Improving Calibration of BatchEnsemble with Data Augmentation

Yeming Wen [*1†] **Ghassen Jerfel** [*1 2] **Rafael Muller** [1] **Michael W. Dusenberry** [1] **Jasper Snoek** [1]
Balaji Lakshminarayanan [1] **Dustin Tran** [1]

Abstract

Efficient ensembles such as BatchEnsemble are a simple drop-in approach to improving a model's accuracy and calibration across in- and out-of-distribution data. While they bridge the performance gap between single model performance and independent deep ensembles, their improvements on calibration are not as substantial as that on accuracy. We examine how to further improve calibration of these models. We investigate the role of data augmentation and show that augmentation techniques which improve single models can surprisingly make the ensemble calibration even worse. We propose a new data augmentation that fixes this pathology and improves BatchEnsemble's calibration. We empirically demonstrate the effectiveness of our approaches on in- and out-of-distribution CIFAR-10, CIFAR-100.

1. Introduction

Efficient ensembles such as BatchEnsemble (Wen et al., 2020; Dusenberry et al., 2020) outperform standard SGD-trained neural nets across accuracy, log-likelihood, and calibration, on both in-distribution test data and out-of-distribution corruptions, at the cost of negligible parameter overhead. In practice, efficient ensembles achieve almost the same accuracy as the much larger deep ensembles (Lakshminarayanan et al., 2017). However, a gap can be observed on calibration metrics, where on CIFAR-10, the naive ensemble improves single model Expected Calibration Error (ECE) from 2.3% to 0.7% while BatchEnsemble only improves the ECE from 2.3% to 1.8%. Motivated by this significant gap, we study how to further improve BatchEnsemble's calibration in this work. Post-processing techniques such as temperature scaling (Guo et al., 2017) can improve calibration and offer a potentially simple solution. However, recent literature shows they are sensitive to the choice of calibration metrics (Nixon et al., 2019) and are not robust to distribution shift (Ovadia et al., 2019). To this end, we examine methods to improve calibration based on the uncertainty decomposition of data, within the scope of BatchEnsemble.

We investigate *data uncertainty* related methods for the purpose of improving calibration. Data uncertainty, which accounts for the inherent stochasticity in a data gathering process, is the guiding principle behind state-of-the-art data augmentation strategies which improve robustness and uncertainty performance. In this work, we empirically study how data augmentation techniques such as Mixup (Zhang et al., 2018), and AugMix (Hendrycks et al., 2020), and moreover, how these techniques specifically perform through the lens of data uncertainty.

Contributions. We surprisingly find that Mixup improves calibration of individual models but degrades the calibration of the ensemble. We conclude from a detailed analysis that soft labels introduce a confidence bias that hinders its combination with ensembles. We propose CAMixup to correct this bias. Empirically, CAMixup significantly improves calibration on Wide ResNet 28-10 on CIFAR-10/100 and their corrupted variants. Finally, we combine CAMixup with AugMix and we find their improvements are complementary, leading to new *state-of-the-art calibration* on CIFAR-10/100 (e.g., 0.4% and 2.3% on CIFAR-10 and CIFAR-10C) with competitive accuracy (e.g., 97.5% and 89.8%). We successfully bridge the calibration gap between BatchEnsemble and deep ensembles.

2. Background

2.1. Efficient Ensembles

Ensembles have demonstrated state-of-the-art accuracy and calibration on both in- and out-of-distribution data (more background in Appendix B). However, the applicability of deep ensembles is limited as they require storing and predicting from multiple instances of models which can grow intractable. Therefore, recent research has focused on reducing both the memory and computational costs of

[*]Equal contribution [†]Work completed as an intern at Google Brain. [1]Google Brain, Mountain View, USA [2]Duke University, Durham, USA. Correspondence to: Yeming Wen <ywen@cs.toronto.edu>, Dustin Tran <trandustin@google.com>.

Presented at the ICML 2020 Workshop on Uncertainty and Robustness in Deep Learning.

deep ensembles. For example, BatchEnsemble (Wen et al., 2020) and Rank-1 BNN (Dusenberry et al., 2020) use multiplicative rank-1 perturbations to efficiently create ensembles of deep neural networks. Other efficient ensembling approaches leverage weight averaging techniques such as Polyak-Ruppert (Ruppert, 1988), checkpointing (Huang et al., 2017), and stochastic weight averaging (Izmailov et al., 2018) to collect multiple sets of weights during training and aggregate them to make predictions with only a single set. In this paper, we build on the rank-1 perturbation setup and specifically on BatchEnsemble which we briefly introduce in the following.

BatchEnsemble: For a given network layer i, we define the shared weight matrix among K ensemble members as $\mathbf{W}_i \in \mathbb{R}^{m \times d}$. For simplicity, we will omit i in the following. A tuple of trainable vectors $\mathbf{r}_k \in \mathbb{R}^m$ and $\mathbf{s}_k \in \mathbb{R}^n$ are associated with each ensemble member k. Then the weight matrix corresponding to each ensemble member in BatchEnsemble is given by:

$$\mathbf{W}'_k = \mathbf{W} \circ \mathbf{F}_k, \text{ where } \mathbf{F}_k = \mathbf{r}_k \mathbf{s}_k^\top \in \mathbb{R}^{m \times d},$$

where $\circ$ denotes the element-wise product. For a given layer, BatchEnsemble's forward pass can be rewritten as $\mathbf{y} = \phi(\mathbf{W}'_k \mathbf{x}) = \phi\left(\left(\mathbf{W} \circ \mathbf{r}_k \mathbf{s}_k^\top\right) \mathbf{x}\right) = \phi\left((\mathbf{W}(\mathbf{x} \circ \mathbf{s}_k)) \circ \mathbf{r}_k\right)$. where ϕ is the activation function, and $\mathbf{x} \in \mathbb{R}^d, \mathbf{y} \in \mathbb{R}^m$ is a single example. Using these rank-1 perturbations significantly reduces the number of parameters over a naive ensemble. To fully take advantage of hardware accelerators, the above can be vectorized as $\phi\left(\left((\mathbf{X} \circ \mathbf{S})\mathbf{W}^\top\right) \circ \mathbf{R}\right)$ where each row of $\mathbf{X} \in \mathbb{R}^{B \times d}$ is an example in a mini-batch and each row of $\mathbf{R} \in \mathbb{R}^{B \times m}$ and $\mathbf{S} \in \mathbb{R}^{B \times d}$ is a choice of ensemble member. Following Wen et al. (2020), we use ensemble size 4.

2.2. Data Augmentation

A simple and effective choice to capture data uncertainty is data augmentation. With input data augmentation, the input prior $p(\tilde{x} \mid x)$, where $\tilde{x}$ is the augmentation, posits semantically preserving transformations of x (e.g., random flips) and the output prior is a delta at $\tilde{y} = y$ in order to encourage the model to make invariant predictions under the transformations. In this work, we investigate Mixup (Zhang et al., 2018) and AugMix (Hendrycks et al., 2020).

