYOL	+ i-Mix
Image	CIFAR-10	92.5 $\pm$ 0.3	**95.5** $\pm$ 0.1	93.3 $\pm$ 0.1	**95.9** $\pm$ 0.1	94.1 $\pm$ 0.1	**96.1** $\pm$ 0.1
	CIFAR-100	69.2 $\pm$ 0.2	**74.5** $\pm$ 0.2	71.7 $\pm$ 0.3	**77.8** $\pm$ 0.1	73.3 $\pm$ 0.4	**78.9** $\pm$ 0.2
Speech	Commands	83.7 $\pm$ 0.4	**91.1** $\pm$ 0.3	96.0 $\pm$ 0.1	**98.2** $\pm$ 0.2	96.4 $\pm$ 0.1	**97.9** $\pm$ 0.1
Tabular	CovType	65.8 $\pm$ 0.3	**69.5** $\pm$ 0.6	66.4 $\pm$ 0.2	**69.1** $\pm$ 0.5	65.9 $\pm$ 0.2	**69.5** $\pm$ 0.2

Table 1: Comparison of contrastive representation learning methods and i-Mix in different domains.



(a) CIFAR-10 (b) CIFAR-100

Figure 2: Comparison of performance gains by applying i-Mix to N-pair contrastive learning with different model sizes and number of epochs on CIFAR-10 and 100.

ImageNet, which has shown to be effective (Chen et al., 2020a;b). We use ResNet-50 (He et al., 2016) as a backbone network. Models are trained with a batch size of 256 (i.e., 512 including augmented data) for up to 4000 epochs on CIFAR-10 and 100, and with a batch size of 512 for 800 epochs on ImageNet. For ImageNet experiments, we use the CutMix (Yun et al., 2019) version of i-Mix.

The Speech Commands dataset (Warden, 2018) contains 51k training, 7k validation, and 7k test data in 12 classes. We apply a set of data augmentations randomly in sequence: changing amplitude, speed, and pitch in time domain, stretching, time shifting, and adding background noise in frequency domain. Augmented data are then transformed to a 32×32 mel spectogram. We use the same architecture with image experiments. Models are trained with a batch size of 256 for 500 epochs.

For tabular dataset experiments, we consider Forest Cover Type (CovType) and Higgs Boson (Higgs) from UCI repository (Asuncion & Newman, 2007). CovType contains 15k training and validation, and 566k test data in 7 classes, and Higgs contains 10.5M training and 0.5M test data for binary classification. For Higgs, we also use a subset of 100k training data to experiment at a different scale. Since the domain knowledge for data augmentations on tabular data is limited, only a masking noise with the probability 0.2 is considered as a data augmentation. We use a 5-layer MLP with batch normalization (Ioffe & Szegedy, 2015) as a backbone network. Models are trained with a batch size of 512 for 1000 epochs.

4.2 Main Results

Table 1 shows the wide applicability of i-Mix to state-of-the-art contrastive representation learning methods for multiple domains. i-Mix results in consistent improvements on the classification accuracy, e.g., up to 6.1% when i-Mix is applied to MoCo v2 on CIFAR-100. Interestingly, we observe that linear classifiers on top of representations learned with i-Mix without fine-tuning the pre-trained part often yield a classification accuracy on par with simple end-to-end supervised learning from random initialization,[9] e.g., i-Mix vs. end-to-end supervised learning performance is 96.1% vs. 95.5% on CIFAR-10, 78.9% for both on CIFAR-100, and 98.2% vs. 98.0% on Speech Commands.

4.3 Regularization Effect of i-Mix

A better regularization method often benefits from longer training of deeper models, which is more critical when training on a small dataset. To investigate the regularization effect of i-Mix, we first

[9] Supervised learning with regularization methods, e.g., MixUp, outperforms i-Mix here. However, linear evaluation on top of self-supervised representation learning is a proxy to measure the quality of representations learned without labels, such that it is not supposed to be compared with the performance of supervised learning.

Domain	Dataset	MoCo v2	+ i-Mix
Image	ImageNet-100	84.1	**87.0**
	ImageNet-1k	70.9	**71.3**

Domain	Dataset	N-pair	+ i-Mix
Tabular	Higgs (100k)	73.3	**73.5**
	Higgs (10M)	**76.3**	75.4

Table 2: Comparison of contrastive learning and i-Mix on large-scale datasets.

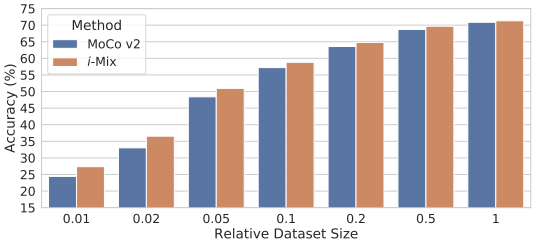


Figure 3: Comparison of MoCo v2 and i-Mix trained on the different size of ImageNet.

Aug	CIFAR-10		CIFAR-100		Speech Commands		CovType		Higgs (100k)		Higgs (10M)	
	N-pair	+ i-Mix*	N-pair	+ i-Mix*	N-pair	+ i-Mix	N-pair	+ i-Mix	N-pair	+ i-Mix	N-pair	+ i-Mix
-	17.0 ± 3.4	**79.7** ± 0.1	3.0 ± 0.8	**50.7** ± 0.3	62.4 ± 0.1	**72.7** ± 0.5	41.6 ± 0.5	**68.5** ± 0.3	58.8	**72.6**	56.0	**74.6**
✓	92.5 ± 0.3	**95.5** ± 0.1	69.2 ± 0.2	**74.5** ± 0.2	83.7 ± 0.4	**91.1** ± 0.3	65.8 ± 0.3	**69.5** ± 0.6	73.3	**73.5**	**76.3**	75.4

Table 3: Comparison of contrastive learning and i-Mix with and without data augmentation.

*InputMix is applied.

make a comparison between N-pair contrastive learning and i-Mix by training with different model sizes and number of training epochs on the pretext task. We train ResNet-18, 50, 101, and 152 models with varying number of training epochs from 500 to 4000. Figure 2 shows the performance of N-pair contrastive learning (solid box) and i-Mix (dashed box). When models are trained for 500 epochs, the performance of i-Mix is just on par with the baseline. However, i-Mix improves the performance by significant margins when trained longer. This is because i-Mix introduces more variance to the pretext dataset, such that it requires more update to learn them. In addition, deeper models benefit from strong regularization of i-Mix, achieving 96.1% on CIFAR-10 and 75.6% on CIFAR-100 using ResNet-152 after 4000 epochs of training. On the other hand, models trained without i-Mix start to show decrease in performance, possibly due to overfitting to the pretext task when trained longer. The trend clearly shows that i-Mix results in better representations via improved regularization.

