Interpolation-aware models for train-test consistency in *Mixup*

Yinhan Hu, Congying Han*, Tiande Guo
School of Mathematical Sciences, University of Chinese Academy of Sciences, Beijing 100049, China

ABSTRACT

Mixup is a learning principle that trains a neural network on convex combinations of pairs of examples and their labels. Despite of its good performance, there is an inherent inconsistency between training and testing in *mixup*, which makes theoretical understanding difficult and hurts the performance in some cases. In this work, we propose *λ-mixup* to alleviate this inconsistency. Specifically, *λ-mixup* reformulates the model to take the interpolation coefficient (λ) as input as well, so that a class of models indexed by λ is learned and we can select one specific coefficient or multiple coefficients for ensembles depending on the testing distribution. We theoretically demonstrate that, with enough data and model capacity, *λ-mixup* can recover the original conditional distribution. Moreover, we conduct image classification tasks on multiple datasets, including CIFAR-10, CIFAR-100 and Tiny-Imagenet, showing that comparing with *mixup*, *λ-mixup* exhibits better generalization, calibration and robustness to adversarial attacks and out-of-distribution transformations.

Keywords: Image classification, mixup, calibration, robustness

1. INTRODUCTION

Data augmentation[1,2] is a widely used method for generating more data to improve the generalization performance and robustness. Generally, data augmentations are simple transformations designed by experts with specific domain knowledge to keep the labels unchanged in supervised setting. For images as inputs[3], such transformations include rotation, cropping, translation, and color jittering, etc. Different from this, *mixup*[4] is proposed to use the convex combinations of pairs of inputs and their labels to train deep models. And extensive experiments have shown that *mixup* improves the generalization performance, increases the robustness to adversarial examples and improves model calibration[5].

Despite of its effectiveness, *mixup* is inherently inconsistent between training and testing. Specifically, we use $\lambda \in [0,1]$ to denote the interpolation coefficient between samples, and P_λ to denote the data distribution induced by interpolating samples with λ. Without loss of generality, λ can be limited in $[0.5,1]$ for that P_λ is identical to $P_{1-\lambda}$ by symmetry.

In *mixup*, we force the model F to fit the data distribution P_λ for all $\lambda \in [0.5,1]$ during training and test the model usually on original data distribution, which is identical to P_1. This inconsistency brings some problems. Firstly, it makes theoretical analysis more difficult. Building connection between the training distribution and the testing distribution in *mixup* is tricky. It is hard to answer the question of when and why *mixup* works. Additionally, the inconsistency in *mixup* may restrain its power in practice. It is more difficult for the model to fit all interpolation distributions instead of just one. And the possible overlap between these distributions could cause performance degradation or even failure. Recently, Reference[6] also found that *mixup*, despite improving calibration, tend to be under-confident. So combining it with ensembles can harm calibration.

In this paper, we propose a method, termed *λ-mixup*, to tackle the problem of inconsistency. As in Figure 1, *λ-mixup* reformulates the model F to explicitly take interpolation coefficient λ as input as well. On the one hand, with proper design of the model architecture, such as parameter sharing across different λs, *λ-mixup* retains the regularization effect brought by fitting the interpolation distributions. On the other hand, *λ-mixup* reduces the inconsistency between training and testing, for that different interpolation distributions are fitted with different models essentially. During training, a class of models indexed by λ are learned to fit different P_λ respectively. Then different models can be chosen depending on test distribution. For in-distribution testing, we can simply set λ to be equal to 1. For out-of-distribution testing, we can use a model with a smaller λ or use ensembles of models with different λs to further boost the performance. Through experiments on image classification tasks, we show that *λ-mixup* is better than *mixup* on fitting the original data

* hancy@ucas.ac.cn

distribution, which leads to higher classification accuracy and better calibration performance. Furthermore, *λ-mixup* increases the robustness against adversarial examples and out-of-distribution transformations.