Mixup: By varying both in the data prior to encourage linearly interpolating predictions, Mixup was shown to be effective for generalization and calibration of deep neural networks (Zhang et al., 2018; Thulasidasan et al., 2019). Some existing works study why Mixup leads to better generalization (Guo et al., 2018; Shimada et al., 2019) and improves adversarial robustness (Beckham et al., 2019; Pang et al., 2020; Mangla et al., 2020). Given an example (x_i, y_i), Mixup applies

$$\tilde{x}_i = \lambda x_i + (1 - \lambda) x_j, \quad \tilde{y}_i = \lambda y_i + (1 - \lambda) y_j.$$

Here, x_j is sampled from the training dataset (taken from the minibatch), and $\lambda \sim \text{Beta}(a, a)$ for a fixed hyperparameter $a > 0$. Another related data augmentation technique in this work is AugMix (Hendrycks et al., 2020), which we used in Section 3.3 to further improve calibration. We delegate the introduction of it to Appendix B.

3. BatchEnsemble with Data Augmentation

We investigate the role of data uncertainty in calibration, within the scope of BatchEnsemble. It is tempting to expect compounding benefits by combining state-of-the-art data augmentation techniques and ensembles. Surprisingly, we find that augmentation techniques which improve single models can be detrimental to ensemble calibration as illustrated by Mixup. We thus propose a novel data augmentation technique to address this pathology and improve calibration of BatchEnsemble.

3.1. Miscalibration of Ensembles with Mixup

We seek to combine data uncertainty techniques and ensembles to improve calibration. Ensembles are the best known and most straightforward approach to improving calibration (Ovadia et al., 2019). Tab. 1 and Tab. 4 confirm that ensembling improves calibration over a single model in both BatchEnsemble and deep (naive) ensembles. In addition, Thulasidasan et al. (2019) showed that Mixup improves calibration in a single model. However, Tab. 1 illustrates that combining Mixup with ensembles (by applying Mixup to each ensemble member) leads to *worse* calibration (for both small Mixup coefficient $\lambda = 0.2$ and large $\lambda = 1$). This is counter-intuitive because we tend to expect ensembles to be better calibrated than their individual members. While Tab. 1 only confirms this on BatchEnsemble, we can also consistently see this pattern on deep ensembles (Fig. 6 and Tab. 4 in the Appendix F).

Why do Mixup ensembles degrade calibration? Fig. 1 plots a variant of reliability diagrams (DeGroot & Fienberg, 1983), computing the difference between a model's predicted confidence and its accuracy. We group predictions into $M = 15$ bins according to their confidence, which we define as the value of the max softmax output, and compute the accuracy in each bin. Let B_m be the examples whose predicted confidence falls into the interval $(\frac{m-1}{M}, \frac{m}{M}]$. The accuracy and the confidence of bin B_m is

$$\text{Acc}(B_m) = \frac{1}{|B_m|} \sum_{x_i \in B_m} \mathbb{1}(\hat{y}_i = y_i),$$

$$\text{Conf}(B_m) = \frac{1}{|B_m|} \sum_{x_i \in B_m} \hat{p}_i,$$

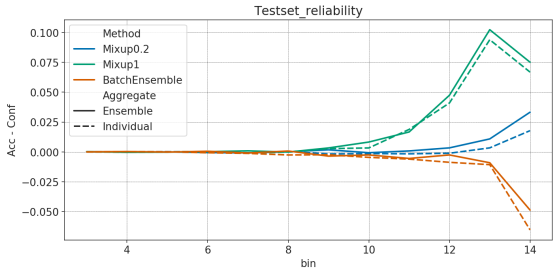
where $\hat{y}_i$ and y_i are the predicted and true labels for example x_i and $\hat{p}_i$ is the confidence of example x_i. A positive difference Acc $-$ Conf implies underconfidence; negative implies overconfidence; zero is perfect calibration.

Method/Metric		CIFAR-10 Acc(↑)	CIFAR-10 ECE(↓)	CIFAR-10 cA/cE	CIFAR-100 Acc(↑)	CIFAR-100 ECE(↓)	CIFAR-100 cA/cE
BatchEnsemble	In	95.88	2.3%	76.9/15.4%	80.64	8.7%	53.4/23.5%
	En	96.22	1.8%	77.5/12.9%	81.85	2.8%	54.1/19.1%
Mixup0.2 BE	In	96.43	0.8%	80.4/8.7%	81.44	1.5%	56.2/11.1%
	En	96.75	1.5%	**81.5/7.2%**	82.79	3.9%	58.0/9.10%
Mixup1 BE	In	96.67	5.5%	78.9/9.6%	81.32	6.6%	56.9/8.0%
	En	**96.98**	6.4%	80.0/9.3%	**83.12**	9.7%	59.3/**8.8%**
CAMixup0.2 BE	In	96.41	0.8%	80.2/10.8%	81.54	2.4%	56.7/13%
	En	96.63	**0.5%**	80.9/7.9%	82.79	**1.8%**	58.4/9.3%
CAMixup1 BE	In	96.56	1.8%	80.3/12.9%	81.56	2.4%	57.8/11%
	En	96.94	1.2%	81.1/9.7%	83.02	2.3%	**59.7**/8.9%

Table 1: Results for Wide ResNet-28-10 BatchEnsemble (Wen et al., 2020) (**BE**), averaged over 3 seeds. Numbers in blue highlight that combining Mixup (Zhang et al., 2018) and ensembles worsens calibration on in-distribution data. **In**: Individual model; **En**: Ensemble model; **cA/cE**: Accuracy and ECE on CIFAR-10C and CIFAR-100C (Hendrycks & Dietterich, 2019). Results on deep ensembles can be found in Tab. 4 in the appendix.

The backbone model in Fig. 1 is BatchEnsemble with ensemble size 4. The plot has 6 lines: we train three independent BatchEnsemble models with large, small, and no Mixup; and for each model, we compute the calibration of both ensemble and individual predictions. Fig. 1a shows that only Mixup models have positive (Acc − Conf) values on the test set, which suggests that Mixup encourages underconfidence. Mixup ensemble's positive value is also greater than Mixup individual's. This suggests that Mixup ensembles compound in encouraging underconfidence, leading to worse calibration than when not ensembling (Tab. 1).

Is this issue specific to mixup? At the core of the issue is that Mixup conflates data uncertainty with model uncertainty. Its soft labels can correct for the overconfidence of single models who have no other recourse. However, when combined with ensembles which provide model uncertainty, the correction is unnecessary. The pathology extends to other strategies which soften labels: image classification benchmarks tend to be deterministic, and soft labels encourage predictions on training data to be less confident about their true targets even if they are correct. The behavior can also be found with aggressive label smoothing; and we find that AugMix, which does not manipulate labels, does not share this pathology (Appendix D.2's Fig. 4).



(a) Reliability plot.