Next, we study the effect of i-Mix with varying dataset sizes for the pretext tasks. Table 2 shows the effect of i-Mix on large-scale datasets[10] from image and tabular domains. We observe that i-Mix is particularly effective when the amount of training data is reduced, e.g., ImageNet-100 consists of images from 100 classes, thus has only 10% of training data compared to ImageNet-1k. However, the performance gain is reduced when the amount of training data is large. we further study representations learned with different pretext dataset sizes from 1% to 100% of the ImageNet training data in Figure 3. Here, different from ImageNet-100, we reduce the amount of data for each class, but maintain the number of classes the same. We observe that the performance gain by i-Mix is more significant when the size of the pretext dataset is small. Our study suggests that i-Mix is effective for regularizing self-supervised representations when training from a limited amount of data. We believe that this is aligned with findings in Zhang et al. (2018) for MixUp in supervised learning. Finally, when large-scale unlabeled data is available, we expect i-Mix would still be useful in obtaining better representations when trained longer with deeper and larger models.

4.4 Contrastive Learning without Domain-Specific Data Augmentation

Data augmentations play a key role in contrastive representation learning, and therefore it raises a question when applying them to domains with a limited or no knowledge of such augmentations. In this section, we study the effectiveness of i-Mix as a data-driven augmentation strategy for contrastive representation learning, which can be adapted to different domains. Table 3 shows the performance of N-pair contrastive learning and i-Mix with and without data augmentations. We observe significant performance gains with i-Mix when other data augmentations are not applied. For example, compared to the accuracy of 92.5% on the downstream task when other data augmentations are applied, contrastive learning achieves only 17.0% when trained without any data augmentations. This suggests that data augmentation is an essential part for the success of contrastive representation

[10]Here, "scale" corresponds to the amount of data rather than image resolution.

Pretext	CIFAR-10		CIFAR-100	
Downstream	N-pair	+ i-Mix	N-pair	+ i-Mix
CIFAR-10	92.5 $\pm$ 0.3	**95.5** $\pm$ 0.1	84.9 $\pm$ 0.3	**86.9** $\pm$ 0.1
CIFAR-100	61.8 $\pm$ 0.3	**65.1** $\pm$ 0.3	69.2 $\pm$ 0.2	**74.5** $\pm$ 0.2

(a) CIFAR-10 and 100 as the pretext dataset

VOC Object Detection	ImageNet	
	MoCo v2	+ i-Mix
AP	57.3 $\pm$ 0.1	**57.5** $\pm$ 0.4
AP_{50}	82.5 $\pm$ 0.2	**82.7** $\pm$ 0.2
AP_{75}	63.8 $\pm$ 0.3	**64.2** $\pm$ 0.7

(b) ImageNet as the pretext dataset

Table 4: Comparison of contrastive learning and i-Mix in transfer learning.

learning (Chen et al., 2020a). However, i-Mix is able to learn meaningful representations without other data augmentations and achieves close to the accuracy of 80% on CIFAR-10.

In Table 3, InputMix is applied together with i-Mix to further improve the performance on image datasets. For each principal data, we mix two auxiliary data, with mixing coefficients ($0.5\lambda_1 + 0.5$, $0.5\lambda_2$, $0.5\lambda_3$), where $\lambda_1, \lambda_2, \lambda_3 \sim \text{Dirichlet}(1, 1, 1)$.[11] In the above example, while i-Mix is better than baselines, adding InputMix further improves the performance of i-Mix, i.e., from 49.6% to 79.7% on CIFAR-10, and from 24.3% to 50.7% on CIFAR-100. This confirms that InputMix can further improve the performance when domain-specific data augmentations are not available, as discussed in Section 3.3.

Moreover, we verify its effectiveness on other domains beyond the image domain. For example, the performance improves from 62.4% to 72.7% on the Speech Commands dataset when we assume no other data augmentations are available. We also observe consistent improvements in accuracy for tabular datasets, even when the training dataset size is large. Although the domain knowledge for data augmentations is important to achieve state-of-the-art results, our demonstration shows the potential of i-Mix to be used for a wide range of application domains where domain knowledge is particularly limited.

4.5 Transferability of i-Mix

In this section, we show the improved transferability of the representations learned with i-Mix. The results are provided in Table 4. First, we train linear classifiers with downstream datasets different from the pretext dataset used to train backbone networks and evaluate their performance, e.g., CIFAR-10 as pretext and CIFAR-100 as downstream datasets or vice versa. We observe consistent performance gains when learned representations from one dataset are evaluated on classification tasks of another dataset. Next, we transfer representations trained on ImageNet to the PASCAL VOC object detection task (Everingham et al., 2010). We follow the settings in prior works (He et al., 2020; Chen et al., 2020b): the parameters of the pre-trained ResNet-50 are transferred to a Faster R-CNN detector with the ResNet50-C4 backbone (Ren et al., 2015), and fine-tuned end-to-end on the VOC 07+12 trainval dataset and evaluated on the VOC 07 test dataset. We report the average precision (AP) averaged over IoU thresholds between 50% to 95% at a step of 5%, and AP_{50} and AP_{75}, which are AP values when IoU threshold is 50% and 75%, respectively. Similar to Table 2, we observe small but consistent performance gains in all metrics. Those results confirm that i-Mix improves the quality of learned representations, such that performances on downstream tasks are improved.

5 Conclusion

We propose i-Mix, a data-driven augmentation strategy applicable to a class of self-supervised learning. The key idea of i-Mix is to introduce a virtual label to each data instance, and mix both inputs and the corresponding virtual labels. We show that i-Mix is applicable to state-of-the-art self-supervised representation learning methods including SimCLR, MoCo, and BYOL, which consistently improves the performance in a variety of settings and domains. Our experimental results indicate that i-Mix is particularly effective when the training dataset size is small or data augmentation is not available, each of which are prevalent in practice.

[11] This guarantees that the mixing coefficient for the principal data is larger than 0.5 to prevent from training with noisy labels. Note that Beckham et al. (2019) also sampled mixing coefficients from the Dirichlet distribution for mixing more than two data.

REFERENCES

Dario Amodei, Sundaram Ananthanarayanan, Rishita Anubhai, Jingliang Bai, Eric Battenberg, Carl Case, Jared Casper, Bryan Catanzaro, Qiang Cheng, Guoliang Chen, et al. Deep speech 2: End-to-end speech recognition in english and mandarin. In *ICML*, 2016.

