Infinite Class Mixup

Thomas Mensink* [1]
mensink@google.com
Pascal Mettes* [2]
p.s.m.mettes@uva.nl

[1] Google Research
[2] University of Amsterdam

Abstract

Mixup is a widely adopted strategy for training deep networks, where additional samples are augmented by interpolating inputs and labels of training pairs. Mixup has shown to improve classification performance, network calibration, and out-of-distribution generalisation. While effective, a cornerstone of Mixup, namely that networks learn linear behaviour patterns between classes, is only indirectly enforced since the output interpolation is performed at the probability level. This paper seeks to address this limitation by mixing the classifiers directly instead of mixing the labels for each mixed pair. We propose to define the target of each augmented sample as a uniquely new classifier, whose parameters are a linear interpolation of the classifier vectors of the input pair. The space of all possible classifiers is continuous and spans all interpolations between classifier pairs. To make optimisation tractable, we propose a dual-contrastive Infinite Class Mixup loss, where we contrast the classifier of a mixed pair to both the classifiers and the predicted outputs of other mixed pairs in a batch. Infinite Class Mixup is generic in nature and applies to many variants of Mixup. Empirically, we show that it outperforms standard Mixup and variants such as RegMixup and Remix on balanced, long-tailed, and data-constrained benchmarks, highlighting its broad applicability. The code is available online at: https://github.com/psmmettes/icm.

1 Introduction

There is a strong dependence between generalisation of deep networks and their access to rich and diverse samples for training [5], since deep neural networks tend to overfit to training sampled, or even memorise them [16]. Mixup forms a canonical approach to counteract this inclination [66]. With Mixup, new samples are created by linearly interpolating input pairs and their corresponding ground truth outputs. By augmenting training pairs, a network is given insight into the linear transitions between classes, which helps to alleviate over-fitting.

Over the years, Mixup has shown to consistently improve down-stream performance for images [1, 64], videos [37], point clouds [9], graphs [64], and more. Several works have observed that the improved performance of Mixup can be attributed to better network calibration [48] and out-of-manifold regularisation [17]. Due to its simplicity and strong empirical results, a wide range of Mixup variants have also been proposed *e.g.*, to improve long-tailed recognition [11], semi-supervised learning [3], and fairness [12]. At the core of training with Mixup is the following intuition:

*Equal advising, writing, and coding. Order decided by coin flip.

... Mixup extends the training distribution by incorporating the prior knowledge that linear interpolations of feature vectors should lead to linear interpolations of the associated targets.
Zhang et al. [66]

An assumption shared by all Mixup variants is that *interpolation of the targets* should be at the probability level, which results in using cross-entropy losses with *interpolated one-hot ground truth targets*. For a Mixup interpolation between an image of a *dog* and an image of a *cat* with interpolation ratio λ, the loss should enforce probability λ for the *dog* class and 1-λ for the *cat* class. Linear interpolation at the probability level, however, does not strictly imply linear classifier interpolation. This paper argues instead that *linear interpolations of input images should lead to linear interpolations of classifiers*. From this view, for the interpolated image from the *dog* and *cat*, the target is a new (and unique) classifier, with its parameters given as the linear interpolation between the current *dog* and *cat* classifier weights. This allows us to directly enforce linear interpolation in the classifier space.

Since the space of convex interpolations between all class pairs is continuous, there are an infinite number of Mixup classifiers to integrate over in a cross-entropy formulation, hence we name our proposed method *Infinite Class Mixup*. To make the optimisation tractable, we propose a dual-contrastive loss. For each mixed pair in a batch, we obtain a pair-specific classifier. We seek to optimise each pair towards their specific classifier and away from all other classifiers in the batch, resulting in an identity matrix with mixed pairs and their classifiers along the axes. We optimise with cross-entropy simultaneously over both axes, which have complementary gradient flows, and simply sum their losses. These contrastive losses are not instantly applicable in standard Mixup, as it does not provide a direct setup to obtain positive and negative pairs. In Infinite Class Mixup, however, these pairs arise naturally because each example corresponds to a uniquely defined classifier, allowing contrastive losses between all unique examples and all classifiers in a batch.

We show how Infinite Class Mixup can be integrated into different Mixup variants. Empirically, we find that our Infinite Class formulation improves classification in standard, data-constrained, and imbalanced settings, outperforming both the conventional formulation and recent variants such as RegMixup [43] and Remix [11]. Infinite Class Mixup does not require additional parameters and has similar computational cost compared to standard Mixup.

2 Related work

Mixup. Mixup as proposed by Zhang et al. [66] is outlined for images by linearly interpolating image pairs for every pixel. Rather than interpolating on a global level, several works have proposed variants that interpolate images at a local level. For example, CutMix mixed images by masking part of one image onto a region of the other image, with the mask size equal to the interpolation ratio [64]. Similarly, GridMix [1] and RICAP [47] focus on a few subregions by dividing images into grids and randomly assigning image patches to the grids. PuzzleMix [24], co-Mixup [25], and Attentive CutMix [53] additionally include saliency or attention information to improve foreground selection in the image mixing, while StyleMix mixed both content and style for more visually coherent image mixes [20]. AugMix performs Mixup between images and transformed versions [19] and AlignMixup geometrically aligns images in feature space before mixing [51]. TokenMix mixes images at the token level for effective use in transformer models [34]. In Mixup without hesitation, the Mixup strategy is periodically turned off and on to speed up convergence and increase robustness to the interpolation hyperparameter [63], while RegMixup combines the cross-entropy loss

of the individual samples with the loss for mixed samples [43]. Manifold Mixup proposes to perform the interpolation on a latent manifold within the network, rather than the at the input-level [52], which has shown to be effective for few-shot learning as well [58]. Other variants include TransMix [7], AutoMix [36], and RecursiveMix [61].