Figure 1: Reliability diagrams and confidence counts on CIFAR-10. Table 1 shows that ensemble individual BatchEnsemble improves test ECE from 2.3% to 1.8%. However, Mixup0.2-individual achieves test ECE 0.8% and Mixup0.2-ensemble worsens it to 1.5%. The plots show that ensembling makes predictions less confident. Ensembling and Mixup both encourage underconfidence, and when predictions are overconfident, this improves calibration. However, combining the two can lead to excessive underconfidence which thus worsens calibration.

Mixup displays a similar trend under distribution shift, as shown by Appendix D's Fig. 2. However, the models tend to be overconfident as one moves further from the original distribution (high corruption intensities), so encouraging underconfidence is not an issue. This explains why Mixup ensembles maintain low calibration error on out-of-distribution in Tab. 1. Thus, in the next section, we propose Confidence Adjusted Mixup (CAMixup) to adjust the under-confidence of Mixup ensembles on the test set while maintaining its good calibration on out-of-distribution dataset.

3.2. Confidence Adjusted Mixup Ensembles

We introduce Confidence adjusted Mixup (CAMixup) in this section. Instead of a uniform Mixup hyperparameter for all examples in the training set, we propose to adjust the hyperparameter of each class by the difference between its accuracy and confidence. Tab. 5 in the Appendix shows that some classes are more difficult than others for deep neural networks to predict correctly. The intuition of CAMixup is that we want to apply Mixup on hard examples on which models tend to be overconfident. On easy examples, we

Method/Metric	CIFAR-10			CIFAR-100		
	Acc(↑)	ECE(↓)	cA/cECE	Acc(↑)	ECE(↓)	cA/cECE
AugMix BE	97.36	1.02%	89.49/2.6%	83.57	2.96%	67.12/7.1%
AugMixup BE	**97.52**	1.71%	**90.05**/2.8%	**83.77**	4.19%	**69.26**/4.8%
CAugMixup BE	97.47	**0.45%**	89.81/**2.4%**	83.74	**2.35%**	68.71/**4.4%**

Table 2: Results for Wide ResNet-28-10 BatchEnsemble on in- and out-of-distribution CIFAR-10/100 with various data augmentations, averaged over 3 seeds. **AugMix**: AugMix + BatchEnsemble; **AugMixup**: AugMix + small Mixup BatchEnsemble; **CAugMixup**: AugMix + small CAMixup BatchEnsemble. Adding Mixup to AugMix model increases test accuracy and corrupt accuracy at the cost of calibration decay on testset. CAMixup bridges the gap.

impose standard data-augmentation without Mixup. This partially prevents Mixup models from being overconfident on difficult examples while maintaining its good calibration on out-of-distribution inputs.[1]

Denote the accuracy and confidence of class i as $\text{Acc}(C_i)$ and $\text{Conf}(C_i)$. We adjust Mixup's λ in Eqn. 2.2 by the sign of $\text{Acc}(C_i) - \text{Conf}(C_i)$, which are defined as $\text{Acc}(C_i) = \frac{1}{|C_i|}\sum_{x_j \in C_i} \mathbb{1}(\hat{y}_j = i)$ and $\text{Conf}(C_i) = \frac{1}{|C_i|}\sum_{x_j \in C_i} \hat{p}_i$.

$$\lambda_i = \begin{cases} 0 & \text{Acc}(C_i) > \text{Conf}(C_i) \\ \lambda & \text{Acc}(C_i) \leq \text{Conf}(C_i). \end{cases} \quad (1)$$

If the model is already underconfident ($\text{Acc}(C_i) > \text{Conf}(C_i)$), Mixup is not applied to examples in the class ($\lambda = 0$). However, if $\text{Acc}(C_i) \leq \text{Conf}(C_i)$, the model is overconfident on this class, and Mixup is applied to reduce model confidence. We compute the accuracy and confidence on a validation dataset after each training epoch.

Tab. 1 presents results of CAMixup. The experiments show that CAMixup improves calibration of Mixup ensembles by more than 2X on the test set using a small Mixup coefficient. Its calibration on out-of-distribution data is also on par with Mixup ensembles. The improvement increases to 5X if we use a large Mixup coefficient. We observe a minor decrease in test accuracy by at most 0.1%, but this is a worthwhile trade-off given the improvement in test ECE. Appendix D's Fig. 3b demonstrates how CAMixup corrects underconfidence. We provided the values of λ_i of each class at the end of training on CIFAR-10 in Appendix G. The classes for which the model assigns Mixup match the challenging classes presented in Tab. 5. A more detailed visualization of accuracy and ECE across distribution shift over all methods is delegated to Appendix D.2. CAMixup is also effective in deep ensembles but fails to match BatchEnsemble's improvement (Appendix F's Tab. 4).

[1]We focus on classification, where classes form a natural grouping of easy to hard examples. However, the same idea can be used on metadata that we'd like to balance uncertainty estimates, e.g., gender and age groups.

3.3. Combining CAMixup and AugMix

Mixup interpolates images between classes while AugMix maintains labels. AugMix improves calibration by applying further data augmentations with strong empirical results. It is natural to combine these two techniques as they manipulate data in complementary input space. This combination allows the model to encounter both diverse data augmentations and soft labels under a linearly interpolating regime. The naive combination AugMixup (AugMix + Mixup) can be written as $x = \lambda * \text{AugMix}(x_1) + (1-\lambda) * \text{AugMix}(x_2)$ and $y = \lambda * y_1 + (1-\lambda) * y_2$.

Consistent with results on Mixup, AugMixup leads to underconfidence on in-distribution data (Appendix D) and bad overall calibration (Tab. 2). With the proposed fix CAMixup, the combination CAugMix (CAMixup + AugMix) improves calibration while retaining highest accuracy for efficient ensembles. This successfully bridges the calibration gap between BatchEnsemble and AugMix deep ensembles (Appendix F's Tab. 4). We compare to deep ensembles with Augmix as they have the best calibration results (although worse accuracy without mixup).

To the best of our knowledge, these results are state-of-the-art in the literature: Dusenberry et al. (2020) report 0.8% and 1.8% for CIFAR-10 and CIFAR-100 along with 8% ECE and 11.7% ECE for corruptions; Guo et al. (2017) report 0.54% and 2.3% for the possibly smaller Wide ResNet 32 on CIFAR-10 and CIFAR-100 with temperature scaling, but Ovadia et al. (2019) demonstrated that temperature scaling does not extend to distribution shift.

4. Conclusion

We examine directions in both data uncertainty and model uncertainty for improving calibration of BatchEnsemble. We delegate more discussion and limitation of this work to Appendix E. For data uncertainty, we propose CAMixup which corrects the bad calibration of Mixup + BatchEnsemble. The resulting variant CAugMixup leads to the new state-of-the-art in calibration across CIFAR-10/100. We successfully bridge the calibration gap between BatchEnsemble and deep ensembles.

References

Beckham, C., Honari, S., Lamb, A., Verma, V., Ghadiri, F., Hjelm, R. D., and Pal, C. J. Adversarial mixup resynthesizers. *ArXiv*, abs/1903.02709, 2019.