Sanjeev Arora, Hrishikesh Khandeparkar, Mikhail Khodak, Orestis Plevrakis, and Nikunj Saunshi. A theoretical analysis of contrastive unsupervised representation learning. In *ICML*, 2019.

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

Arthur Asuncion and David Newman. Uci machine learning repository, 2007.

Philip Bachman, R Devon Hjelm, and William Buchwalter. Learning representations by maximizing mutual information across views. In *NeurIPS*, 2019.

Christopher Beckham, Sina Honari, Vikas Verma, Alex M Lamb, Farnoosh Ghadiri, R Devon Hjelm, Yoshua Bengio, and Chris Pal. On adversarial mixup resynthesis. In *NeurIPS*, 2019.

Yoshua Bengio, Pascal Lamblin, Dan Popovici, and Hugo Larochelle. Greedy layer-wise training of deep networks. In *NIPS*, 2007.

Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *PAMI*, 35(8):1798–1828, 2013.

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

Mathilde Caron, Piotr Bojanowski, Julien Mairal, and Armand Joulin. Unsupervised pre-training of image features on non-curated data. In *ICCV*, 2019.

Olivier Chapelle, Jason Weston, Léon Bottou, and Vladimir Vapnik. Vicinal risk minimization. In *NIPS*, 2001.

Ting Chen, Xiaohua Zhai, Marvin Ritter, Mario Lucic, and Neil Houlsby. Self-supervised gans via auxiliary rotation loss. In *CVPR*, 2019.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. *arXiv preprint arXiv:2002.05709*, 2020a.

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

Ekin D Cubuk, Barret Zoph, Dandelion Mane, Vijay Vasudevan, and Quoc V Le. Autoaugment: Learning augmentation policies from data. In *CVPR*, 2019a.

Ekin D Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V Le. Randaugment: Practical data augmentation with no separate search. *arXiv preprint arXiv:1909.13719*, 2019b.

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.

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.

Terrance DeVries and Graham W Taylor. Dataset augmentation in feature space. *arXiv preprint arXiv:1702.05538*, 2017a.

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

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

Carl Doersch, Abhinav Gupta, and Alexei A Efros. Unsupervised visual representation learning by context prediction. In *ICCV*, 2015.

Alexey Dosovitskiy, Jost Tobias Springenberg, Martin Riedmiller, and Thomas Brox. Discriminative unsupervised feature learning with convolutional neural networks. In *NeurIPS*, 2014.

Alexey Dosovitskiy, Philipp Fischer, Jost Tobias Springenberg, Martin Riedmiller, and Thomas Brox. Discriminative unsupervised feature learning with exemplar convolutional neural networks. *PAMI*, 38(9):1734–1747, 2015.

Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *IJCV*, 88(2):303–338, 2010.

Maurice Fréchet. Sur la distance de deux lois de probabilité. *COMPTES RENDUS HEBDOMADAIRES DES SEANCES DE L ACADEMIE DES SCIENCES*, 244(6):689–692, 1957.

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

Ian J Goodfellow, David Warde-Farley, Mehdi Mirza, Aaron Courville, and Yoshua Bengio. Maxout networks. In *ICML*, 2013.

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.

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. *arXiv preprint arXiv:2006.07733*, 2020.

Hongyu Guo. Nonlinear mixup: Out-of-manifold data augmentation for text classification. In *AAAI*, 2020.

Hongyu Guo, Yongyi Mao, and Richong Zhang. Augmenting data with mixup for sentence classification: An empirical study. *arXiv preprint arXiv:1905.08941*, 2019.

Michael Gutmann and Aapo Hyvärinen. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In *AISTATS*, 2010.

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*, 2020.

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

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

Olivier J Hénaff, Ali Razavi, Carl Doersch, SM Eslami, and Aaron van den Oord. Data-efficient image recognition with contrastive predictive coding. *arXiv preprint arXiv:1905.09272*, 2019.

Dan Hendrycks, Mantas Mazeika, Saurav Kadavath, and Dawn Song. Using self-supervised learning can improve model robustness and uncertainty. In *NeurIPS*, 2019.

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. In *ICLR*, 2020.

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *NeurIPS*, 2017.

R Devon Hjelm, Alex Fedorov, Samuel Lavoie-Marchildon, Karan Grewal, Phil Bachman, Adam Trischler, and Yoshua Bengio. Learning deep representations by mutual information estimation and maximization. In *ICLR*, 2019.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*, 2015.

Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. Supervised contrastive learning. *arXiv preprint arXiv:2004.11362*, 2020.

Dahun Kim, Donghyeon Cho, Donggeun Yoo, and In So Kweon. Learning image representations by completing damaged jigsaw puzzles. In *WACV*, 2018.

Bruno Korbar, Du Tran, and Lorenzo Torresani. Cooperative learning of audio and video models from self-supervised synchronization. In *NeurIPS*, 2018.

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

Kimin Lee, Honglak Lee, Kibok Lee, and Jinwoo Shin. Training confidence-calibrated classifiers for detecting out-of-distribution samples. In *ICLR*, 2018.

Michelle A Lee, Yuke Zhu, Krishnan Srinivasan, Parth Shah, Silvio Savarese, Li Fei-Fei, Animesh Garg, and Jeannette Bohg. Making sense of vision and touch: Self-supervised learning of multimodal representations for contact-rich tasks. In *ICRA*, 2019.

Yibo Lin, Yuki Watanabe, Taiki Kimura, Tetsuaki Matsunawa, Shigeki Nojima, Meng Li, and David Z Pan. Data efficient lithography modeling with residual neural networks and transfer learning. In *Proceedings of the 2018 International Symposium on Physical Design*, pp. 82–89, 2018.

Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts. In *ICLR*, 2017.

Thomas Lucas, Corentin Tallec, Jakob Verbeek, and Yann Ollivier. Mixed batches and symmetric discriminators for gan training. In *ICML*, 2018.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *NeurIPS*, 2013.

Ishan Misra and Laurens van der Maaten. Self-supervised learning of pretext-invariant representations. In *CVPR*, 2020.

Takeru Miyato, Shin-ichi Maeda, Masanori Koyama, and Shin Ishii. Virtual adversarial training: a regularization method for supervised and semi-supervised learning. *PAMI*, 41(8):1979–1993, 2018.

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

Mehdi Noroozi and Paolo Favaro. Unsupervised learning of visual representations by solving jigsaw puzzles. In *ECCV*, 2016.