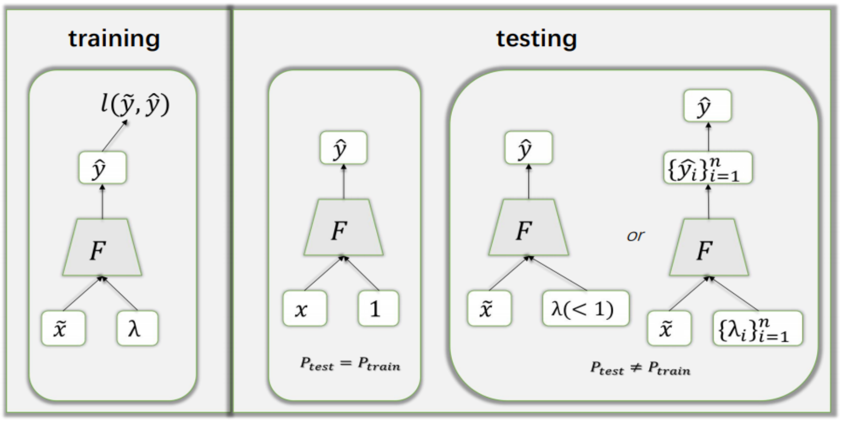

Figure 1. Overview of *λ-mixup*.

Note: The model takes λ as input explicitly during training and choose proper λ(s) for testing.

2. RELATED WORKS

2.1 Mixup

Mixup, introduced in Reference[4], augments data using convex combinations of pairs of inputs and their labels. Several works have tried to theoretically build connections between *mixup* training and empirical risk minimization and explain why *mixup* improves generalization and robustness. References[7,8] showed that *mixup* training can be viewed as minimizing the empirical loss along with a data-dependent regularization term. Reference[9] characterized a practical failure case of *mixup*, and also identified conditions under which *mixup* can provably minimize the original risk. Reference[10] proposed the concept of boundary thickness to analysis *mixup*.

Moreover, inspired by *mixup*, researchers come up with various methods of interpolating. Reference[11] introduced *manifold mixup* to apply *mixup* on intermediate representation as well. CutMix[12] cuts a rectangular region from one image and pastes it onto another, and generates label based on the area of each image. Instead of choosing regions randomly, saliency can be used to guide the interpolation to generate more meaningful data[13-15]. Reference[16] proposed AutoMix, learning to interpolate simultaneously. Reference[17] aligns features of two images before interpolating. Additionally, Reference[10] proposed to interpolate samples with noise for better robustness. Similarly, Reference[18] combines *mixup* and noise injecting. Different from the methods mentioned above, our method *λ-mixup* focuses on reducing the inconsistency between training and testing other than new ways of interpolation.

2.2 Calibration

Modern neural networks are poorly calibrated, which means there is a gap between model's confidence and its true correctness. Let X, Y denote input and label random variable, $\hat{Y}, \hat{P}$ denote the class prediction and associated confidence (predicted probability). One notion of miscalibration is the difference in expectation between confidence and accuracy: $E_{\hat{P}}[|P(\hat{Y} = Y| \hat{P} = p) - p|]$. To approximate the calibration error in expectation, Expected Calibration Error (ECE)[19] discretizes the probability interval into a fixed number of bins, and computes a weighted average of the bins' accuracy/confidence difference. More precisely,

$$\text{ECE} = \sum_{b=1}^{B} \frac{n_b}{N} |\text{acc}(b) - \text{conf}(b)| \tag{1}$$

where B denotes the number of bins, and each bin is indexed by $b \in \{1, 2, \cdots, B\}$. n_b is the number of predictions in bin b, N is the total number of data points, and acc(b) is the fraction of predictions in bin b that are correct (accuracy) and conf(b) is the mean of the probabilities in the bin (confidence)[20]. partitions probability interval into equally spaced bins.

For a trained model, most of the samples lie within the highest confidence bins, which cause the imbalance between bins. So an alternative is to partition probability interval so that each bin have the same number of samples. In this paper, we adopt the latter one to measure the calibration performance.

Recently, many variants of ECE have been proposed to measure calibration more properly[21-24]. Other methods measuring calibration include likelihood measures, Brier score[25], Bayesian methods[26], and conformal prediction[27].

Post-hoc transformations of predictions[20,28], model ensembles[29,30], and data augmentation have been shown to improve calibration. Specifically, Reference[5] showed that *mixup* can improve calibration. Furthermore, Reference[6] found that *mixup* usually makes model under-confident. We show that our method *λ-mixup* can further improve model calibration.

3. METHOD

Let $\mathcal{X} \subset \mathbb{R}^n$ be input space and $\mathcal{Y} = \{1,2,\cdots,K\}$ the output space. X and Y are random vectors(variables) on $\mathcal{X}$ and $\mathcal{Y}$ respectively, with $P(X,Y)$ as the joint distribution. Given a set of training data $\mathcal{D} = \{(x_i,y_i)\}_{i=1}^n$, where $(x_i,y_i) \sim P$, for $i = 1,\cdots,n$, $P(X,Y)$ can be approximated by the empirical distribution $\hat{P}(x,y) = \frac{1}{n}\sum_{i=1}^n \delta(x = x_i, y = y_i)$), where $\delta(x = x_i, y = y_i)$ is a Dirac mass centered at (x_i,y_i).