Mixup and its variants focus on interpolating outputs at the probability level. This paper complements current Mixup literature by performing the output interpolation at the classifier-level instead of at the probability-level for improved down stream performance. For the inputs, we follow the original Mixup, so our proposed output interpolation could be combined with variants on input interpolation, such as Manifold Mixup [38, 52].

Mixup in contrastive learning. Mixup variants have been proposed for contrastive learning. For example, Lee *et al*. [31] and Ren *et al*. [45] perform Mixup in contrastive self-supervised learning with virtual labels, where one of the two views of an example is replaced with a Mixup variant of that view. In this paper, we use mixup to create an image and a classifier, and use these in a contrastive learning framework. Koshla *et al*. [23] extend contrastive learning, to a supervised version, where multiple images from the same class can be used in a batch. We, on the other hand, use mixup to create unique pairs, so that each batch has only unique classes by construction. The concurrent Two-Way Loss [28] also performs contrastive learning on both axes of a sample-class matrix for multi-label classification. In contrast, we generate interpolated classes as per-sample targets and propose a dual-axis objective to improve general classification.

Adapting Mixup to other tasks. Mixup outlines a general formulation to interpolate image samples. A wide range of works have therefore proposed task-specific extensions to Mixup. For long-tailed recognition problems, extensions include Remix [11], balanced Mixup [14], label occurrence-balanced Mixup [68], and dynamic Mixup [15], all of which bias the mixing coefficients toward minority classes. For out-of-distribution detection and robustness, extensions include adversarial vertex Mixup [32] and Mixup during inference [41]. For semi-supervised data, Mixmatch provides labelled and unlabelled mixed images [3], while Mix-and-Unmix in feature space improves semi-supervised object detection [26].

Mixup has furthermore been extended to regression [62], facial expression recognition [44], fairness [12], self-knowledge distillation [60], retrieval [42], domain adaptation [57, 59, 70], COVID-19 detection in images [21], zero-shot learning [58], and more. Our proposed Infinite Class Mixup is also viable to task-specific Mixup formulations, as highlighted by comparisons and experiments on long-tailed and data-constrained recognition tasks.

Mixup beyond images and videos. Mixup has also shown to be an effective learning strategy beyond interpolating pixels. PointMixup performs Mixup for point clouds by performing the interpolation between two training point clouds through optimal transport [9] and PointCutMix generalises CutMix to point clouds [67]. Other point cloud Mixup methods include Rigid SubSet Mixup [30] and Point MixSwap [50]. Mixup has also been investigated for LiDAR [56], graphs [54], speaker verification [71], vision-language navigation [33], single-view 3D reconstruction [10], and language processing [29, 39, 46, 55, 69]. We focus on Mixup for images, but our approach is generic and can be applied to many Mixup variants.

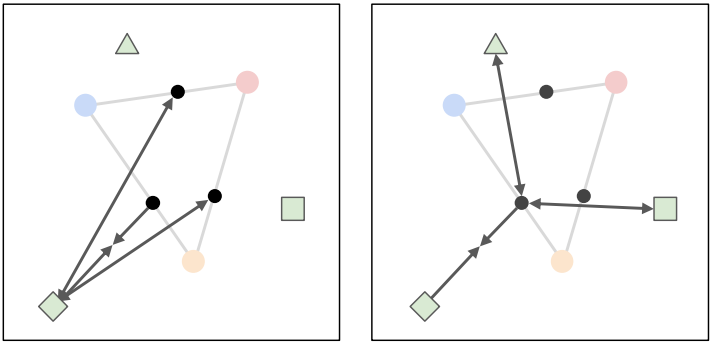

Figure 1: **Dual-contrastive learning in Infinite Class Mixup.** The blue, red, and yellow circles denote the original classifiers. The green shapes denote mixed samples, the black dots denote interpolated classifiers. Our contrastive loss is defined between a mixed example and all classes (*left*) and between a class and all mixed examples in the same batch (*right*).

3 Mixup with Infinite Classifiers

Mixup uses interpolation between training examples to create an infinite amount of training data, which improves test generalisation, typically defined as follows:

$$\tilde{\boldsymbol{x}} = \lambda\ \boldsymbol{x}_a + (1-\lambda)\ \boldsymbol{x}_b, \qquad \tilde{\boldsymbol{y}} = \lambda\ \boldsymbol{y}_a + (1-\lambda)\ \boldsymbol{y}_b, \tag{1}$$

where the mixup image $\tilde{\boldsymbol{x}}$ is the result of interpolating input image $\boldsymbol{x}_a$ of class A and $\boldsymbol{x}_b$ of class B, with interpolation ratio λ ($0 \leq \lambda \leq 1$). The mixup target $\tilde{\boldsymbol{y}}$ is the interpolation of the one-hot encoded ground truth vectors $\boldsymbol{y}_a$ and $\boldsymbol{y}_b$. The interpolation ratio λ is drawn from a parameterised Beta(α, α) distribution.

Interpolated classifiers. The idea behind Infinite Class Mixup is that every mixup image $\tilde{\boldsymbol{x}}$ corresponds to its own class $C_{\tilde{\boldsymbol{y}}}$, defined as an interpolated class from the C original classes using the mixup weights $\tilde{\boldsymbol{y}}$. The class $C_{\tilde{\boldsymbol{y}}}$ is fully specified by the C original classes and the mixing weights $\tilde{\boldsymbol{y}}$, since the mixing weights are continuous there are infinite many possible interpolated classes.