Mehdi Noroozi, Hamed Pirsiavash, and Paolo Favaro. Representation learning by learning to count. In *ICCV*, 2017.

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

Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

Andrew Owens and Alexei A Efros. Audio-visual scene analysis with self-supervised multisensory features. In *ECCV*, 2018.

Tianyu Pang, Kun Xu, and Jun Zhu. Mixup inference: Better exploiting mixup to defend adversarial attacks. *arXiv preprint arXiv:1909.11515*, 2019.

Daniel S Park, William Chan, Yu Zhang, Chung-Cheng Chiu, Barret Zoph, Ekin D Cubuk, and Quoc V Le. Specaugment: A simple data augmentation method for automatic speech recognition. *arXiv preprint arXiv:1904.08779*, 2019.

Deepak Pathak, Philipp Krahenbuhl, Jeff Donahue, Trevor Darrell, and Alexei A Efros. Context encoders: Feature learning by inpainting. In *CVPR*, 2016.

Mirco Ravanelli, Jianyuan Zhong, Santiago Pascual, Pawel Swietojanski, Joao Monteiro, Jan Trmal, and Yoshua Bengio. Multi-task self-supervised learning for robust speech recognition. In *ICASSP*, 2020.

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

Pierre Sermanet, Corey Lynch, Yevgen Chebotar, Jasmine Hsu, Eric Jang, Stefan Schaal, Sergey Levine, and Google Brain. Time-contrastive networks: Self-supervised learning from video. In *ICRA*, 2018.

Woojoo Sim, Kibok Lee, Dingdong Yang, Jaeseung Jeong, Ji-Suk Hong, Sooryong Lee, and Honglak Lee. Automatic correction of lithography hotspots with a deep generative model. In *Optical Microlithography XXXII*, volume 10961, pp. 1096105. International Society for Optics and Photonics, 2019.

Kihyuk Sohn. Improved deep metric learning with multi-class n-pair loss objective. In *NeurIPS*, 2016.

Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *CVPR*, 2015.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *CVPR*, 2016.

Sunil Thulasidasan, Gopinath Chennupati, Jeff A Bilmes, Tanmoy Bhattacharya, and Sarah Michalak. On mixup training: Improved calibration and predictive uncertainty for deep neural networks. In *NeurIPS*, 2019.

Yonglong Tian, Dilip Krishnan, and Phillip Isola. Contrastive multiview coding. In *ECCV*, 2020a.

Yonglong Tian, Chen Sun, Ben Poole, Dilip Krishnan, Cordelia Schmid, and Phillip Isola. What makes for good views for contrastive learning. *arXiv preprint arXiv:2005.10243*, 2020b.

Leonid Nisonovich Vaserstein. Markov processes over denumerable products of spaces, describing large systems of automata. *Problemy Peredachi Informatsii*, 5(3):64–72, 1969.

Vikas Verma, Alex Lamb, Christopher Beckham, Amir Najafi, Ioannis Mitliagkas, Aaron Courville, David Lopez-Paz, and Yoshua Bengio. Manifold mixup: Better representations by interpolating hidden states. In *ICML*, 2019.

Pete Warden. Speech commands: A dataset for limited-vocabulary speech recognition. *arXiv preprint arXiv:1804.03209*, 2018.

Jason W Wei and Kai Zou. Eda: Easy data augmentation techniques for boosting performance on text classification tasks. *arXiv preprint arXiv:1901.11196*, 2019.

Yue Wu, Yinpeng Chen, Lijuan Wang, Yuancheng Ye, Zicheng Liu, Yandong Guo, Zhengyou Zhang, and Yun Fu. Incremental classifier learning with generative adversarial networks. *arXiv preprint arXiv:1802.00853*, 2018a.

Zhirong Wu, Yuanjun Xiong, Stella X Yu, and Dahua Lin. Unsupervised feature learning via non-parametric instance discrimination. In *CVPR*, 2018b.

Mang Ye, Xu Zhang, Pong C Yuen, and Shih-Fu Chang. Unsupervised embedding learning via invariant and spreading instance feature. In *CVPR*, 2019.

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.

Xiaohua Zhai, Avital Oliver, Alexander Kolesnikov, and Lucas Beyer. S4l: Self-supervised semi-supervised learning. In *ICCV*, 2019.

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

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

Xiang Zhang, Junbo Zhao, and Yann LeCun. Character-level convolutional networks for text classification. In *NeurIPS*, 2015.

Zhun Zhong, Liang Zheng, Guoliang Kang, Shaozi Li, and Yi Yang. Random erasing data augmentation. In *AAAI*, 2020.

A MORE APPLICATIONS OF i-MIX

In this section, we introduce more variations of i-Mix. For conciseness, we use v_i to denote virtual labels for different methods. We make the definition of v_i for each application clear.

A.1 i-MIX FOR SIMCLR

For each anchor, SimCLR takes other anchors as negative samples such that the virtual labels must be extended. Let $x_{N+i} = \tilde{x}_i$ for conciseness, and $v_i \in \{0,1\}^{2N}$ be the virtual label indicating the positive sample of each anchor, where $v_{i,N+i} = 1$ and $v_{i,j \neq N+i} = 0$. Note that $v_{i,i} = 0$ because the anchor itself is not counted as a positive sample. Then, Eq. (4) can be represented in the form of the cross-entropy loss:

$$\ell_{\text{SimCLR}}(x_i, v_i; \mathcal{B}) = -\sum_{n=1}^{2N} v_{i,n} \log \frac{\exp\big(s(f_i, f_n)/\tau\big)}{\sum_{k=1,k\neq i}^{2N} \exp\big(s(f_i, f_k)/\tau\big)}. \tag{A.1}$$

The application of i-Mix to SimCLR is straightforward: for two data instances (x_i, v_i), (x_j, v_j) and a batch of data $\mathcal{B} = \{x_i\}_{i=1}^{2N}$, the i-Mix loss is defined as follows:[12]

$$\ell_{\text{SimCLR}}^{i\text{-Mix}}\big((x_i, v_i), (x_j, v_j); \mathcal{B}, \lambda\big) = \ell_{\text{SimCLR}}(\lambda x_i + (1-\lambda)x_j, \lambda v_i + (1-\lambda)v_j; \mathcal{B}). \tag{A.2}$$

Note that only the input data of Eq. (A.2) is mixed, such that f_i in Eq. (A.1) is an embedding vector of the mixed data while the other f_n's are the ones of clean data. Because both clean and mixed data need to be fed to the network f, i-Mix for SimCLR requires twice more memory and training time compared to SimCLR when the same batch size is used.