The classification task is to learn a function $F\colon \mathcal{X} \to \Delta^{K-1}$, where $\Delta^{K-1} = \{p \in \mathbb{R}^K : p_i \geq 0, \sum p_i = 1\}$ is probability simplex, to approximate the conditional distribution $P(Y|X)$. One way to learn F is to directly minimize the mean KL divergence between the empirical conditional distribution $\hat{P}(Y|X)$ and $F(x)$:

$$\min_F \mathbb{E}_{x\sim\hat{P}(x)} \mathrm{KL}(\hat{P}(Y|x) \| F(x)) \tag{2}$$

which is equivalent to Maximum Likelihood Estimate:

$$\max_F \mathbb{E}_{(x,y)\sim\hat{P}} \log\left(F_y(x)\right) = \frac{1}{n}\sum_{i=1}^n \log\left(F_{y_i}(x_i)\right) \tag{3}$$

Given $\lambda \in [0,1]$, we denote $\widehat{P_\lambda}(X,Y)$ on $\mathcal{X} \times \mathcal{Y}$ as the data distribution interpolated by λ. Specifically, the sampling process from $\widehat{P_\lambda}(X,Y)$ is as follow: two data points $(x_1,y_1),(x_2,y_2)$ are first sampled from $\hat{P}(X,Y)$, then x_λ is generated by interpolating x_1 and x_2, and y_λ is chosen from $\{y_1,y_2\}$ with probability λ. Mathematically,

$$x_\lambda = \lambda x_1 + (1-\lambda)x_2 \tag{4}$$

$$y_\lambda \sim P_{y_\lambda} = \lambda P_{y_1} + (1-\lambda)P_{y_2} \tag{5}$$

where P_{y_λ}, P_{y_2} denote the one-hot label corresponding y_1 and y_2. Obviously, $\widehat{P_\lambda}(X,Y)$ is identical to $\widehat{P_{1-\lambda}}(X,Y)$ by symmetry, and $\widehat{P_1}(X,Y)$ (with λ equals to 1) is $\hat{P}(X,Y)$. The *mixup* training objective is:

$$\max_F E_{(x_1,y_1),(x_2,y_2)\sim\hat{P},\lambda\sim p(\lambda)}[\lambda \log\left(F_{y_1}(x_\lambda)\right) + (1-\lambda)\log\left(F_{y_2}(x_\lambda)\right)] \tag{6}$$

which is equivalent to:

$$\min_F \mathbb{E}_{\lambda\sim p(\lambda)} \mathbb{E}_{x_\lambda\sim\hat{P}(x_\lambda)} \mathrm{KL}(\hat{P}(Y|x_\lambda) \| F(x_\lambda)) \tag{7}$$

We can see that *mixup* training is actually minimizing the KL divergence between *F* and all the interpolated conditional distribution $\widehat{P_\lambda}(Y|X)$. Intuitively, data interpolation may lose some information, which forces the model to focus on more robust features. So making classifier *F* to approximate $\widehat{P_\lambda}$ for different λs may regularize each other, which could benefit the learning. However, it brings problems too. Apparently, this causes the inconsistency between training and testing. After all, we care about the original $P(Y|X)$ the most. Also, the objective is much harder. In fact, we can see from experiments 4.1 that the closer λ is to 0.5, the harder it is to approximate $\widehat{P_\lambda}$ with *F*. More importantly, for different λs, $\widehat{P_\lambda}$ may overlap with one another, which causes contradictions, leading to under-fitting or total failure in some cases. To alleviate the problems, we propose *λ-mixup*, which reformulates *F* as a mapping from $\mathcal{X} \times [0,1]$ to Δ^{K-1}, which takes λ as input as well. The training objective remains the same as *mixup*:

$$\max_F \mathbb{E}_{(x_1,y_1),(x_2,y_2)\sim\hat{P},\lambda\sim p(\lambda)}[\lambda \log\left(F_{y_1}(x_\lambda,\lambda)\right) + (1-\lambda)\log\left(F_{y_2}(x_\lambda,\lambda)\right) \tag{8}$$

Instead of using one model to approximate all $\widehat{P_\lambda}$, *λ-mixup* learns a class of models indexed by λ to approximate $\widehat{P_\lambda}$ respectively. For simplicity, λ can be limited in [0.5, 1] due to symmetry.