Cubuk, E. D., Zoph, B., Mané, D., Vasudevan, V., and Le, Q. V. Autoaugment: Learning augmentation strategies from data. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 113–123, 2019a.

Cubuk, E. D., Zoph, B., Shlens, J., and Le, Q. V. Randaugment: Practical automated data augmentation with a reduced search space. *arXiv: Computer Vision and Pattern Recognition*, 2019b.

DeGroot, M. H. and Fienberg, S. E. The Comparison and Evaluation of Forecasters. *The Statistician*, 32(1/2):12, March 1983. ISSN 00390526. doi: 10.2307/2987588. URL https://www.jstor.org/stable/10.2307/2987588?origin=crossref.

Dietterich, T. G. Ensemble methods in machine learning. In *Multiple Classifier Systems*, 2000.

Dusenberry, M. W., Jerfel, G., Wen, Y., Ma, Y.-a., Snoek, J., Heller, K., Lakshminarayanan, B., and Tran, D. Efficient and scalable Bayesian neural nets with rank-1 factors. 2020.

Fort, S., Hu, H., and Lakshminarayanan, B. Deep Ensembles: A Loss Landscape Perspective. *arXiv preprint arXiv:1912.02757*, 2019.

Guo, C., Pleiss, G., Sun, Y., and Weinberger, K. Q. On Calibration of Modern Neural Networks. In *International Conference on Machine Learning (ICML)*, volume cs.LG. Cornell University Library, August 2017. URL http://arxiv.org/abs/1706.04599v2.

Guo, H., Mao, Y., and Zhang, R. Mixup as locally linear out-of-manifold regularization. In *AAAI*, 2018.

Hansen, L. K. and Salamon, P. Neural network ensembles. *IEEE Trans. Pattern Anal. Mach. Intell.*, 12:993–1001, 1990.

Hendrycks, D. and Dietterich, T. Benchmarking neural network robustness to common corruptions and perturbations. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=HJz6tiCqYm.

Hendrycks, D., Mu, N., Cubuk, E. D., Zoph, B., Gilmer, J., and Lakshminarayanan, B. Augmix: A simple data processing method to improve robustness and uncertainty. *ArXiv*, abs/1912.02781, 2020.

Huang, G., Sun, Y., Liu, Z., Sedra, D., and Weinberger, K. Q. Deep networks with stochastic depth. In *ECCV*, 2016.

Huang, G., Li, Y., Pleiss, G., Liu, Z., Hopcroft, J. E., and Weinberger, K. Q. Snapshot ensembles: Train 1, get m for free. *arXiv preprint arXiv:1704.00109*, 2017.

Izmailov, P., Podoprikhin, D., Garipov, T., Vetrov, D., and Wilson, A. G. Averaging weights leads to wider optima and better generalization. In *Uncertainty in Artificial Intelligence*, 2018.

Krizhevsky, A. Learning multiple layers of features from tiny images. 2009.

Lakshminarayanan, B., Pritzel, A., and Blundell, C. Simple and scalable predictive uncertainty estimation using deep ensembles. In *Neural Information Processing Systems*, 2017.

Maclin, R. and Opitz, D. W. Popular ensemble methods: An empirical study. *J. Artif. Intell. Res.*, 11:169–198, 1999.

Mangla, P., Singh, V., Havaldar, S. J., and Balasubramanian, V. N. Varmixup: Exploiting the latent space for robust training and inference. *ArXiv*, abs/2003.06566, 2020.

Nixon, J., Dusenberry, M., Zhang, L., Jerfel, G., and Tran, D. Measuring Calibration in Deep Learning. *arXiv:1904.01685 [cs, stat]*, April 2019. URL http://arxiv.org/abs/1904.01685.

Ovadia, Y., Fertig, E., Ren, J., Nado, Z., Sculley, D., Nowozin, S., Dillon, J. V., Lakshminarayanan, B., and Snoek, J. Can you trust your model's uncertainty? Evaluating predictive uncertainty under dataset shift. In *Neural Information Processing Systems*, 2019.

Pang, T., Xu, K., and Zhu, J. Mixup inference: Better exploiting mixup to defend adversarial attacks. *ArXiv*, abs/1909.11515, 2020.

Perrone, M. P. and Cooper, L. N. When networks disagree: Ensemble methods for hybrid neural networks. 1992.

Romero, A., Ballas, N., Kahou, S. E., Chassang, A., Gatta, C., and Bengio, Y. FitNets: Hints for thin deep nets. *CoRR*, abs/1412.6550, 2015.

Ruppert, D. Efficient estimations from a slowly convergent Robbins-Monro process. Technical report, Cornell University Operations Research and Industrial Engineering, 1988.

Shimada, T., Yamaguchi, S., Hayashi, K., and Kobayashi, S. Data interpolating prediction: Alternative interpretation of mixup. *ArXiv*, abs/1906.08412, 2019.

Srivastava, R. K., Greff, K., and Schmidhuber, J. Highway networks. *CoRR*, abs/1505.00387, 2015.

Thulasidasan, S., Chennupati, G., Bilmes, J. A., Bhattacharya, T., and Michalak, S. E. On mixup training: Improved calibration and predictive uncertainty for deep neural networks. In *NeurIPS*, 2019.

Wen, Y., Tran, D., and Ba, J. BatchEnsemble: An alternative approach to efficient ensemble and lifelong learning. In *International Conference on Learning Representations*, 2020.

Zhang, H., Cissé, M., Dauphin, Y., and Lopez-Paz, D. mixup: Beyond empirical risk minimization. *ArXiv*, abs/1710.09412, 2018.

A. Dataset Details

CIFAR: We consider two CIFAR datasets, CIFAR-10 and CIFAR-100 (Krizhevsky, 2009). Each consists of a training set of size 50K and a test set of size 10K. They are natural images with 32x32 pixels. Each class has 5,000 training images and 500 training images on CIFAR-10 and CIFAR-100 respectively. In our experiments, we follow the standard data pre-processing schemes including zero-padding with 4 pixels on each sise, random crop and horizon flip (Romero et al., 2015; Huang et al., 2016; Srivastava et al., 2015). If a training method requires validation dataset such as CAMixup, we use separate $2,500$ images from 50K training images as the validation set.

CIFAR-C: It's important to test whether our trained model is well calibrated under distribution shift. CIFAR-10 corruption dataset (Hendrycks & Dietterich, 2019) is designed to accomplish this. The dataset consists of 15 types of corruptions to the images. Each corruption types have 5 intensities. Thus, in total CIFAR-10C has 75 corrupted datasets. Notice that the corrupted dataset is used as a testset without training on it. Ovadia et al. (2019) benchmarked a number of methods on CIFAR-10 corruption. Similarly, we can apply the same corruptions to CIFAR-100 dataset to obtain CIFAR-100C.

B. Other Related Work

In this appendix section, we introduce some related works on Ensembles and Test Time Augmentation (TTA).