A.2 i-MIX FOR SUPERVISED CONTRASTIVE LEARNING

Supervised contrastive learning has recently shown to be effective for supervised representation learning and it often outperforms the standard end-to-end supervised classifier learning (Khosla et al., 2020). Suppose an one-hot label $y_i \in \{0,1\}^C$ is assigned to a data x_i, where C is the number of classes. Let $x_{N+i} = \tilde{x}_i$ and $y_{N+i} = y_i$ for conciseness. For a batch of data pairs and their labels $\mathcal{B} = \{(x_i, y_i)\}_{i=1}^{2N}$, let $v_i \in \{0,1\}^{2N}$ be the virtual label indicating the positive samples of each anchor, where $v_{i,j} = 1$ if $y_i = y_{j\neq i}$, and otherwise $v_{i,j} = 0$. Intuitively, $\sum_{j=1}^{2N} v_{i,j} = 2N_{y_i} - 1$ where N_{y_i} is the number of data with the label y_i. Then, the supervised version of the SimCLR (SupCLR) loss function is written as follows:

$$\ell_{\text{SupCLR}}(x_i, v_i; \mathcal{B}) = -\frac{1}{2N_{y_i} - 1} \sum_{n=1}^{2N} v_{i,n} \log \frac{\exp\big(s(f_i, f_n)/\tau\big)}{\sum_{k=1,k\neq i}^{2N} \exp\big(s(f_i, f_k)/\tau\big)}. \tag{A.3}$$

The application of i-Mix to SupCLR is straightforward: for two data instances (x_i, v_i), (x_j, v_j) and a batch of data $\mathcal{B} = \{x_i\}_{i=1}^{2N}$, the i-Mix loss is defined as follows:

$$\ell_{\text{SupCLR}}^{i\text{-Mix}}\big((x_i, v_i), (x_j, v_j); \mathcal{B}, \lambda\big) = \ell_{\text{SupCLR}}(\lambda x_i + (1-\lambda)x_j, \lambda v_i + (1-\lambda)v_j; \mathcal{B}). \tag{A.4}$$

A.3 i-MIX FOR N-PAIR SUPERVISED CONTRASTIVE LEARNING

Note that i-Mix in Eq. (A.4) is not as efficient as SupCLR in Eq. (A.3) due to the same reason in the case of SimCLR. To overcome this, we reformulate SupCLR in the form of the N-pair loss (Sohn, 2016). Suppose an one-hot label $y_i \in \{0,1\}^C$ is assigned to a data x_i, where C is the number of classes. For a batch of data pairs and their labels $\mathcal{B} = \{(x_i, \tilde{x}_i, y_i)\}_{i=1}^{N}$, let $v_i \in \{0,1\}^N$ be the virtual label indicating the positive samples of each anchor, where $v_{i,j} = 1$ if $y_i = y_{j\neq i}$, and otherwise $v_{i,j} = 0$. Then, the supervised version of the N-pair (Sup-N-pair) contrastive loss function is written as follows:

$$\ell_{\text{Sup-N-pair}}(x_i, v_i; \mathcal{B}) = -\frac{1}{N_{y_i}} \sum_{n=1}^{N} v_{i,n} \log \frac{\exp\big(s(f_i, \tilde{f}_n)/\tau\big)}{\sum_{k=1}^{N} \exp\big(s(f_i, \tilde{f}_k)/\tau\big)}. \tag{A.5}$$

Then, the i-Mix loss for Sup-N-pair is defined as follows:

$$\ell_{\text{Sup-N-pair}}^{i\text{-Mix}}\big((x_i, v_i), (x_j, v_j); \mathcal{B}, \lambda\big) = \ell_{\text{Sup-N-pair}}(\lambda x_i + (1-\lambda)x_j, \lambda v_i + (1-\lambda)v_j; \mathcal{B}). \tag{A.6}$$

[12] The j-th data can be excluded from the negative samples, but it does not result in a significant difference.

B PROOF OF THE LINEARITY OF LOSSES WITH RESPECT TO VIRTUAL LABELS

Cross-entropy loss. The loss used in contrastive representation learning works, which is often referred to as InfoNCE (Oord et al., 2018), can be represented in the form of the cross-entropy loss as we showed for N-pair contrastive learning, SimCLR (Chen et al., 2020a), and MoCo (He et al., 2020). Here we provide an example in the case of N-pair contrastive learning. Let $f_{ij}^{\lambda} = f(\lambda x_i + (1-\lambda)x_j)$ for conciseness.

$$
\begin{aligned}
&\ell_{\text{N-pair}}^{i\text{-Mix}}\big((x_i, v_i), (x_j, v_j); \mathcal{B}, \lambda\big) = \ell_{\text{N-pair}}(\lambda x_i + (1-\lambda)x_j, \lambda v_i + (1-\lambda)v_j; \mathcal{B}) \\
&= -\sum_{n=1}^{N} (\lambda v_{i,n} + (1-\lambda)v_{j,n}) \log \frac{\exp\big(s(f_{ij}^{\lambda}, \tilde{f}_n)/\tau\big)}{\sum_{k=1}^{N} \exp\big(s(f_{ij}^{\lambda}, \tilde{f}_k)/\tau\big)} \\
&= -\lambda \sum_{n=1}^{N} v_{i,n} \log \frac{\exp\big(s(f_{ij}^{\lambda}, \tilde{f}_n)/\tau\big)}{\sum_{k=1}^{N} \exp\big(s(f_{ij}^{\lambda}, \tilde{f}_k)/\tau\big)} - (1-\lambda) \sum_{n=1}^{N} v_{j,n} \log \frac{\exp\big(s(f_{ij}^{\lambda}, \tilde{f}_n)/\tau\big)}{\sum_{k=1}^{N} \exp\big(s(f_{ij}^{\lambda}, \tilde{f}_k)/\tau\big)} \\
&= \lambda \ell_{\text{N-pair}}(\lambda x_i + (1-\lambda)x_j, v_i; \mathcal{B}) + (1-\lambda)\ell_{\text{N-pair}}(\lambda x_i + (1-\lambda)x_j, v_j; \mathcal{B}). \qquad \text{(B.1)}
\end{aligned}
$$