More formally, we construct the classifier weights $\boldsymbol{w}_{\tilde{\boldsymbol{y}}}$, mixing the original classifier weights $\boldsymbol{w}_c$ proportionally to the Mixup weights:

$$\boldsymbol{w}_{\tilde{\boldsymbol{y}}} = \sum_c \tilde{\boldsymbol{y}}_c W_c = W\tilde{\boldsymbol{y}}, \tag{2}$$

where $W \in \mathcal{R}^{D \times C}$ is a matrix with the classifier weights of the final layer of a deep neural network (C number of original classes each having a D dimensional weight vector), and $\tilde{\boldsymbol{y}} \in \mathcal{R}^C$ is the vector with per-class mixing weights, *i.e.* the contribution of each class to the Mixup example $\tilde{\boldsymbol{x}}$. Note that this formulation is rather generic, in the case that $\tilde{\boldsymbol{y}}$ corresponds to a one-hot encoding of class c, the classifier weights $\boldsymbol{w}_{\tilde{\boldsymbol{y}}} = \boldsymbol{w}_c$.

Interpolated class probability. We define the class probability of interpolated class $C_{\tilde{\boldsymbol{y}}}$ similar to many softmax style classifiers:

$$p(C_{\tilde{\boldsymbol{y}}}|\tilde{\boldsymbol{x}}) \propto \exp(\tilde{\boldsymbol{r}}^\top W\tilde{\boldsymbol{y}}), \qquad \text{using } \tilde{\boldsymbol{r}} = f_\theta(\tilde{\boldsymbol{x}}), \tag{3}$$

where $\tilde{\boldsymbol{r}}$ is the representation of image $\tilde{\boldsymbol{x}}$ extracted from the penultimate layer of a deep convolutional network $f_\theta(\cdot)$. To learn the values for the parameters $\{\theta, W\}$, we maximise the log-likelihood of the interpolated class prediction, as commonly used:

$$\mathcal{L} = \sum_i \log p(C_{\tilde{\mathbf{y}}_i} | \tilde{\boldsymbol{x}}_i). \tag{4}$$

We use the interpolated classes $C_{\tilde{\mathbf{y}}}$ only during training, at test time we evaluate the original classes C using the classifier weights W and the network f_θ.

In standard cross-entropy optimisation, the class probability of Equation 3 is obtained through a normaliser Z over all ground truth classes, given by $Z = \sum_{c'} \exp(\tilde{\boldsymbol{r}}^\top \boldsymbol{w}_{c'})$. In our approach, the normaliser is given now over all possible interpolated classifiers, which forms a continuous space. Thus, with infinite many classes this sum is intractable and hence we take a contrastive learning view where the normaliser Z depends on the other examples in the batch. Below we introduce two contrastive variants.

3.1 Contrastive learning of mixed samples

Here, we offer a contrastive view on the cross entropy loss of Equation 4. In many contrastive learning settings, positive pairs are formed by an image and its augmented version, and per batch negative pairs are sampled. In Infinite Class Mixup, each Mixup image $\tilde{\boldsymbol{x}}_i$ belongs to a unique class $C_{\tilde{\mathbf{y}}_i}$, this can also be seen as a positive pair which should be *contracted*. Interestingly enough, the sampling of negative pairs, which should be *detracted*, can be done along two axis: across the different Mixup classes in the batch, or across the different images in the batch, see Figure 1. Below we discuss both axes sequentially.

Contrasting classifiers. First we consider the setting where each image is compared to all classifiers, akin to a standard softmax classification network. In this setting the positive pair $(\tilde{\boldsymbol{x}}_i, C_{\tilde{\mathbf{y}}_i})$ is paired with negative pairs $(\tilde{\boldsymbol{x}}_i, C_{\tilde{\mathbf{y}}_j})$, using the same image, but different interpolated classes from the same batch. Then, the normaliser Z in Eq. 3 is defined as follows:

$$Z_{\mathbf{cc}} = \sum_j \exp(\tilde{\boldsymbol{r}}_i \, W^\top \tilde{\mathbf{y}}_j), \tag{5}$$

which results in the following gradient for the classifier weights of class c:

$$\nabla_{W_c} \mathcal{L}_{\mathbf{cc}} = \sum_i \tilde{\boldsymbol{r}}_i \left(\tilde{\mathbf{y}}_{ic} - \sum_j p_{\mathbf{cc}}(C_{\tilde{\mathbf{y}}_j} | \tilde{\boldsymbol{x}}_i) \tilde{\mathbf{y}}_{jc} \right), \tag{6}$$

where we use **cc** to denote that we use contrastive classes, and $p_{\mathbf{cc}}$ denotes the use of normaliser $Z_{\mathbf{cc}}$ for the class probability of Eq. 3. We refer to the supplementary material for the details on the derivation.

Contrasting interpolated images. Second we consider the setting where each classifier is compared to all interpolated images. Instead of contrasting $(\tilde{\boldsymbol{x}}_i, C_{\tilde{\mathbf{y}}_i})$ against pairs with the same image, we use pairs $(\tilde{\boldsymbol{x}}_j, C_{\tilde{\mathbf{y}}_i})$ with the same classifier but different images. This is equivalent in changing the normaliser to:

$$Z_{\mathbf{ci}} = \sum_j \exp(\tilde{\boldsymbol{r}}_j \, W^\top \tilde{\mathbf{y}}_i). \tag{7}$$

The gradient with respect to the weights W_c of a class c are then given by:

$$\nabla_{W_c}\mathcal{L}_{\mathbf{ci}} = \sum_i \tilde{\mathbf{y}}_{ic}\Big(\tilde{\mathbf{r}}_i - \sum_j p_{\mathbf{ci}}(C_{\tilde{\mathbf{y}}_i}|\tilde{\mathbf{x}}_j)\tilde{\mathbf{r}}_j\Big), \tag{8}$$

where **ci** denotes the use of contrastive interpolated images. This derivation is also given in the supplementary material. Learning by contrasting other images is not directly applicable in standard Mixup variants, but is enabled with our Infinite Class perspective.