Theorem 1. Suppose training data $\mathcal{D}$ contains infinite many data, which implies $\widehat{\mathrm{P}}$ equals to P, and F is continuous with respect to λ. Let F^* be the optimal solution of equation (8), with $\mathrm{p}(\lambda)$ is uniform distribution on [0, 1]. Then $\mathrm{F}^*(\mathrm{x}, 1) = \mathrm{P}(\mathrm{Y}|\mathrm{x})$, almost surely for $\mathrm{x} \sim \mathrm{P}(\mathrm{x})$.

Proof. Because F^* is the optimal solution, we have that for λ in almost everywhere in $[0, 1]$, $F^*(x, \lambda) = P(Y|x)$ almost surely for $x \sim P_\lambda$. Then by continuity of F^* andP_λ, we have $F^*(x, 1) = P(Y|x)$ almost surely for $x \sim P(x)$.

Theorem 1 shows that ideally *λ-mixup* can recover the true conditional distribution with λ set to be 1. So during testing, we use $F * (x, 1)$ for in-distribution data, and for out-of distribution data, because the model with smaller λ may focus on more robust features, we use smaller λ or use multiple different λs to further boost performance.

As for the model architecture, *mixup*'s superior performance indicates that this type of regularization actually benefits the learning process. So we do not want models with different λs differs too much. We accomplish this by sharing most parameters across different λs. Specifically, we can treat λ as time t in Neural Ordinary Differential Equations (Neural ODE)[31], and use the same architecture with the ODE function with t replaced by λ.

4. EXPERIMENTS

In this section, we conduct several experiments on image classification tasks to observe what *λ-mixup* actually learns and evaluate its performance in practice. The datasets we use include CIFAR-10, CIFAR-100 and Tiny-Imagenet. Both CIFAR-10 and CIFAR-100 have 50,000 training images and 10,000 testing images of which the resolution is 32×32, from 10 classes for CIFAR-10 and 100 classes for CIFAR-100. Tiny-Imagenet is a dataset of 120,000 labeled images belonging to 200 classes. Each of the 200 categories consists of 500 training images, 50 validation images, and 50 test images, all down-sampled to a fixed resolution of 64 × 64. The model architecture we use in our experiments is based on PreActResNet-18[32], with convolution layer replaced by ConcatSquashConv2d layer and linear layer replaced by ConcatSquashLinear layer in Reference[31]. Other than comparing with original *mixup*, for the number of parameters increases in the new model, we set another baseline which consistently inputs 1 instead of λ during training, referred as 1-*mixup*.

We set $p(\lambda)$ to be the uniform distribution on [0, 1], and train the models for 200 epochs with Stochastic Gradient Descent with momentum = 0.9. The initial learning rate is set to be 0.1, and factored by 0.1 on 100 epoch and 150 epoch. Without specification, we set $\lambda = 1$ for testing as the default choice.

4.1 Approximation to $\widehat{\mathrm{P}_\lambda}$

In this experiment, we compare how well the approximation of $\widehat{P_\lambda}$ between 1-*mixup* and *λ-mixup* across all λs. Specifically, we sample λs equally spaced in [0.5, 1], and construct $\widehat{P_\lambda^{train}}$ with training set, $\widehat{P_\lambda^{test}}$ with testing set. To reduce the computational cost, we use Monte Carlo methods to approximate the expected KL divergence:

$$\mathbb{E}_{x \sim \widehat{P_\lambda}(x)} \mathrm{KL}(\widehat{P_\lambda}(Y|x) \| F(x, \lambda))$$

The results are shown in Figure 2. From Figure 2, both training approximation error and testing approximation error become bigger as λ approaches 0.5, which is in line with the intuition that smaller λ induces harder task. Moreover, when λ is around 1, *λ-mixup* can get better approximation than 1-*mixup*. It shows that *λ-mixup* indeed reduces the inconsistency brought by *mixup* and fit the original distribution better. In addition, when λ is around 0.5, these two methods behave slightly different between CIFAR-10 and CIFAR100. There is an area near 0.5 that *λ-mixup* has a bigger error on CIFAR-10, whereas it performs better across almost all λs on CIFAR-100. One possible reason for this is that CIFAR-10 is a simpler dataset. 1-*mixup* brings stronger regularization effect, knowledge learned from bigger λ can directly guide the learning process for small ones, leading better approximation on relatively small λ. However, on a harder dataset CIFAR-100, this stronger regularization brings side effects, leading the performance goes down for all λs.