L2 loss between L2-normalized feature vectors. The BYOL (Grill et al., 2020) loss is in this type. Let $\tilde{F} = [\tilde{f}_1/\|\tilde{f}_1\|, \ldots, \tilde{f}_N/\|\tilde{f}_N\|] \in \mathbb{R}^{D\times N}$ such that $\tilde{f}_i/\|\tilde{f}_i\| = \tilde{F}v_i$, and $\bar{g} = g(f(\lambda x_i + (1-\lambda)x_j))/\|g(f(\lambda x_i + (1-\lambda)x_j))\|$ for conciseness.

$$
\begin{aligned}
&\ell_{\text{BYOL}}^{i\text{-Mix}}\big((x_i, v_i), (x_j, v_j); \mathcal{B}, \lambda\big) = \ell_{\text{BYOL}}(\lambda x_i + (1-\lambda)x_j, \lambda v_i + (1-\lambda)v_j) \\
&= \left\|\bar{g} - \tilde{F}(\lambda v_i + (1-\lambda)v_j)\right\|^2 = \left\|\bar{g} - \left(\lambda \tilde{F}v_i + (1-\lambda)\tilde{F}v_j\right)\right\|^2 \\
&= 1 - 2\cdot \bar{g}^{\top}\left(\lambda \tilde{F}v_i + (1-\lambda)\tilde{F}v_j\right) + \left\|\lambda \tilde{F}v_i + (1-\lambda)\tilde{F}v_j\right\|^2 \\
&= 2 - 2\cdot \bar{g}^{\top}\left(\lambda \tilde{F}v_i + (1-\lambda)\tilde{F}v_j\right) + \text{const} \\
&= \lambda\|\bar{g} - \tilde{F}v_i\|^2 + (1-\lambda)\|\bar{g} - \tilde{F}v_j\|^2 + \text{const} \\
&= \lambda \ell_{\text{BYOL}}(\lambda x_i + (1-\lambda)x_j, v_i; \mathcal{B}) + (1-\lambda)\ell_{\text{BYOL}}(\lambda x_i + (1-\lambda)x_j, v_j; \mathcal{B}) + \text{const}. \qquad \text{(B.2)}
\end{aligned}
$$

Because $\tilde{F}$ is not backpropagated, it can be considered as a constant.

C MORE EXPERIMENTS

C.1 SETUP

In this section, we describe details of the experimental settings. Note that the learning rate is scaled by the batch size (Goyal et al., 2017): $\text{ScaledLearningRate} = \text{LearningRate} \times \text{BatchSize}/256$.

Image. The experiments on CIFAR-10 and 100 (Krizhevsky & Hinton, 2009) and ImageNet (Deng et al., 2009) are conducted in two stages: following Chen et al. (2020a), the convolutional neural network (CNN) part of ResNet-50 (He et al., 2016)[13] followed by the two-layer multilayer perceptron (MLP) projection head (output dimensions are 2048 and 128, respectively) is trained on the unlabeled pretext dataset with a batch size of 256 (i.e., 512 augmented data) with the stochastic gradient descent (SGD) optimizer with a momentum of 0.9 over up to 4000 epochs. BYOL has an additional prediction head (output dimensions are the same with the projection head), which follows the projection head, only for the model updated by gradient. 10 epochs of warmup with a linear schedule to an initial learning rate of 0.125, followed by the cosine learning rate schedule (Loshchilov & Hutter, 2017) is used. We use the weight decay of 0.0001 at the first stage.

Then, the head of the CNN is replaced with a linear classifier, and only the linear classifier is trained with the labeled downstream dataset. For N-pair contrastive learning and SimCLR, we use a batch size of 512 with the SGD optimizer with a momentum of 0.9 and an initial learning rate of 1 over 100

[13] For small resolution data from CIFAR and Speech Commands, we replaced the kernal, stride, and padding size from (7,2,3) to (3,1,1) in the first convolutional layer, and removed the first max pooling layer, following Chen et al. (2020a).

epochs, where the learning rate is decayed by 0.2 after 80, 90, 95 epochs. For MoCo and BYOL, we use the same learning schedule but an initial learning rate chosen among $\{10, 30, 50, 70\}$. No weight decay is used at the second stage. The quality of representation is evaluated by the top-1 accuracy on the downstream task. We sample a single mixing coefficient $\lambda \sim \text{Beta}(1, 1)$ for each training batch. The temperature is set to be $\tau = 0.5$ for N-pair contrastive learning and SimCLR, and 0.2 for MoCo. Note that the optimal distribution of λ and the optimal value of τ varies over different architectures, methods, and datasets, but the choices above result in a reasonably good performance. The memory bank size is 4096 for MoCo, and the momentum for the exponential moving average (EMA) update is 0.999 for MoCo and BYOL. We do not symmetrize the BYOL loss, as it does not change the performance while increasing computational complexity.

For data augmentation, we follow Chen et al. (2020a): We apply a set of data augmentations randomly in sequence including resized cropping (Szegedy et al., 2015), horizontal flipping with a probability of 0.5, color jittering,[14] and gray scaling with a probability of 0.2. A Gaussian blurring with $\sigma \in [0.1, 2]$ and kernel size of 10% of the image height/width is applied for ImageNet. For evaluation on downstream tasks, we apply padded cropping with the pad size of 4 and horizontal flipping for CIFAR-10 and 100, and resized cropping and horizontal flipping for ImageNet.

Speech. In the experiments on Speech Commands (Warden, 2018), the network is the same with the image domain experiments, except that the number of input channels is one instead of three. We use the temperature is set to be $\tau = 0.5$. 10% of silence data (all zero) are added when training. At the first stage, the model is trained with the SGD optimizer with a momentum of 0.9 and an initial learning rate of 0.0001 over 500 epochs, where the learning rate decays by 0.1 after 300 and 400 epochs and the weight decay of 0.0001. The other settings are the same with the experiments in the image domain. At the second stage, the MLP head is replaced with a linear classifier. For N-pair contrastive learning, we use a batch size of 512 with the SGD optimizer with a momentum of 0.9 and an initial learning rate of 1 over 100 epochs, where the learning rate is decayed by 0.2 after 80, 90, 95 epochs. For MoCo and BYOL, we use the same learning schedule but an initial learning rate of 10. No weight decay is used at the second stage.

For data augmentation,[15] we apply a set of data augmentations randomly in sequence including changing amplitude, speed, and pitch in time domain, stretching, time shifting, and adding background noise in frequency domain. Each data augmentation is applied with a probability of 0.5. Augmented data are then transformed to the mel spectogram in the size of 32×32.