Ensembles: Aggregating the predictions of multiple models into an ensemble is a well-established strategy to improving generalization (Hansen & Salamon, 1990; Perrone & Cooper, 1992; Maclin & Opitz, 1999; Dietterich, 2000). In neural networks, composing an ensemble of models, each trained with a different random initialization, provides diverse predictions (Fort et al., 2019) that have been shown to outperform strong baselines on uncertainty estimation tasks (Lakshminarayanan et al., 2017). Ovadia et al. (2019) found that ensembles achieved state-of-the-art accuracy and uncertainty under a variety of distributional shift benchmarks.

AugMix: Searching or sampling over a set of data augmentation operations can lead to significant improvement on both generalization error and calibration (Cubuk et al., 2019a;b). In this paper, we examine AugMix (Hendrycks et al., 2020). AugMix applies a sum of augmentations, each with random weighting, and with a Jensen-Shannon consistency loss to encourage similarity across the augmentations. It achieves state-of-the-art calibration across in- and out-of-distribution tasks (Hendrycks et al., 2020). We focus on the data augmentation scheme and do not apply the consistency loss (how to best apply it with efficient ensembles remains unclear).

Dataset	CIFAR-10	CIFAR-100
ensemble_size	4	
base_learning_rate	0.1	
per_core_batch_size	64	
num_cores	8	
lr_decay_ratio	0.1	
train_epochs	250	
lr_decay_epochs	[80, 160, 200]	
l2	0.0001	0.0003
random_sign_init	0.5	0.75
SyncEnsemble_BN	False	True

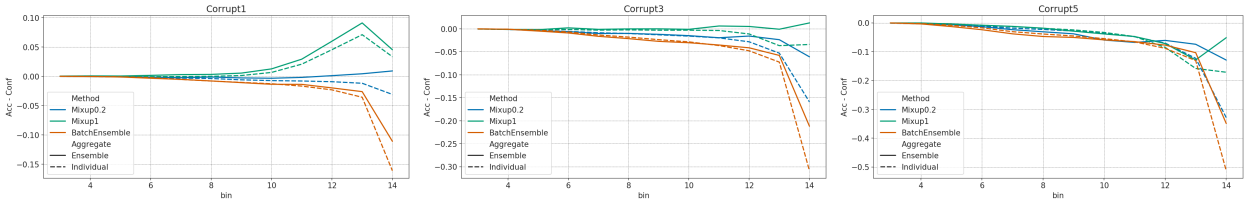
Table 3: Hyperparameters we used in Section 3 regarding to BatchEnsemble. The difference between CIFAR-10 and CIFAR-100 is l2, random_sign_init and whether to use SyncEnsemble_BN.

Let $\mathcal{O}$ be the set of data augmentation operations and k be the number of AugMix iterations. AugMix samples $w_1, \ldots, w_k \sim \text{Dirichlet}(a, \ldots, a)$ for a fixed hyperparameter $a > 0$ and $\text{op}_1, \ldots, \text{op}_k$ from $\mathcal{O}$. Given an interpolation parameter m, sampled from $\text{Beta}(a, a)$, which is denoted by $x_{aug} = \sum_{i=1}^{k} w_i \text{op}_i(x_{orig})$, the augmented input $\tilde{x}_{augmix}$ is:

$$\tilde{x}_{augmix} = m x_{orig} + (1 - m) x_{aug} \tag{2}$$

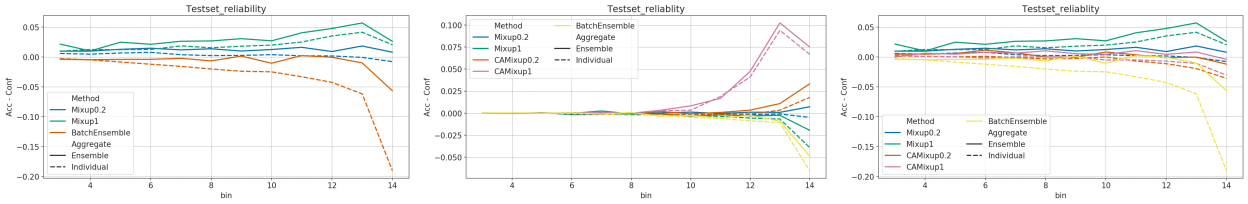
C. Hyperparameters in Section 3

We kept the same set of hyperparameters as the BatchEnsemble model in Wen et al. (2020). All hyperparameters can be found in Tab. 3. The most sensitive hyperparameter we found is whether to use ensemble batch norm, and the value of random_sign_init, which controls the standard deviation of Gaussian distributed initialization of $\mathbf{s}$ and $\mathbf{r}$. We kept BatchEnsemble CIFAR-10 the same as Wen et al. (2020), which does not deploy ensemble batch norm. We enable ensemble batch norm on CIFAR-100 and ImageNet. This allows us to use larger standard deviation in the initialization. The random_sign_init is -0.5 on CIFAR-10 and -0.75 on CIFAR-100 and -0.75 on ImageNet. In the code, we use negative value to denote the standard deviation of Gaussian distribution. A positive value corresponds to random sign initialization where the value represents the probability of being positive. In our case, we only use negative random_sign_init, which means we only consider Gaussian distributed initialization in this work.



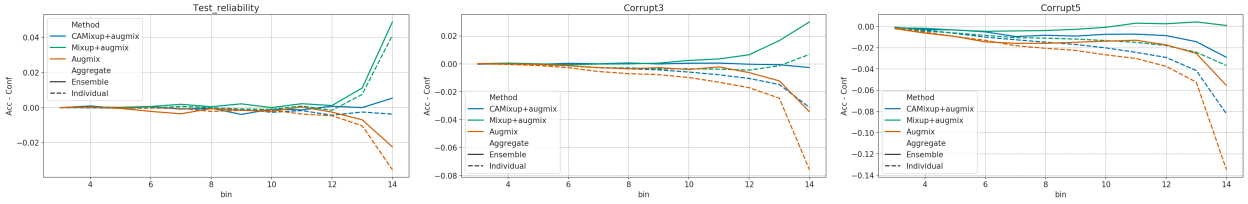
(a) Reliability on corrupt level 1. (b) Reliability corrupt level 3. (c) Reliability corrupt level 5.

Figure 2: Reliability diagram of WideResNet-28-10 **BatchEnsemble** with Mixup on **CIFAR-10**. This plot complements Figure 1. It shows that the miscalibration of Mixup + BatchEnsemble only happens on in-distribution dataset.



(a) Reliability on CIFAR-100. (b) Reliability on CIFAR-10. (c) Reliability on CIFAR-100.

Figure 3: WideResNet-28-10 **BatchEnsembles** with Mixup on **CIFAR-10** and **CIFAR-100 testset**. **a**: Mixup + ensembles is also underconfident on CIFAR-100. **b & c**: We compare Mixup and CAMixup on both CIFAR-10 and CIFAR-100 testset. CAMixup significantly improves ensembles calibration.



(a) Reliability on testset. (b) Reliability on corrupt level 3. (c) Reliability on corrupt level 5.