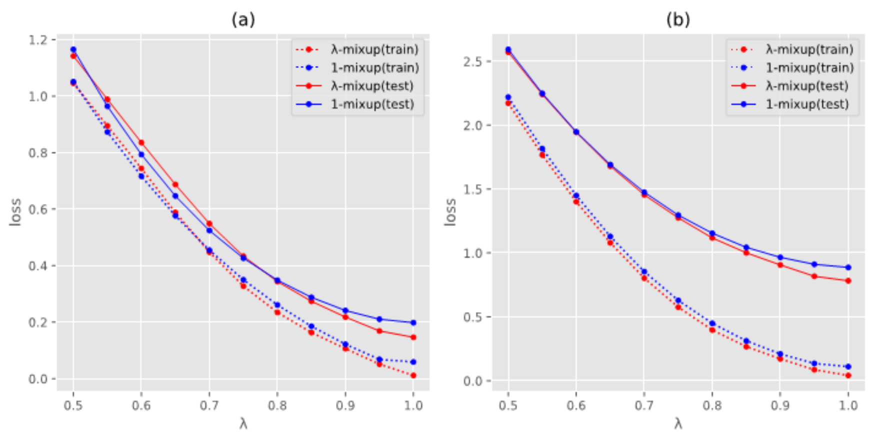

Figure 2. The estimated expected KL divergence between $\widehat{P_\lambda}$ and learned models across different λs. (a): CIFAR-10; (b): CIFAR-100.

4.2 Generalization

We conduct image classification experiments on CIFAR-10, CIFAR-100 and Tiny-Imagenet datasets to evaluate the generalization performance. Specifically, for each method, we train the model with 5 different random seeds and report the mean of best accuracy on validation set. The results are summarized in Table 1. From Table 1, 1-*mixup* performs better than *mixup*, because a slightly bigger model is used in 1-*mixup*. And λ-*mixup* consistently outperforms others on three datasets, especially gains 1.1% improvement on CIFAR-100 comparing with the model trained without λ.

Table 1. Classification accuracy (%) of different methods.

Accuracy	***mixup***	**1-*mixup***	***λ-mixup***
CIFAR-10	95.93	95.95	**96.06**
CIFAR-100	78.35	78.62	**79.7**
Tiny-Imagenet	62.49	62.88	**63.3**

4.3 Calibration

To verify the calibration ability of λ-*mixup*, we compare the expected calibration error (ECE) on CIFAR-10 and CIFAR-100 with the baselines.

Table 2. ECE of different methods.

ECE	***mixup***	**1-*mixup***	***λ-mixup***
CIFAR-10	0.063	0.063	**0.016**
CIFAR-100	0.097	0.095	**0.02**

As shown in Table 2, 1-*mixup* and *mixup* have similar ECE, but λ-*mixup* reduces ECE by a large margin. Also, Figure 3 plots a variant of reliability diagrams[33] on 1-*mixup* and λ-*mixup*. We bin the predictions into M = 10 intervals based on their confidence (probabilities) with each bin contains the same number of predictions and compute the difference between the average confidence and the average accuracy for each bin. A positive difference (Accuracy-Confidence) implies under-confidence; negative implies over-confidence; and zero implies perfect calibration. From figure 3, we can see that 1-*mixup* causes under-confidence and λ-*mixup* is slightly over-confident.

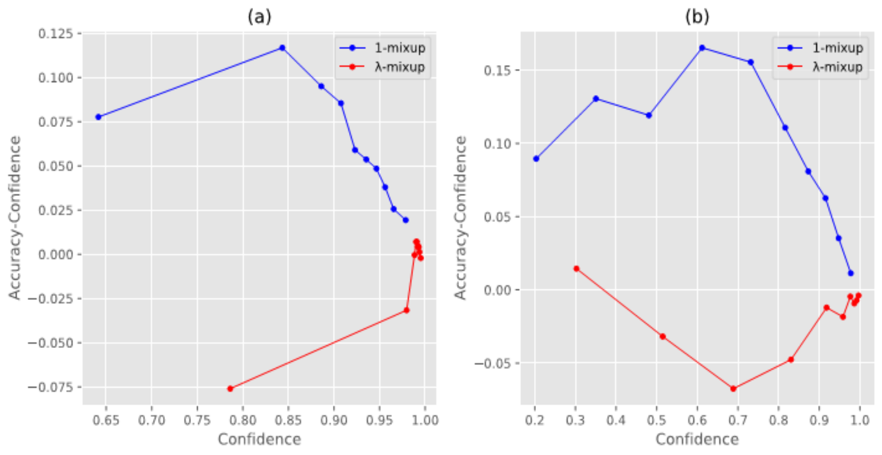

Figure 3. Reliability diagrams on CIFAR-10 and CIFAR-100. (a): CIFAR-10; (b): CIFAR-100.