Joint batch-level optimisation. The two contrastive views visualised in Figure 1 allow for complementary optimisation. Yet the space of all possible pairs to contrast against remains infinite. As is common in contrastive learning [8, 23], we maximise the likelihood of Eq. 3 by contrasting against all other mixed samples in the same batch, resulting in:

$$\mathcal{L} = \sum_i^{|B|} \log(p_{\mathbf{cc}}(C_{\tilde{\mathbf{y}}_i}|\tilde{\mathbf{x}}_i)) + \log(p_{\mathbf{ci}}(C_{\tilde{\mathbf{y}}_i}|\tilde{\mathbf{x}}_i)), \tag{9}$$

where $|B|$ denotes the batch size. This is implemented efficiently, by running a standard classification network up to the score (or logit) matrix $S \in \mathcal{R}^{B\times C}$ with the scores to the *original dataset classes*. Then we compute $\tilde{S} = S\tilde{Y}^\top$, where $\tilde{Y}$ is the $B \times C$ matrix of the stacked Mixup class contributions $\tilde{\mathbf{y}}$, resulting in a $B \times B$ score matrix to the Mixup classes. The contrastive loss (Eq. 9) is then the cross entropy loss over both axes of the score matrix $\tilde{S}$, hence we denote class-axis as $p_{\mathbf{cc}}$ and pair-axis as $p_{\mathbf{ci}}$. In code, we call the cross entropy loss function with $\tilde{S}$ and with its transpose $\tilde{S}^\top$.

Relation to Mixup. In Mixup, the gradient with respect to the classifier weights W_c is:

$$\nabla_{W_c}\mathcal{L} = \sum_i \tilde{\mathbf{r}}_i\Big(\tilde{\mathbf{y}}_{ic} - p(c|\tilde{\mathbf{x}}_i)\Big). \tag{10}$$

The gradients of Mixup and Contrasting classifiers (Eq. 6) are similar, *i.e.* both subtract the expected (predicted) class contribution $\mathbb{E}[\mathbf{y}_c]$ from the ground truth class contribution, resulting in: $\tilde{\mathbf{r}}_i(\tilde{\mathbf{y}}_{ic} - \mathbb{E}[\mathbf{y}_c])$, albeit both estimate the expectation differently. Figure 2 illustrates this difference: in standard Mixup, an example pulls towards the mixed classifiers, and retracts from the other classifiers, but the strength of each depends only on the post-softmax probabilities, *i.e.* the curly lines, which are normalized scores instead of direct indiactors of classifier strength. The contrasting images loss (Eq. 8), however, rather looks at the expected class representation $\mathbb{E}[\tilde{\mathbf{r}}]$ for class $C_{\tilde{\mathbf{y}}_i}$, resulting in $\tilde{\mathbf{y}}_{ic}(\tilde{\mathbf{r}}_i - \mathbb{E}[\tilde{\mathbf{r}}])$. Empirically we validate that the losses are complementary.

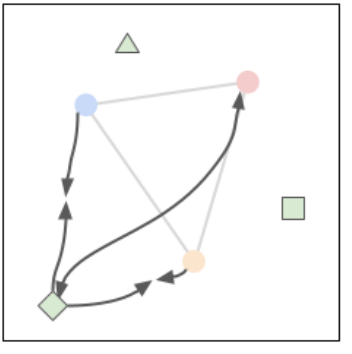

Figure 2: In standard Mixup, samples are aligned with interpolated classes only at the normalized softmax level (curly lines) and cannot generalize to other contrastive axes.

4 Experiments

Implementation details. All experiments are done on ResNet [18] and Wide ResNet [65] architectures. We train networks with Stochastic Gradient Descent for 200 epochs with a

contrastive axis		CIFAR-100 batch size		
class-axis	pair-axis	64	128	512
✓		74.90	76.75	76.17
	✓	75.38	77.62	76.09
✓	✓	**76.20**	**77.90**	**77.08**

	CIFAR-100			
	100%	50%	25%	10%
No Mixup	76.52	66.31	46.78	26.55
Mixup	77.33	68.39	49.68	28.21
IC-Mixup (c)	76.75	68.95	48.58	**30.38**
IC-Mixup (p)	77.62	**69.23**	**50.72**	28.29
IC-Mixup	**77.90**	68.89	50.48	30.19

Table 1: **Ablation studies on contrastive axes** (left) **and training size** (right) in Infinite Class Mixup. Consistently over the size of the batch and the fraction of the dataset size, combining both contrastive axes is effective and outperforms Mixup.

learning rate of 0.1 and a decay by factor 0.2 after 50, 100, and 150 epochs, with momentum 0.9 and weight decay 5e-4. Unless specified otherwise, the batch size is set to 128. For Mixup and Remix, we set α to 0.2, for RegMixup we set α to 20. For Remix, we follow [11] and set the interpolation threshold τ to 0.5 and the imbalance ratio threshold κ to 3. All experiments are run three times and we show the mean accuracy results over the runs.