Figure 4: WideResNet-28-10 **BatchEnsembles** with different data augmentation techniques on **CIFAR-10**. It shows AugMix doesn't induce underconfidence model like Mixup. However, adding Mixup to AugMix leads to underconfidence. This motivates us to apply CAMixup on top of AugMix.

D. More Calibration Results of Mixup-BatchEnsemble

In Section 3.1, we demonstrated that combining Mixup and ensembles leads to worse calibration on testset. In this appendix section, we complement the above conclusion with the analysis on corrupted datasets and with data-augmentation techniques like AugMix.

D.1. Mixup-BatchEnsemble improves calibration under distribution shift

In this section, we provide some complementary results to Section 3.1. In Figure 1, we showed that Mixup + BatchEnsemble leads to miscalibrated model on testset because of underconfidence. Notice that an underconfidence model only leads to bad calibration on in-distribution dataset. It doesn't undermine its calibration on out-of-distribution because all models are overconfident as one moves further from the original distribution (high corruption intensities). As an evidence, Figure 2 shows that combining Mixup and ensembles actually improves calibration on corrupted dataset. For simplicity, we use corruption level 1, 3, 5 as the representatives of CIFAR-10C.

On CIFAR-100, we also observe miscalibration on testset. In Fig. 3a, we plotted the reliability of ensemble and individual predictions of Mixup-BatchEnsemble. The conclusion is consistent to what we observe on CIFAR-10. Individual Mixup-BatchEnsemble is well calibrated because of underconfidence. However, ensembling it compounds the underconfidence, leading to worse calibration. In Fig. 3b and Fig. 3c, we showed that CAMixup effectively corrects the confidence bias and achieves the best calibration con-

sistently on CIFAR-10 and CIFAR-100.

D.2. BatchEnsemble with AugMix, AugMixup and CAugMixup

In Table 2, we showed that combining AugMix and Mixup leads to worse calibration due to the underconfidence although AugMix itself does not. To better understand the insights beyond staring at scalars, we provided the reliability diagram analysis as well. In Figure 4a, we showed that the underconfidence issue of AugMixup (Augmix + Mixup) still exists. It suggests that applying CAMixup to Augmix can correct the underconfidence bias as what we showed in Fig. 3b and Fig. 3c. Notice that AugMixup only miscalibrates on testset rather than the corrupted dataset as shown in Fig. 4b and Appendix C (This is the same as Mixup + BatchEnsemble, see Fig. 2).

Thus, the fix we propose on Mixup BatchEnsemble can be directly applied to AugMixup BatchEnsemble. It is expected that CAugMixup (CAMixup + AugMix) can improve calibration on testset without undermining its calibration under data distribution shift. We demonstrated the empirical results in Table 2. In this appendix, we complement Table 2 by Figure 5. It visualizes the accuracy and ECE of all methods we consider in Section 3 on both in- and out-of-distribution dataset.

E. Limitations and Future Work

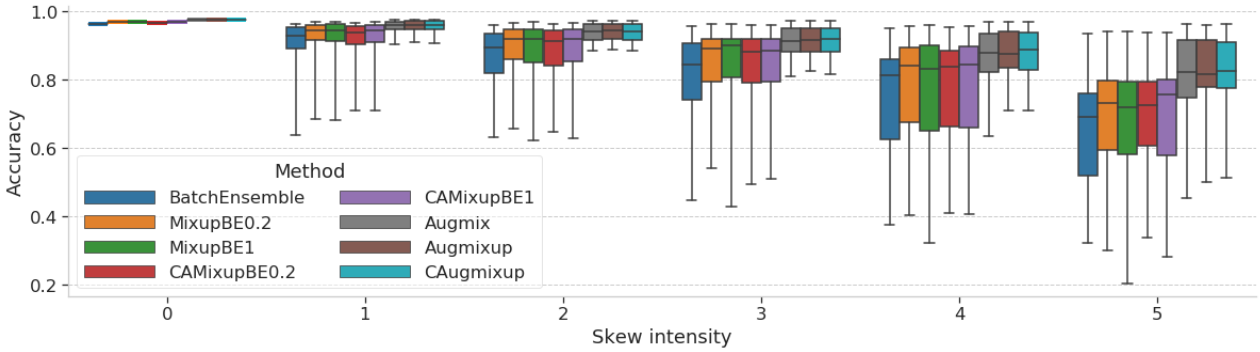
We describe limitations of our work. One limitation of CAMixup in this work is that all examples in the same class still share the same Mixup coefficient. This leaves a room for developing more fine-grained adaptive Mixup mechanism. However, this correlates another active research problem which is how to better define the training difficulty of each image given a deep network. Another limitation is we showed that CAMixup still cannot fully fix the miscalibration of Mixup + deep ensembles in Appendix F, due to the fact that Mixup + deep ensembles leads to even worse calibration than Mixup + BatchEnsemble. This raises a harder question which CAMixup cannot completely solve but also leaves more research room to further understand why Mixup is worse on deep ensembles and how to address it. We consider this work as an initial attempt to bridge the calibration gap between BatchEnsemble and deep ensembles. Thus, we leave the question on how to address the above issues to future work.

Next, we determine whether to use Mixup based on the reliability (Mean Accuracy - Mean Confidence) of each class on validation set. One concern CAMixup is bottlenecked by the number of classes in classification problem. We showed that this works on problems with 10 and 100 classes (CIFAR-10 and CIFAR-100). We are eager to examine this on datasets with more classes such as ImageNet. Additionally, Mixup only shows its effectiveness in the image domain. How to leverage it in other domains such as natural language processing is still unclear. This is an orthogonal research direction of this work.

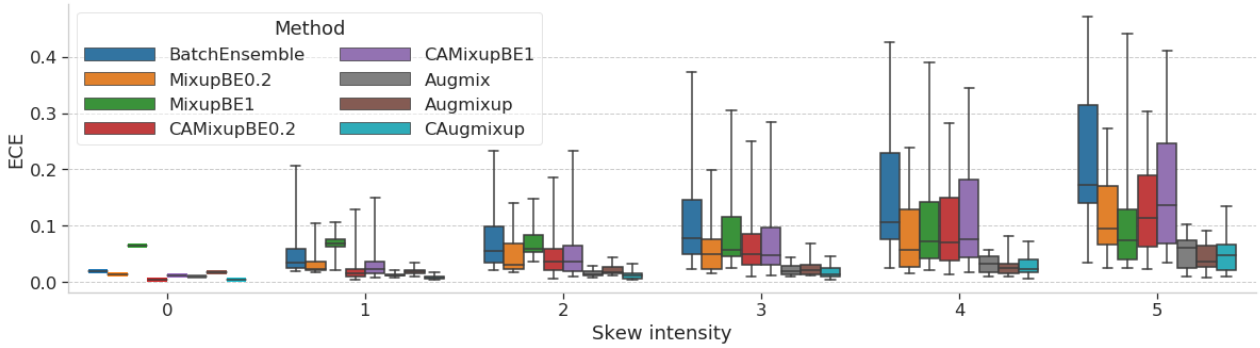
One natural idea to further improve calibration of BatchEnsemble is to combine two components we proposed in this work: data-augmentation and diversity regularizer. However, our preliminary experiments on this more sophisticated approaches did not work. Adding diversity regularizer or (test-time augmentation) TTA to CAugMixup BatchEnsemble (CAMixup + AugMix + BatchEnsemble) decreases its calibration on both in- and out-of-distribution datasets. We are eager to push on this further given the exciting progress from CAMixup.