Tabular. In the experiments on CovType and Higgs (Asuncion & Newman, 2007), we take a five-layer MLP with batch normalization as a backbone network. The output dimensions of layers are (2048-2048-4096-4096-8192), where all layers have batch normalization followed by ReLU except for the last layer. The last layer activation is maxout (Goodfellow et al., 2013) with 4 sets, such that the output dimension is 2048. On top of this five-layer MLP, we attach two-layer MLP (2048-128) as a projection head. We sample a single mixing coefficient $\lambda \sim \text{Beta}(2, 2)$ for each training batch. We use the temperature $\tau = 0.1$. For N-pair contrastive learning, at the first stage, the model is trained with a batch size of 512 with the Adam optimizer with beta values of 0.9 and 0.999, an initial learning rate of 5×10^{-5} and 2.5×10^{-5} over 1000 epochs for CovType and Higgs, respectively, and the weight decay of 0.0001. For MoCo and BYOL with CovType, we use the same learning schedule but an initial learning rate of 0.0002 and 5×10^{-5}, respectively. The other settings are the same with the experiments in the image domain. At the second stage, the MLP head is replaced with a linear classifier. The classifier is computed by linear regression from the feature matrix obtained without data augmentation to the label matrix using the pseudoinverse. Since the prior knowledge on tabular data is very limited, only the masking noise with a probability of 0.2 is considered as a data augmentation.

C.2 Variations of i-Mix

In this section, we additionally compare the MixUp (Zhang et al., 2018) and CutMix (Yun et al., 2019) variation of i-Mix on N-pair contrastive learning and SimCLR. To distinguish them, we call them i-MixUp and i-CutMix, respectively. To be fair with the memory usage in the pretext task stage,

[14] Specifically, brightness, contrast, and saturation are scaled by a factor uniformly sampled from $[0.6, 1.4]$ at random, and hue is rotated in the HSV space by a factor uniformly sampled from $[-0.1, 0.1]$ at random.

[15] https://github.com/tugstugi/pytorch-speech-commands

Pretext	Downstream	N-pair			SimCLR		
		Vanilla	i-MixUp	i-CutMix	Vanilla	i-MixUp	i-CutMix
CIFAR-10	CIFAR-10	92.4 $\pm$ 0.1	**94.8** $\pm$ 0.2	94.7 $\pm$ 0.1	92.5 $\pm$ 0.1	**94.8** $\pm$ 0.2	**94.8** $\pm$ 0.2
	CIFAR-100	60.2 $\pm$ 0.3	**63.3** $\pm$ 0.2	61.5 $\pm$ 0.2	60.0 $\pm$ 0.2	**61.4** $\pm$ 1.0	57.1 $\pm$ 0.4
CIFAR-100	CIFAR-10	84.4 $\pm$ 0.2	**86.2** $\pm$ 0.2	85.1 $\pm$ 0.2	84.4 $\pm$ 0.2	**85.2** $\pm$ 0.3	83.7 $\pm$ 0.6
	CIFAR-100	68.7 $\pm$ 0.2	**72.3** $\pm$ 0.2	**72.3** $\pm$ 0.4	68.7 $\pm$ 0.2	**72.3** $\pm$ 0.2	71.7 $\pm$ 0.2

Table C.1: Comparison of N-pair contrastive learning and SimCLR with i-MixUp and i-CutMix on them with ResNet-50 on CIFAR-10 and 100. We run all experiments for 1000 epochs. i-MixUp improves the accuracy on the downstream task regardless of the data distribution shift between the pretext and downstream tasks. i-CutMix shows a comparable performance with i-MixUp when the pretext and downstream datasets are the same, but it does not when the data distribution shift occurs.

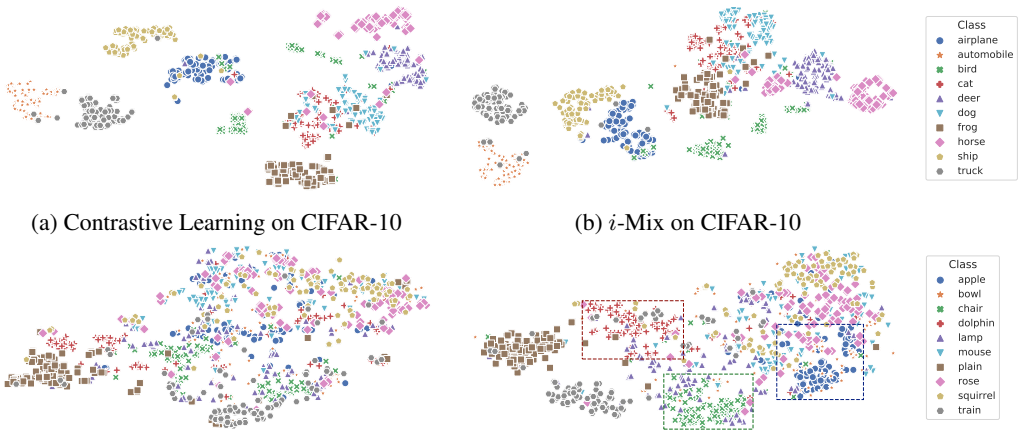
Pretext	Downstream	Self-Supervised Pretext			Supervised Pretext		
		SimCLR	N-pair	+ i-Mix	SimCLR	N-pair	+ i-Mix
CIFAR-10	CIFAR-10	92.5 $\pm$ 0.1	92.4 $\pm$ 0.1	**94.8** $\pm$ 0.2	95.6 $\pm$ 0.3	95.7 $\pm$ 0.1	**97.0** $\pm$ 0.1
	CIFAR-100	60.0 $\pm$ 0.2	60.2 $\pm$ 0.3	**63.3** $\pm$ 0.2	58.6 $\pm$ 0.2	**58.9** $\pm$ 0.5	57.8 $\pm$ 0.6
CIFAR-100	CIFAR-10	84.4 $\pm$ 0.2	84.4 $\pm$ 0.2	**86.2** $\pm$ 0.2	86.5 $\pm$ 0.4	86.7 $\pm$ 0.2	**88.7** $\pm$ 0.2
	CIFAR-100	68.7 $\pm$ 0.2	68.7 $\pm$ 0.2	**72.3** $\pm$ 0.2	74.3 $\pm$ 0.2	74.6 $\pm$ 0.3	**78.4** $\pm$ 0.2

Table C.2: Comparison of the N-pair self-supervised and supervised contrastive learning methods and i-Mix on them with ResNet-50 on CIFAR-10 and 100. We also provide the performance of formulations proposed in prior works: SimCLR (Chen et al., 2020a) and its supervised version (Khosla et al., 2020). We run all experiments for 1000 epochs. i-Mix improves the accuracy on the downstream task regardless of the data distribution shift between the pretext and downstream tasks, except the case that the pretest task has smaller number of classes than that of the downstream task. The quality of representation depends on the pretext task in terms of the performance of transfer learning: self-supervised learning is better on CIFAR-10, while supervised learning is better on CIFAR-100.