4.1 Ablations and comparisons

Effect of dual-contrastive loss. In Table 1 (left), we show the effect of the two contrastive axes in Infinite Class Mixup, as well as their summed loss. We compare the three variants on CIFAR-100 with a ResNet-34 architecture on three batch sizes. We report the mean accuracy over three runs for all settings. For smaller batch sizes, we find that the pair-axis performs better, while we observe the reverse for large batch sizes. Across all three batch sizes, summing the losses of both contrastive axes is beneficial and improves the classification accuracy. We ran additional baselines where we contrast each interpolated example either to its two source classes, or to all original classes, instead of to all interpolated classes of the current batch. These baselines do not outperform the proposed setup, and they do not directly allow for optimization over the pair-axis. Hence we follow the (more standard) contrastive approach. We conclude that both contrastive axes of our Infinite Class Mixup are complementary and should be combined to improve down-stream performance.

Comparison to Mixup. In Table 1 (right), we draw a comparison to the baseline Empirical Risk Minimisation without Mixup and to conventional Mixup. We report results for various training set sizes of CIFAR-100, where for each percentage we perform random stratified sampling per class to obtain a reduced training set. When using the entire training set, Infinite Class Mixup obtains a mean accuracy of 77.90 compared to 77.33 for Mixup and 76.52 for the No Mixup baseline. The improvements for Infinite Class Mixup also hold for all other training set sizes, however depending on the amount of data the class-axis only or the pair-axis only variant might perform slightly better. Overall, the fewer examples available, the more profitable it is to apply Infinite Class Mixup. We conclude from this experiment that Infinite Class Mixup is a viable alternative to Mixup for image classification.

Comparison to RegMixup. In Figure 3 we draw a comparison to the recently introduced RegMixup [43]. RegMixup provides a simple yet effective change to Mixup; rather than only training on mixed pairs, the mixed loss is added to a cross-entropy loss over the individual samples in a batch. This allows for exploring balanced interpolations between classes

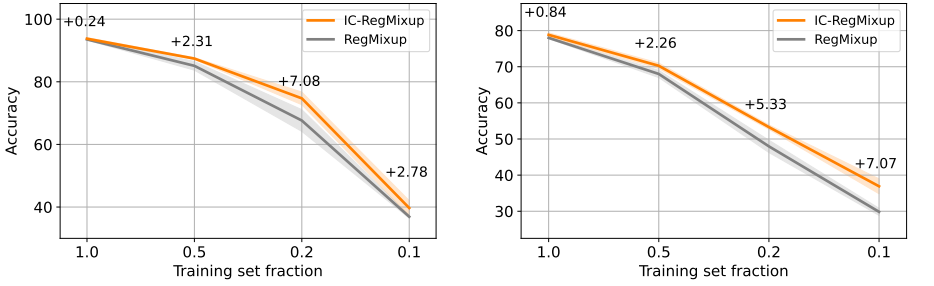

Figure 3: **Comparing RegMixup [43] to Infinite Class RegMixup across various dataset sizes** on CIFAR-10 (left) and CIFAR-100 (right) with a ResNet-34 architecture. On both datasets, Infinite Class RegMixup outperforms RegMixup. For data-constrained settings with smaller training sets, mixing with infinite classes is preferred.

($\alpha >> 1$), rather than sampling interpolations close to individual classes ($\alpha << 1$). We follow Pinto et al. [43] and use $\alpha = 20$. On CIFAR-100 we find that Infinite Class RegMixup improves the mean accuracy on CIFAR-100 from 77.99 to 78.83 compared to RegMixup. As the training set size decreases, the difference in performance increases, up to an improvement of 7.07 p.p. when using 10% of the training data, from 29.84 to 36.91. We observe similar improvements for CIFAR-10 and conclude that our Infinite Class formulation is also beneficial for RegMixup. Due to the improved results of RegMixup over Mixup overall, we recommend Infinite Class RegMixup for the best classification accuracy.

4.2 Learning in constrained and long-tailed settings

Data-constrained learning. In Table 2 (left), we show experiments on ciFAIR-100 and ciFAIR-10, two datasets created to mimic learning with limited labels [2, 6]. For this experiment, we train using the default hyperparameters akin to the other experiments. Overall, we find that mixing samples is an effective tool when dealing with limited samples. The best performance on both datasets is obtained using Infinite Class RegMixup. In Table 2 (middle), we compare to the state-of-the-art on ciFAIR-10. Brigato et al. [6] have recently shown the big effect of precise hyperparameter tuning in such data-constrained settings, with top results on a tuned WideResNet-16-8 architecture. When using the same hyperparameters and and architecture, supplemented with Infinite Class RegMixup, the results improve from 58.22 to 61.84, reiterating the potential of our approach in data-constrained settings.

Long-tailed recognition. For the imbalanced setting, we investigate on LT-CIFAR100 and LT-CIFAR10 for imbalanced ratios 0.1 and 0.01 [13]. We do not adapt network training to the long-tailed domain and start from standard empirical risk minimization on the imbalanced training sets. We then incorporate Mixup [66] and Remix, a variant of Mixup where the interpolation ratios of samples and classes is decoupled to account for long-tailed classes [11]. Specifically, given interpolation ratio λ and hyperparameters τ (interpolation threshold) and κ (imbalance ratio threshold), the interpolation ratio λ_y for the class probabilities is 0 if $n_i/n_j \geq \kappa$ and $\lambda < \kappa$, with n_i the sample ratio of class i. The interpolation ratio is 1 if $n_i/n_j \leq 1/\kappa$ and $1-\lambda < \kappa$, and λ otherwise. Remix favours class assignments to

	ciFAIR-100	ciFAIR-10
No Mixup	41.96	45.58
Mixup	43.83	47.63
IC-Mixup	43.11	49.03
RegMixup	47.55	53.17
IC-RegMixup	**47.67**	**55.54**

	ciFAIR-10
Bietti et al. [0]	51.03
Oyallon et al. [40]	54.21
Kayhan and Gemert [24]	55.00
Ulicny et al. [45]	56.50
Kobayashi [24]	57.50
Brigato et al. [0]	58.22
+ IC-RegMixup	**61.84**