We also obtained some negative results when exploring other sensible approaches to improve calibration of BatchEnsemble. We think these negative results are also inspiring. They can motivate the community to understand the insights why they do not work. One particular method we explore is to have a global mixture weights in BatchEnsemble. Instead of uniformly averaging the predictions from each ensemble member, we can have a trainable weighted average. This leverages the end-to-end differentiability of BatchEnsemble. However, we found this did not improve the current BatchEnsemble baseline. The resulting weights always favor one of the ensemble member.

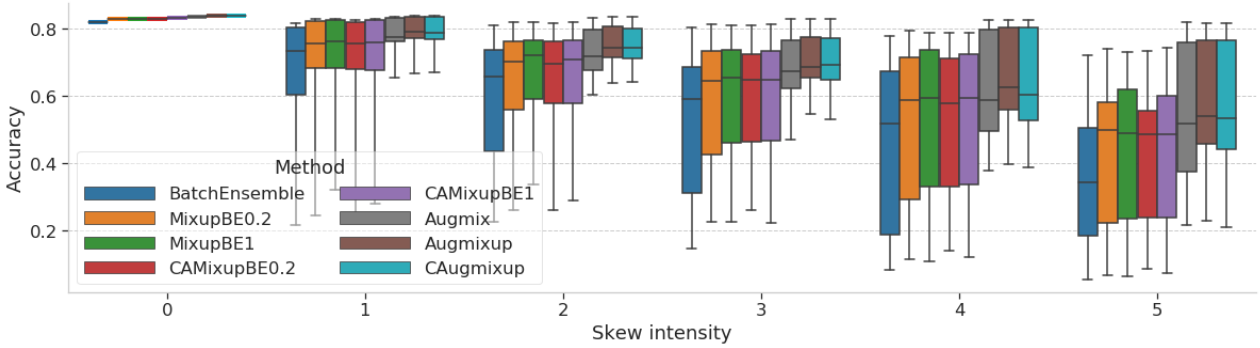
Another potential improvement we explore is higher rank perturbation in BatchEnsemble. Higher rank structure increases the expressive power of BatchEnsemble, leading to more diverse ensemble members. We implemented perturbation with rank-2 and rank-3 structures. However, we found that it is difficult to train a BatchEnsemble model with higher rank structure. We cannot achieve 100% training accuracy on neither CIFAR-10 or CIFAR-100. We postulate that this is because the optimizer (SGD) we use assumes an independent relationship among all trainable parameters. However, higher rank structures introduce more correlation among parameters (more than rank-1). How to address the training difficulty of higher rank BatchEnsemble is an interesting research direction.



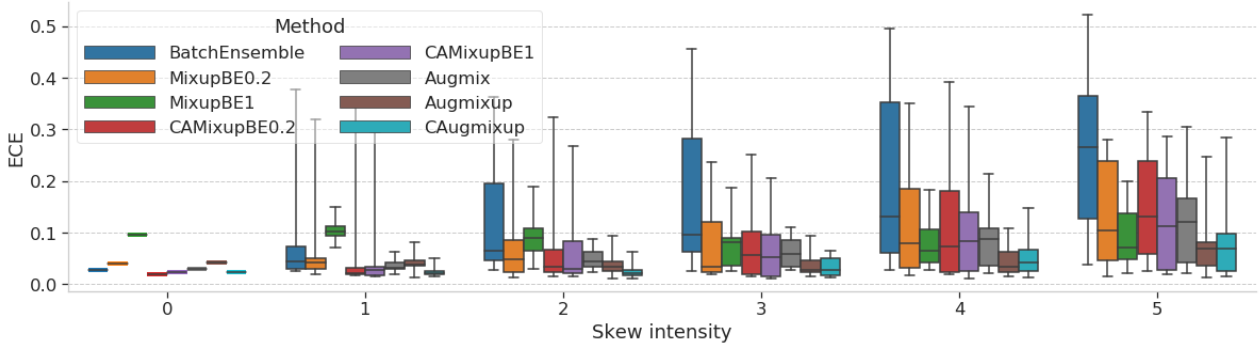
(a) Test Accuracy on CIFAR-10 and CIFAR-10C.



(b) Test ECE on CIFAR-10 and CIFAR-10C.



(c) Test Accuracy on CIFAR-100 and CIFAR-100C.



(d) Test ECE on CIFAR-100 and CIFAR-100C.

Figure 5: WideResNet-28-10 BatchEnsemble with different data augmentation techniques. It shows that our proposed method CAugMixup achieves the best trade-off among all methods we consider.

Dataset		CIFAR-10			CIFAR-100		
Metric		Acc(↑)	ECE(↓)	cA/cE	Acc(↑)	ECE(↓)	cA/cE
Deep Ensembles	In	96.0	2.31%	76.16/15.3%	79.8	8.5%	51.3/23.9%
	En	96.6	0.78%	76.77/9.78%	82.7	**2.1%**	54.1/13.8%
Mixup0.2 DE		97.02	2.76%	83.55/4.7%	83.61	4.85%	58.74/8.9%
Mixup1 DE		97.11	6.15%	83.33/8.0%	83.90	9.42%	61.02/8.9%
CAMixup0.2 DE		96.88	1.21%	81.94/4.6%	83.29	3.13%	57.92/9.3%
CAMixup1 DE		96.95	1.92%	83.01/4.4%	83.68	5.22%	59.18/8.6%
AugMix DE		97.39	**0.59%**	89.50/**3.3%**	84.15	5.13%	68.21/6.7%
AugMixup DE		**97.56**	2.71%	**90.03**/4.3%	**84.85**	6.86%	**69.31**/7.6%
CAugMixup DE		97.48	1.89%	89.94/4.7%	84.64	5.29%	69.19/**5.9%**

Table 4: Mixup/AugMix/AugMixup/CAugMixup on **deep ensembles**. We can conclude that Mixup worsens ensemble predictions in deep ensembles as well as in BatchEnsemble (See Fig. 6a and the bad calibration of Mixup0.2 DE & Mixup1 DE in this table). This suggests we can use CAMixup on deep ensembles as well. However, the improvement is not as obvious as it is on BatchEnsemble, leading to the fact that AugMix is the most calibrated (in- and out-of-distribution) data augmentation strategy on deep ensembles.