4.4 Adversarial robustness

Adversarial robustness measures the performance of a model on adversarial examples. Adversarial examples are inputs formed by applying small but intentionally perturbations to examples, such that the model outputs incorrect results with high confidence. It has been shown that large neural networks degrade a lot under adversarial attacks. PGD is a simple and effective method generating adversarial examples based on the gradients of the targeted model. We evaluate the robustness of λ-*mixup* with λ set to be 1 during testing under PGD attacks on CIFAR10 and CIFAR-100. Specifically, we use an l_∞ attack with 10 steps and with step size being 2 pixels and report the results of three different attack ranges, namely 8-pixel, 6-pixel, and 4-pixel. As shown in Table 3, λ-*mixup* consistently improves adversarial robustness, especially on CIFAR-10.

Table 3. Classification accuracy (%) under PGD attacks.

PGD-attack		**4 pixels**	**6 pixels**	**8 pixels**
CIFAR-10	*mixup*	10.32	6.19	4.55
	1-*mixup*	8.78	5.36	3.96
	λ-*mixup*	**21.92**	**18.12**	**17.02**
CIFAR-100	*mixup*	0.2	0.04	0.02
	1-*mixup*	0.26	0.04	0.02
	λ-*mixup*	**0.88**	**0.3**	**0.18**

4.5 OOD robustness

Out-of-distribution (OOD) robustness measures the performance of a model on the data distribution which is different from training distribution. In realistic application, test data samples usually have some corruptions, causing mismatch between training distribution and testing distribution. So out-of-distribution robustness is important in some cases. Following Reference[34], we use both CIFAR-10C and CIFAR-100C to evaluate OOD robustness. We design two strategies to apply λ-*mixup* in these circumstances. One is to use a smaller λ, another is to use multiple λs for model ensembles. Figure 4 depicts the mean accuracy of models using different λ. We can see that, as expected, the highest performance is obtained with λ smaller than 1, around 0.75 for CIFAR-10 and 0.95 for CIFAR-100. And the best performance is comparable or better than 1-*mixup*. As for model ensembles, we choose n λs equally spaced between [0.5, 1] with 0.5 and 1 being chosen, and use the average of the outputs as the final prediction.

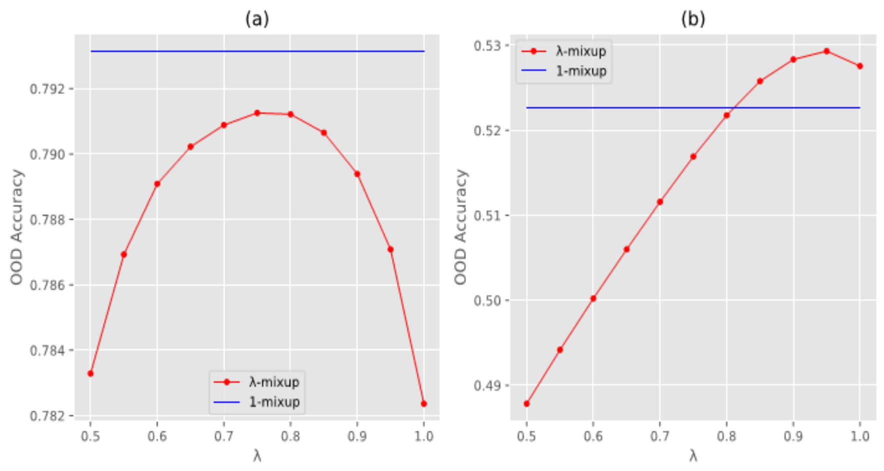

Figure 4. Mean accuracy on CIFAR-10C and CIFAR-100C using different λ. (a): CIFAR-10; (b): CIFAR-100.

Table 4. Mean accuracy (%) on CIFAR-10C and CIFAR-100C.

OOD	***mixup***	**1-*mixup***	***λ-mixup***	***λ-mixup* (3)**	***λ-mixup* (6)**	***λ-mixup* (11)**
CIFAR-10	79.31	79.31	78.24	79.35	**79.44**	79.44
CIFAR-100	52.07	52,27	**52,75**	52.72	52.5	52.4

Note: λ-*mixup*(n) stands for ensembles with n λs.