	LT-CIFAR100		LT-CIFAR10	
	0.1	0.01	0.1	0.01
ERM	58.54	37.44	88.63	71.87
Mixup	62.68	39.21	89.63	72.82
IC-Mixup	**64.30**	**43.31**	**89.89**	**76.81**
	+1.62	+4.10	+0.26	+3.99
Remix	61.36	38.04	89.57	72.65
IC-Remix	**64.56**	**46.01**	**90.26**	**79.28**
	+3.20	+5.97	+0.67	+6.63

Table 2: **Learning in constrained and long-tailed settings with Infinite Class Mixup.** Left: Infinite Class Mixup on top of RegMixup performs best on datasets with few samples per class. Middle: This formulation also outperforms other methods optimised for data-constrained settings. Right: Infinite Class Mixup also benefits long-tailed recognition.

long-tailed classes. In Table 2 (right), we report the results for empirical risk minimisation, Mixup, Remix, and our Infinite Class variants of Mixup and Remix. On both datasets, we find that our Infinite Class formulations improve over the standard formulations, especially for larger imbalance ratios. On LT-CIFAR100 (using 0.01 as imbalance ratio) the performance improves from 39.21 to 43.31 for Mixup and from 38.04 to 46.01 for Remix. We observe similar performance gains on LT-CIFAR10. Overall, while Remix performs on par or slightly below Mixup, Infinite Class Remix obtains the highest scores. We thus conclude that Infinite Class Remix is beneficial for imbalanced learning.

4.3 What does Infinite Class Mixup learn differently?

We have performed analyses on the impact of linear interpolation between classifiers for Mixup. We investigate (i) the example confidence as a function of the interpolation ratio and (ii) the difference in classifier dot products as a function of the interpolation ratio. For both analyses, we take ResNet-34 networks trained on CIFAR-100 and sample a single test image for each class. Then we sample all image pairs and construct interpolated images using $0 \leq \lambda \leq 1$ with step size 0.1, thus resulting in 99K interpolated images.

Lower confidence for ambiguous interpolations. In Figure 4 (left), we show the mean confidence scores, computed by their class-independent squared norms, as a function of the interpolation ratio over all image pairs. We investigate networks trained with Mixup and with Infinite Class Mixup. As noted by Liu et al. [35], the larger the norm, the more confident the prediction. When training with Mixup, interpolated samples have similar average norms compared to their original samples. For Infinite Class Mixup, however, the average norm is higher for uninterpolated images. In other words, Infinite Class Mixup is better equipped at differentiating ambiguous mixed samples and canonical unmixed samples.

Better differentiation between interpolated classes. Finally, in Figure 4 (right), we show the mean and stddev class confidence difference as a function of the interpolation ratio. For each image pair and interpolation ratio, we feed the mixed image to the network and compute the dot product with the classifiers of the labels of the pair. We take the dot product of the first class and subtract the dot product with the other class. Infinite Class Mixup shows a stronger relation between the interpolation ratio and the class confidence score, indicating that our formulation learns to better separate classes as a function of their inter-class ambiguity.

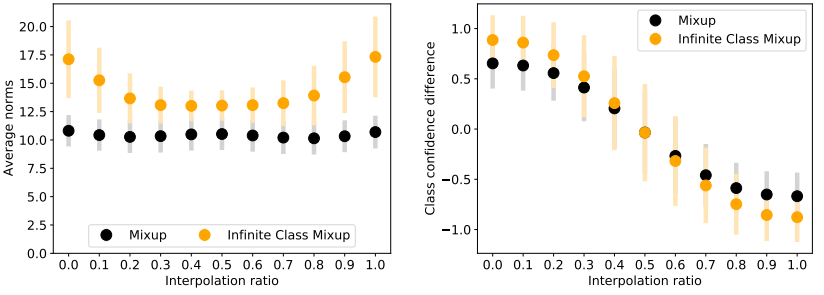

Figure 4: **Understanding what Infinite Class Mixup learns.** Left: Infinite Class Mixup is on average more confident for un-interpolated images, which helps to differentiate between classes during testing. Right: The further the interpolation ratio is from 0.5, the bigger the difference in class confidence in Infinite Class Mixup. Our approach learns to better separate classes as a function of their inter-class ambiguity.

5 Conclusions

Mixup is a popular and effective algorithm for network training where images and their corresponding label vectors are linearly interpolated to generate new samples and diversify the training set. A linear interpolation of label vectors does however not ensure linear behaviour between classifiers as a function of the interpolation ratio. We introduce Infinite Class Mixup, where we interpolate images and their corresponding *classifiers* directly. Each interpolated image is matched with a unique vector in classifier space, defined by a linear interpolation between the classifier vectors of the classes of the original image pair. We show how this setup can be optimised by contrasting simultaneously over both axes of the logit matrix between all image pairs and corresponding interpolated classes, improving classification in balanced, long-tailed, and data-constrained settings.

References

[1] Kyungjune Baek, Duhyeon Bang, and Hyunjung Shim. Gridmix: Strong regularization through local context mapping. *Pattern Recognition*, 2021.

[2] Björn Barz and Joachim Denzler. Do we train on test data? purging cifar of near-duplicates. *Journal of Imaging*, 2020.

[3] David Berthelot, Nicholas Carlini, Ian Goodfellow, Nicolas Papernot, Avital Oliver, and Colin A Raffel. Mixmatch: A holistic approach to semi-supervised learning. *NeurIPS*, 2019.