F. Deep Ensembles with Mixup

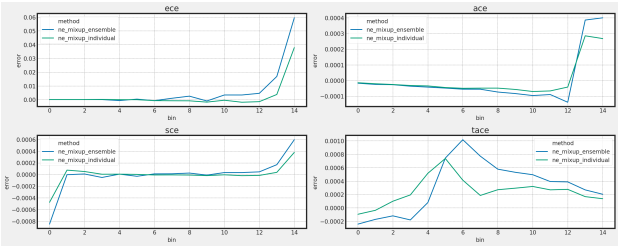
We showed that CAMixup improves Mixup BatchEnsemble calibration on testset without undermining its calibration under distribution shift in Section 3.2. In this section, we show that the improvement can also be observed on deep ensembles. In Fig. 6, we showed the underconfidence bias we observed on Mixup + BatchEnsemble also exists on Mixup + deep ensembles, with an even more obvious trend. Beyond commonly used ECE measure, we also explore other calibration measures. They further confirmed our underconfidence intuition. We provide some brief explanation on how to calculate ACE, SCE and TACE.

ACE measure is the same as ECE except for the binning scheme. Rather than equally divide the confidence evenly into several bins, ACE choses an adaptive scheme which spaces the bin intervals so that each contains an equal number of predictions. SCE is the same as ECE except that it accounts for all classes into calibration measure rather than just looking at the class with maximum probability. The softmax predictions induce infinitesimal probabilities. These tiny predictions can wash out the calibration score. TACE is proposed to set a threshold to only include predictions with large predictive probability, to address the above issue.

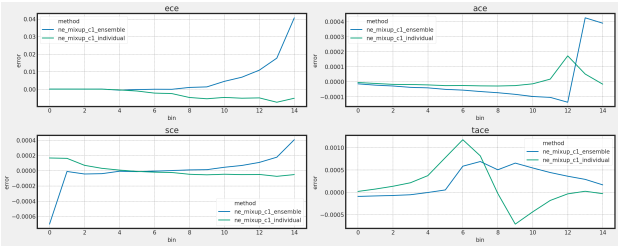
We present the results of Mixup, CAMixup, AugMix, AugMixup and CAugMixup on deep ensembles in Tab. 4. We notice that the improvement of CAMixup on deep ensembles is smaller than its improvement on BatchEnsemble. We postulate that this is because Mixup + deep ensembles is much badly calibrated than Mixup + BatchEnsemble. For example, AugMixup + deep ensembles achieve 2.71% and 6.86% ECE on CIFAR-10 and CIFAR-100. In the meanwhile, AugMixup + BatchEnsemble achieve 1.71% and 4.19%. Thus, even if CAMixup can improve the calibration of Mixup + deep ensembles, it still cannot beat AugMix + deep ensembles. As a result, when we say we close the calibration gap between BatchEnsemble and deep ensembles, we are comparing CAugMixup BatchEnsemble (BatchEnsemble + CAMixup + Augmix) to AugMix deep ensembles. This is because AugMix deep ensembles achieve the best calibration among all variants we tried. How to completely fix the underconfidence in deep ensembles is a natural extension of this work. Since we focus on bridging the calibration gap between BatchEnsemble and deep ensembles, we delegate the complete fix in deep ensembles to the future work.

G. Visualization of CAMixup Coefficient

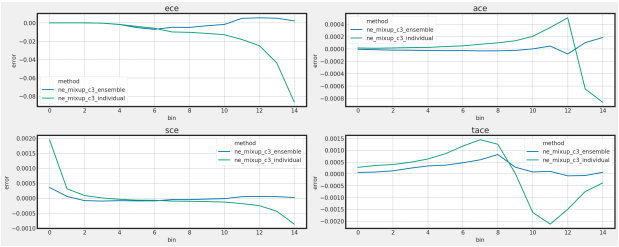
In this section, we first examine the test accuracy of each class in a WideResNet28-10 BatchEnsemble model trained on CIFAR-10 in Table 5. This inspires us to assign different Mixup coefficients to each class by their training difficulty, in order to correctly adjust the confidence of Mixup. We found that easier classes like class 1 and class 2 are not assigned mixup in Fig. 7. Harder classes like class 3 and class 4 are assigned mixup operation. These match the intuition that applying mixup on easy examples on which we want to be confident, while preventing overconfidence on hard examples.



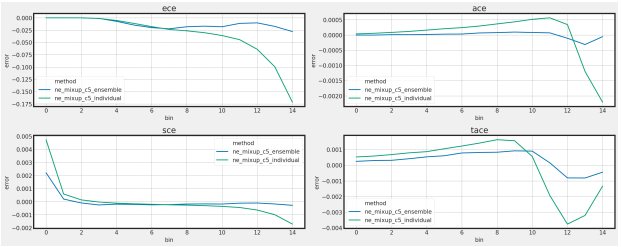
(a) Reliability on testset.



(b) Reliability on corrupt level 3.



(c) Reliability on corrupt level 3.



(d) Reliability on corrupt level 5.

Figure 6: WideResNet-28-10 **Deep Ensembles** with Mixup on CIFAR-10. We plotted the reliability diagram of ensemble and individual predictions. Besides ECE, we also plotted other calibration metrics such as ACE, SCE and TACE proposed in Nixon et al. (2019). All metrics verify the conclusion that Mixup + Ensembles leads to underconfidence on testset.

Table 5: Per class test accuracy of WideResNet-28-10 **BatchEnsemble** model on **CIFAR-10**. The average test accuracy is 96.22.

	Class 1	Class 2	Class 3	Class 4	Class 5	Class 6	Class 7	Class 8	Class 9	Class 10
Test Acc	96.65	97.78	94.42	90.62	96.99	93.53	97.54	96.88	97.88	96.43

```
tf.Tensor(
[[ 0.00589937  0.01310045 -0.00892758 -0.02443546 -0.00026906  0.01391917
   0.00239867  0.00897223  0.0042854  -0.00256133]
 [ 0.01087421  0.01225281 -0.01574224 -0.01335502 -0.00226063 -0.00933707
  -0.00162303  0.00742406  0.00376123  0.01478541]
 [ 0.01999706  0.01883298 -0.00297344 -0.01457781  0.01298314  0.00137413
   0.01013154  0.01584345  0.0017342   0.00031549]
 [ 0.00601727  0.01783884 -0.01117814 -0.03307533  0.02022719 -0.01600844
  -0.00079262  0.01721436  0.02901292  0.02871311]], shape=(4, 10), dtype=float32)
mixup coeff
tf.Tensor(
[[1.   1.   0.25 0.25 0.25 1.   1.   1.   1.   0.25]
 [1.   1.   0.25 0.25 0.25 0.25 0.25 1.   1.   1.  ]
 [1.   1.   0.25 0.25 1.   1.   1.   1.   1.   1.  ]
 [1.   1.   0.25 0.25 1.   0.25 0.25 1.   1.   1.  ]], shape=(4, 10), dtype=float32)
```

Figure 7: WideResNet-28-10 **BatchEnsemble** with different data augmentation techniques. It shows that our proposed method CAugMixup achieves the best trade-off among all methods we consider.