we reduce the batch size of i-MixUp and i-CutMix by half (256 to 128) for SimCLR. Following the learning rate adjustment strategy in Goyal et al. (2017), we also decrease the learning rate by half (0.125 to 0.0625) when the batch size is reduced. We note that i-MixUp and i-CutMix on SimCLR take approximately 2.5 times more training time to achieve the same number of training epochs. The results are provided in Table C.1. We first verify that the N-pair formulation results in no worse performance than that of SimCLR. This justifies to conduct experiments using the N-pair formulation instead of that of SimCLR, which is simpler and more efficient, especially when applying i-Mix, while not losing the performance. When pretext and downstream tasks share the training dataset, i-CutMix often outperforms i-MixUp, though the margin is small. However, i-CutMix shows a worse performance in transfer learning.

Table C.2 compares the performance of SimCLR, N-pair contrastive learning, and i-Mix on N-pair contrastive learning when the pretext task is self-supervised and supervised contrastive learning. We confirm that the N-pair formulation results in no worse performance than that of SimCLR in supervised contrastive learning as well. i-Mix improves the performance of supervised contrastive learning from 95.7% to 97.0% on CIFAR-10, similarly to improvement achieved by MixUp for supervised learning where it improves the performance of supervised classifier learning from 95.5% to 96.6%. On the other hand, when the pretext dataset is CIFAR-100, the performance of supervised contrastive learning is not better than that of supervised learning: MixUp improves the performance of supervised classifier learning from 78.9% to 82.2%, and i-Mix improves the performance of supervised contrastive learning from 74.6% to 78.4%.

While supervised i-Mix improves the classification accuracy on CIFAR-10 when trained on CIFAR-10, the representation does not transfer well to CIFAR-100, possibly due to overfitting to 10 class classification. When pretext dataset is CIFAR-100, supervised contrastive learning shows a better performance than self-supervised contrastive learning regardless of the distribution shift, as it learns sufficiently general representation for linear classifier to work well on CIFAR-10 as well.



(a) Contrastive Learning on CIFAR-10

(b) i-Mix on CIFAR-10

(c) Contrastive Learning on CIFAR-100

(d) i-Mix on CIFAR-100

Figure C.1: t-SNE visualization of embeddings trained by contrastive learning and i-Mix with ResNet-50 on CIFAR-10. (a,b): Classes are well-clustered in both cases when applied to CIFAR-10. (c,d): When models are transferred to CIFAR-100, classes are more clustered for i-Mix than contrastive learning, as highlighted in dashed boxes. We show 10 classes for a better visualization.

Pretext	Downstream	FED ($\times 10^{-4}$) ($\downarrow$)		Training Acc (%) ($\uparrow$)		Test Acc (%) ($\uparrow$)	
		N-pair	+ i-Mix	N-pair	+ i-Mix	N-pair	+ i-Mix
CIFAR-10	CIFAR-10	30.0	**16.7**	**96.1**	**96.1**	92.4	**94.8**
	CIFAR-100	13.8	**7.9**	**70.7**	69.5	60.2	**63.3**
CIFAR-100	CIFAR-10	15.2	**9.7**	88.1	**88.8**	84.4	**86.2**
	CIFAR-100	30.4	**13.3**	**85.6**	79.0	68.7	**72.3**

Table C.3: Comparison of N-pair contrastive learning and i-Mix with ResNet-50 on CIFAR-10 and 100 in terms of the Fréchet embedding distance (FED) between training and test data distribution on the embedding space, and training and test accuracy. $\uparrow$ ($\downarrow$) indicates that the higher (lower) number is the better. i-Mix improves contrastive learning in all metrics, which shows that i-Mix is an effective regularization method for the pretext task, such that the learned representation is more generalized.

C.3 Qualitative Embedding Analysis

Figure C.1 visualizes embedding spaces learned by N-pair contrastive learning and i-Mix on CIFAR-10 and 100. When the downstream dataset is the same with the pretext task, both contrastive learning and i-Mix cluster classes well, as shown in Figure C.1(a) and C.1(b). However, when the downstream task is transferred to CIFAR-100, i-Mix in Figure C.1(d) clusters classes better than contrastive learning in Figure C.1(c). Specifically, clusters of "apple," "chair," and "dolphin," can be found in Figure C.1(d) while they spread out in Figure C.1(c). Also, "rose" and "squirrel" are more separated in Figure C.1(d) than C.1(c). This shows that the representation learned with i-Mix is more generalizable than vanilla contrastive learning.

C.4 Quantitative Embedding Analysis

To estimate the quality of representation by the similarity between training and test data distribution, we measure the Fréchet embedding distance (FED): similarly to the Fréchet inception distance (FID) introduced in Heusel et al. (2017), FED is the Fréchet distance (Fréchet, 1957; Vaserstein, 1969) between the set of training and test embedding vectors under the Gaussian distribution assumption. For conciseness, let $\bar{f}_i = f(x_i)/\|f(x_i)\|$ be an ℓ_2 normalized embedding vector; we normalize embedding vectors as we do when we measure the cosine similarity. Then, with the estimated mean $m = \frac{1}{N}\sum_{i=1}^{N} \bar{f}_i$ and the estimated covariance $S = \frac{1}{N}\sum_{i=1}^{N}(\bar{f}_i - m)(\bar{f}_i - m)^\top$, the FED can be defined as

$$d^2\big((m^{\mathtt{tr}}, S^{\mathtt{tr}}), (m^{\mathtt{te}}, S^{\mathtt{te}})\big) = \|m^{\mathtt{tr}} - m^{\mathtt{te}}\|^2 + \mathrm{Tr}\big(S^{\mathtt{tr}} + S^{\mathtt{te}} - 2(S^{\mathtt{tr}}S^{\mathtt{te}})^{\frac{1}{2}}\big). \tag{C.1}$$

As shown in Table C.3, i-Mix improves FED over contrastive learning, regardless of the distribution shift. Note that the distance is large when the training dataset of the downstream task is the same with that of the pretext task. This is because the model is overfit to the training dataset, such that the distance from the test dataset, which is unseen during training, has to be large.

On the other hand, Table C.3 shows that i-Mix reduces the gap between the training and test accuracy. This implies that i-Mix is an effective regularization method for pretext tasks, such that the learned representation is more generalizable in downstream tasks.