Table 4 reports the results with n taking the value of 3, 6, and 10. From Table 4, we see that model ensembles bring improvement comparing with 1-*mixup*. Noticeably, on CIFAR-100, using more λs degrades the performance comparing with the best λ-*mixup* with λ set to be 0.95. This is probably because that CIFAR-100 is a harder dataset. The model with λ near 0.5 loses too much useful information, which brings side effects in model ensembles.

5. CONCLUSION

To reduce the inherent inconsistency between training and testing in *mixup*, we propose λ-*mixup*, which forces the model to explicitly take as input the interpolation coefficient λ as well. We theoretically demonstrate that, ideally, with infinite many data and big enough model capacity, λ-*mixup* can recover the true conditional distribution $P(Y|X)$. During testing, different strategies of setting λ can be chosen depending on testing distribution, 1 for in-distribution data and smaller λ or multiple λs for out-of-distribution data. We conduct plenty of experiments showing that λ-*mixup* improves generalization, calibration and robustness.

In this paper, we maintained the regularization effect by parameter sharing across all λs, there could be more mathematical ways to realize it. Also, we didn't change the specific training procedure, it is promising to design more sophisticated training procedure to better utilize the mixed samples. Moreover, better methods of using the infinite many models trained by λ-*mixup* could potentially bring further improvements.

ACKNOWLEDGMENTS

This paper is supported by National Key RD Program of China (2021YFA1000403). This paper is supported by the National Natural Science Foundation of China (Nos. 11731013) and Supported by the Strategic Priority Research Program of Chinese Academy of Sciences (Grant No. XDA27000000) and the Fundamental Research Funds for the Central Universities.

REFERENCES

[1] Simard, P., LeCun, Y., Denker, J. and Victorri, B., "Transformation invariance in pattern recognition: tangent distance and tangent propagation," Neural Networks: Tricks of the Trade, 239-274 (1998).

[2] Wong, S., Gatt, A., Stamatescu, V. and McDonnell, M., "Understanding data augmentation for classification: when to warp?" Inter. Conf. on Digital Image Computing: Techniques and Applications (DICTA), 1-6 (2016).

[3] Krizhevsky, A., Sutskever, I. and Hinton, G., "Imagenet classification with deep convolutional neural networks," Communications of the ACM 60, 84-90 (2012).

[4] Zhang, H. Y., Ciss´e, M., Dauphin, Y. and Lopez-Paz, D., "Mixup: Beyond empirical risk minimization," 6th Inter. Conf. on Learning Representations, (2018).

[5] Thulasidasan, S., Chennupati, G. and Bilmes, J., "On mixup training: Improved calibration and predictive uncertainty for deep neural networks," Advances in Neural Information Processing Systems 32, (2019).

[6] Wen, Y. M., Jerfel, G., Muller, R., Dusenberry, M., Snoek, J., Lakshminarayanan, B. and Tran, D., "Combining ensembles and data augmentation can harm your calibration," 9th Inter. Conf. on Learning Representations, (2021).

[7] Zhang, L. J., Deng, Z., Kawaguchi, K., Ghorbani, A. and Zou, J., "How does mixup help with robustness and generalization?" 9th Inter. Conf. on Learning Representations, (2021).

[8] Carratino, L., Ciss'e, M., Jenatton, R. and Vert, J., "On mixup regularization," ArXiv preprint abs/2006.06049, (2020).

[9] Chidambaram, M., Wang, X., Hu, Y. Z., Wu, C. W. and Ge, R., "Towards understanding the data dependency of mixup-style training," ArXiv preprint abs/2110.07647, (2021).

[10] Yang, Y. Q., Khanna, R., Yu, Y. D., Gholami, A., Keutzer, K., Gonzalez, J., Ramchandran, K. and Mahoney, M., "Boundary thickness and robustness in learning models," Advances in Neural Information Processing Systems 33, 6223-34 (2020).

[11] Verma, V., Lamb, A., Beckham, C., Najafi, A., Mitliagkas, I., Lopez-Paz, D. and Bengio, Y., "Manifold mixup: Better representations by interpolating hidden states," Inter. Conf. on Machine Learning, 6438-47 (2019).

[12] Yun, S. D., Han, D., Oh, S. J., Chun, S., Choe, J. and Yoo, Y., "Cutmix: Regularization strategy to train strong classifiers with localizable features," Proc. of the IEEE/CVF Inter. Conf. on Computer Vision, 6023-32 (2019).