[4] Alberto Bietti, Grégoire Mialon, Dexiong Chen, and Julien Mairal. A kernel perspective for regularizing deep neural networks. In *ICML*, 2019.

[5] Rishi Bommasani, Drew A Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, et al. On the opportunities and risks of foundation models. *arXiv*, 2021.

[6] Lorenzo Brigato, Björn Barz, Luca Iocchi, and Joachim Denzler. Tune it or don't use it: Benchmarking data-efficient image classification. In *ICCVw*, 2021.

[7] Jie-Neng Chen, Shuyang Sun, Ju He, Philip HS Torr, Alan Yuille, and Song Bai. Transmix: Attend to mix for vision transformers. In *CVPR*, 2022.

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

[9] Yunlu Chen, Vincent Tao Hu, Efstratios Gavves, Thomas Mensink, Pascal Mettes, Pengwan Yang, and Cees G M Snoek. Pointmixup: Augmentation for point clouds. In *ECCV*, 2020.

[10] Ta-Ying Cheng, Hsuan-Ru Yang, Niki Trigoni, Hwann-Tzong Chen, and Tyng-Luh Liu. Pose adaptive dual mixup for few-shot single-view 3d reconstruction. In *AAAI*, 2022.

[11] Hsin-Ping Chou, Shih-Chieh Chang, Jia-Yu Pan, Wei Wei, and Da-Cheng Juan. Remix: rebalanced mixup. In *ECCV*, 2020.

[12] Ching-Yao Chuang and Youssef Mroueh. Fair mixup: Fairness via interpolation. In *ICLR*, 2021.

[13] Yin Cui, Menglin Jia, Tsung-Yi Lin, Yang Song, and Serge Belongie. Class-balanced loss based on effective number of samples. In *CVPR*, 2019.

[14] Adrian Galdran, Gustavo Carneiro, and Miguel A González Ballester. Balanced-mixup for highly imbalanced medical image classification. In *MICCAI*, 2021.

[15] Jixiang Gao, Jingjing Chen, Huazhu Fu, and Yu-Gang Jiang. Dynamic mixup for multi-label long-tailed food ingredient recognition. *TMM*, 2022.

[16] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

[17] Hongyu Guo, Yongyi Mao, and Richong Zhang. Mixup as locally linear out-of-manifold regularization. In *AAAI*, 2019.

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

[19] 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.

[20] Minui Hong, Jinwoo Choi, and Gunhee Kim. Stylemix: Separating content and style for enhanced data augmentation. In *CVPR*, 2021.

[21] Junlin Hou, Jilan Xu, Rui Feng, Yuejie Zhang, Fei Shan, and Weiya Shi. Cmc-cov19d: Contrastive mixup classification for covid-19 diagnosis. In *ICCVw*, 2021.

[22] Osman Semih Kayhan and Jan C van Gemert. On translation invariance in cnns: Convolutional layers can exploit absolute spatial location. In *CVPR*, 2020.

[23] Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. Supervised contrastive learning. *NeurIPS*, 2020.

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

[25] Jang-Hyun Kim, Wonho Choo, Hosan Jeong, and Hyun Oh Song. Co-mixup: Saliency guided joint mixup with supermodular diversity. In *ICLR*, 2021.

[26] JongMok Kim, Jooyoung Jang, Seunghyeon Seo, Jisoo Jeong, Jongkeun Na, and Nojun Kwak. Mum: Mix image tiles and unmix feature tiles for semi-supervised object detection. In *CVPR*, 2022.

[27] Takumi Kobayashi. T-vmf similarity for regularizing intraclass feature distribution. In *CVPR*, 2021.

[28] Takumi Kobayashi. Two-way multi-label loss. In *CVPR*, 2023.

[29] Soonki Kwon and Younghoon Lee. Explainability-based mix-up approach for text data augmentation. *KDD*, 2022.

[30] Dogyoon Lee, Jaeha Lee, Junhyeop Lee, Hyeongmin Lee, Minhyeok Lee, Sungmin Woo, and Sangyoun Lee. Regularization strategy for point cloud via rigidly mixed sample. In *CVPR*, 2021.

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

[32] Saehyung Lee, Hyungyu Lee, and Sungroh Yoon. Adversarial vertex mixup: Toward better adversarially robust generalization. In *CVPR*, 2020.

[33] Chong Liu, Fengda Zhu, Xiaojun Chang, Xiaodan Liang, Zongyuan Ge, and Yi-Dong Shen. Vision-language navigation with random environmental mixup. In *ICCV*, 2021.

[34] Jihao Liu, Boxiao Liu, Hang Zhou, Hongsheng Li, and Yu Liu. Tokenmix: Rethinking image mixing for data augmentation in vision transformers. In *ECCV*, 2022.

[35] Weiyang Liu, Zhen Liu, Zhiding Yu, Bo Dai, Rongmei Lin, Yisen Wang, James M Rehg, and Le Song. Decoupled networks. In *CVPR*, 2018.

[36] Zicheng Liu, Siyuan Li, Di Wu, Zihan Liu, Zhiyuan Chen, Lirong Wu, and Stan Z Li. Automix: Unveiling the power of mixup for stronger classifiers. In *ECCV*, 2022.

[37] Wang Lu, Jindong Wang, Yiqiang Chen, Sinno Jialin Pan, Chunyu Hu, and Xin Qin. Semantic-discriminative mixup for generalizable sensor-based cross-domain activity recognition. *IMWUT*, 2022.

[38] Puneet Mangla, Nupur Kumari, Abhishek Sinha, Mayank Singh, Balaji Krishnamurthy, and Vineeth N Balasubramanian. Charting the right manifold: Manifold mixup for few-shot learning. In *WACV*, 2020.

[39] Bastien Moysset and Ronaldo Messina. Manifold mixup improves text recognition with ctc loss. In *ICDAR*, 2019.

[40] Edouard Oyallon, Eugene Belilovsky, and Sergey Zagoruyko. Scaling the scattering transform: Deep hybrid networks. In *ICCV*, 2017.

[41] Tianyu Pang, Kun Xu, and Jun Zhu. Mixup inference: Better exploiting mixup to defend adversarial attacks. In *ICLR*, 2020.

[42] Yash Patel, Giorgos Tolias, and Jiří Matas. Recall@ k surrogate loss with large batches and similarity mixup. In *CVPR*, 2022.

[43] Francesco Pinto, Harry Yang, Ser-Nam Lim, Philip HS Torr, and Puneet K Dokania. Regmixup: Mixup as a regularizer can surprisingly improve accuracy and out distribution robustness. *arXiv*, 2022.

[44] Andreas Psaroudakis and Dimitrios Kollias. Mixaugment & mixup: Augmentation methods for facial expression recognition. In *CVPRw*, 2022.

[45] Sucheng Ren, Huiyu Wang, Zhengqi Gao, Shengfeng He, Alan Yuille, Yuyin Zhou, and Cihang Xie. A simple data mixing prior for improving self-supervised learning. In *CVPR*, 2022.

[46] Lichao Sun, Congying Xia, Wenpeng Yin, Tingting Liang, Philip S Yu, and Lifang He. Mixup-transformer: dynamic data augmentation for nlp tasks. In *COLING*, 2020.

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

[48] 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.

[49] Matej Ulicny, Vladimir A Krylov, and Rozenn Dahyot. Harmonic networks for image classification. In *BMVC*, 2019.

[50] Ardian Umam, Cheng-Kun Yang, Yung-Yu Chuang, Jen-Hui Chuang, and Yen-Yu Lin. Point mixswap: Attentional point cloud mixing via swapping matched structural divisions. In *ECCV*, 2022.

[51] Shashanka Venkataramanan, Ewa Kijak, Laurent Amsaleg, and Yannis Avrithis. Alignmixup: Improving representations by interpolating aligned features. In *CVPR*, 2022.

[52] 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 *ICML*, 2019.

[53] Devesh Walawalkar, Zhiqiang Shen, Zechun Liu, and Marios Savvides. Attentive cutmix: An enhanced data augmentation approach for deep learning based image classification. In *ICASSP*, 2020.

[54] Yiwei Wang, Wei Wang, Yuxuan Liang, Yujun Cai, and Bryan Hooi. Mixup for node and graph classification. In *WWW*, 2021.

[55] Yuan Wu, Diana Inkpen, and Ahmed El-Roby. Mixup regularized adversarial networks for multi-domain text classification. In *ICASSP*, 2021.

[56] Aoran Xiao, Jiaxing Huang, Dayan Guan, Kaiwen Cui, Shijian Lu, and Ling Shao. Polarmix: A general data augmentation technique for lidar point clouds. In *NeurIPS*, 2022.

[57] Bingrong Xu, Zhigang Zeng, Cheng Lian, and Zhengming Ding. Few-shot domain adaptation via mixup optimal transport. *TIP*, 2022.

[58] Bingrong Xu, Zhigang Zeng, Cheng Lian, and Zhengming Ding. Generative mixup networks for zero-shot learning. *TNNLS*, 2022.

[59] Minghao Xu, Jian Zhang, Bingbing Ni, Teng Li, Chengjie Wang, Qi Tian, and Wenjun Zhang. Adversarial domain adaptation with domain mixup. In *AAAI*, 2020.

[60] Chuanguang Yang, Zhulin An, Helong Zhou, Linhang Cai, Xiang Zhi, Jiwen Wu, Yongjun Xu, and Qian Zhang. Mixskd: Self-knowledge distillation from mixup for image recognition. In *ECCV*, 2022.

[61] Lingfeng Yang, Xiang Li, Borui Zhao, Renjie Song, and Jian Yang. Recursivemix: Mixed learning with history. In *NeurIPS*, 2022.

[62] Huaxiu Yao, Yiping Wang, Linjun Zhang, James Zou, and Chelsea Finn. C-mixup: Improving generalization in regression. In *NeurIPS*, 2022.

[63] Hao Yu, Huanyu Wang, and Jianxin Wu. Mixup without hesitation. In *ICIG*, 2021.

[64] 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.

[65] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. In *BMVC*, 2016.

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

[67] Jinlai Zhang, Lyujie Chen, Bo Ouyang, Binbin Liu, Jihong Zhu, Yujin Chen, Yanmei Meng, and Danfeng Wu. Pointcutmix: Regularization strategy for point cloud classification. *Neurocomputing*, 2022.

[68] Shaoyu Zhang, Chen Chen, Xiujuan Zhang, and Silong Peng. Label-occurrence-balanced mixup for long-tailed recognition. In *ICASSP*, 2022.

[69] Jiahao Zhao, Penghui Wei, and Wenji Mao. Robust neural text classification and entailment via mixup regularized adversarial training. In *SIGIR*, 2021.

[70] Qianyu Zhou, Zhengyang Feng, Qiqi Gu, Jiangmiao Pang, Guangliang Cheng, Xuequan Lu, Jianping Shi, and Lizhuang Ma. Context-aware mixup for domain adaptive semantic segmentation. *TCSVT*, 2022.

[71] Yingke Zhu, Tom Ko, and Brian Mak. Mixup learning strategies for text-independent speaker verification. In *Interspeech*, 2019.