[13] Kim, J. H., Choo, W. and Song, H. O., "Puzzle mix: Exploiting saliency and local statistics for optimal mixup," Inter. Conf. on Machine Learning, 5275-85 (2020).

[14] Kim, J. H., Choo, W., Jeong, H. and Song, H. O., "Comixup: Saliency guided joint mixup with supermodular diversity," 9th Inter. Conf. on Learning Representations, (2021).

[15] Uddin, A., Monira, M., Shin, W., Chung, T. and Bae, S., "Saliencymix: A saliency guided data augmentation strategy for better regularization," 9th Inter. Conf. on Learning Representations, (2021).

[16] Liu, Z. C., Li, S. Y., Wu, D., Chen, Z. Y., Wu, L. R., Guo, J. Z. and Li, S., "Automix: Unveiling the power of mixup," ArXiv preprint abs/2103.13027, (2021).

[17] Venkataramanan, S., Avrithis, S., Kijak, E. and Amsaleg, L., "Alignmix: Improving representation by interpolating aligned features," ArXiv preprint abs/2103.15375, (2021).

[18] Lim, S. H., Erichson, N., Utrera, F., Xu, W. and Mahoney, M., "Noisy feature mixup," ArXiv preprint abs/2110.02180, (2021).

[19] Naeini, M., Cooper, G. and Hauskrecht, M., "Obtaining well calibrated probabilities using bayesian binning," 29th AAAI Conf. on Artificial Intelligence, (2015).

[20] Guo, C., Pleiss, G., Sun, Y. and Weinberger, K., "On calibration of modern neural networks," Proc. of the 34th Inter. Conf. on Machine Learning, 1321-30 (2017).

[21] Nixon, J., Dusenberry, M., Zhang, L. C., Jerfel, G., and Tran, D., "Measuring calibration in deep learning," CVPR workshops (Long Beach) 2(7), (2019).

[22] Roelofs, R., Cain, N., Shlens, J. and Mozer, M., "Mitigating bias in calibration error estimation," Int. Conf. on Artificial Intelligence and Statistics, 4036-54 (2022).

[23] Vaicenavicius, J., Widmann, D., Andersson, C., Lindsten, F., Roll, J. and Sch¨on, T., "Evaluating model calibration in classification," 22nd Inter. Conf. on Artificial Intelligence and Statistics, 3459-67 (2019).

[24] Gupta, K., Rahimi, A., Ajanthan, T., Mensink, T., Sminchisescu, C. and Hartley, R., "Calibration of neural networks using splines," 9th Inter. Conf. on Learning Representations, (2021).

[25] Brier, G., et al., "Verification of forecasts expressed in terms of probability," Monthly Weather Review 78(1), 1-3 (1950).

[26] Gelman, A., Carlin, J., Stern, H., Dunson, D., Vehtari, A. and Rubin, D., [Bayesian Data Analysis] 3rd ed., Chapman and Hall, London, (2013).

[27] Shafer, G. and Vovk, V., "A tutorial on conformal prediction," Journal of Machine Learning Research 9, 371-421 (2008).

[28] Kull, M., Perell´o-Nieto, M., K¨angsepp, M., Filho, T., Song, H. and Flach, P., "Beyond temperature scaling: Obtaining well-calibrated multiclass probabilities with dirichlet calibration," Advances in Neural Information Processing Systems 32, (2019).

[29] Lakshminarayanan, B., Pritzel, A. and Blundell, C., "Simple and scalable predictive uncertainty estimation using deep ensembles," Advances in Neural Information Processing Systems 30, (2017).

[30] Wen, Y. M., Tran, D. and Ba, J., "Batchensemble: An alternative approach to efficient ensemble and lifelong learning," 8th Inter. Conf. on Learning Representations, (2020).

[31] Chen, R., Rubanova, Y., Bettencourt, J. and Duvenaud, D., “Neural ordinary differential equations,” Advances in Neural Information Processing Systems 31, (2018).
[32] He, K. M., Zhang, X. Y., Ren, S. Q. and Sun, J., “Identity mappings in deep residual networks,” European Conf. on Computer Vision, 630-45 (2016).
[33] Degroot, M. and Fienberg, S., “The comparison and evaluation of forecasters,” The Statistician 32, 12-22 (1983).
[34] Hendrycks, D., Mu, N., Cubuk, E., Zoph, B., Gilmer, J. and Lakshminarayanan, B., “Augmix: A simple data processing method to improve robustness and uncertainty,” 8th Inter. Conf. on Learning Representations, (2020).