Using Mixup as a Regularizer Can Surprisingly Improve Accuracy and Out Distribution Robustness

Anonymous Author(s)
Affiliation
Address
email

Abstract

We show that the effectiveness of the well celebrated Mixup (Zhang et al., 2018) can be further improved if instead of using it as the sole learning objective, it is being utilized as an additional regularizer to the standard cross-entropy loss. This simple change not just provides much improved accuracy but also significantly improves the quality of the predictive uncertainty estimation of Mixup in most cases under various forms of covariate shifts and out-of-distribution detection experiments. Note, standard Mixup otherwise yield much degraded performance on out-of-distribution detection experiments, perhaps, as we show empirically, because of its tendency to learn models that throughout exhibit high-entropy which makes it difficult to differentiate between in-distribution and out-distribution samples. To show the efficacy of our approach (RegMixup), we provide thorough analysis and experiments on vision datasets (CIFAR-10/100 and ImageNet) and compare it with a suite of well-known approaches for reliable uncertainty estimation.

1 Introduction

In real-world machine learning applications one is interested in obtaining models that can reliably process novel inputs. However, though deep learning models have enabled breakthroughs in multiple fields, they are known to be unreliable when exposed to samples obtained from a distribution that is different from the training distribution. Larger the extent of this difference between train and test distributions, more unreliable these models normally are. This has led to a growing interest in developing approaches that encourage reliable predictions from a model even when they are exposed to unseen situations Liu et al. (2020a,b); Wen et al. (2021); Lakshminarayanan et al. (2017). Most of these recent approaches either use expensive ensembles, or propose non-trivial modifications to the neural network architectures in order to obtain reliable models. These approaches, in most cases, trade in-distribution performance (accuracy) to gain reliability when exposed to: (1) out-of-distribution (OOD) samples; and (2) covariate shift (CS) (Quionero-Candela et al., 2009) samples.

Towards developing reliable models, we investigate the well known Mixup (Zhang et al., 2018) as it is extremely popular in improving a model's accuracy and robustness Wen et al. (2021); Hendrycks et al. (2020b). It has already been observed that Mixup can help in retaining good accuracy when the test inputs are affected by superficial variations that do not affect the target label (i.e., they undergo CS) (Hendrycks et al., 2020b). However, we find that its performance degrades significantly when exposed to completely unseen samples with potentially different labels than the ones it was exposed to during training (OOD). This is crucial as it is not guaranteed that in real-world test samples will always belong to one of the categories seen during training, and in such situations, a model should be able to reliably reject those samples instead of making wrong predictions. We observe that the primary reason for such poor OOD performance is that Mixup, because of the way it is trained, ends up providing high predictive entropy for almost all the samples it receives. Therefore, it becomes

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

difficult to differentiate in-distribution samples from out-of-distribution ones. We would like to highlight that our observation is in contrast to the prior work (Thulasidasan et al., 2019) which suggests that Mixup provides reliable uncertainty estimates for OOD data as well.

We propose a simple yet effective fix to the aforementioned issue with Mixup: we suggest to train on a mixture distribution that combines the original training data distribution and the Mixup vicinal distribution together. We call this approach RegMixup. We provide proper justifications behind this proposal and show much improved performance with this simple change to Mixup.

One of the core strengths of our approach is its **simplicity**. As opposed to recently proposed techniques to improve uncertainty estimation like SNGP Liu et al. (2020a) and DUQ (van Amersfoort et al., 2020), it does not require any modifications to the architecture and is extremely simple to implement. *It does not trade accuracy in order to improve uncertainty estimates*, and is a single deterministic model, hence, extremely efficient compared to the highly competitive Deep Ensembles (DE) (Lakshminarayanan et al., 2017). Summary of our contributions:

- We provide a simple modification to Mixup that significantly improves its in-distribution, covariate shift, and out-of-distribution performances.
- Through extensive experiments using ImageNet-1K, CIFAR10/100 and their various CS counterparts along with multiple OOD datasets we show that, overall, RegMixup outperforms recent state-of-the-art single-model approaches. In most cases, it outperforms the extremely competitive and expensive DE as well.

2 RegMixup: Mixup as a regularizer

Preliminary The principle of risk minimization Vapnik (1991) is to estimate a function $f \in \mathcal{F}$ that, for a given loss function $\ell(.,.)$, minimizes the expected risk over the entire data-distribution $P(\mathbf{x}, \mathbf{y})$. The risk to be optimized is defined as $R(f) = \int \ell(f(\mathbf{x}), \mathbf{y}) dP(\mathbf{x}, \mathbf{y})$. Since the distribution $P(\mathbf{x}, \mathbf{y})$ is unknown, a crude yet widely used approximation is to first obtain a training dataset $\mathcal{D} = \{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^n$ sampled from the distribution P and then obtain f by minimizing the empirical risk defined as $R_e(f) = 1/n \sum_{i=1}^n \ell(f(\mathbf{x}_i), \mathbf{y}_i)$. This is equivalent to approximating the entire data-distribution space by a finite n number of Delta distributions positioned at each $(\mathbf{x}_i, \mathbf{y}_i)$, written as $P_e(\mathbf{x}, \mathbf{y}) = 1/n \sum_{i=1}^n \delta_{\mathbf{x}_i}(\mathbf{x}) \delta_{\mathbf{y}_i}(\mathbf{y})$. This approximation to the original risk minimization is widely known as the *Empirical Risk Minimization (ERM)* Vapnik (1991).

ERM has been successfully used in a plenty of real-world applications and undoubtedly has provided efficient and accurate solutions to many learning problems. However, the quality of the ERM solution clearly depends on how closely P_e mimics the true distribution P, and on the capacity of the function class $\mathcal{F}$ as well. In situations where the function class is extremely rich, and models with very high capacity (like neural networks) are prone to undesirable behaviours such as overfitting and memorization, a good approximation to P is generally needed to enforce suitable inductive biases in the model. For a fixed training dataset, one could potentially fit a richer distribution, instead of a delta distribution, in the vicinity of each input-output pair to estimate a more informative risk computed in a *region* around each sample. This is precisely the principle behind the well-known *Vicinal Risk Minimization (VRM)* Chapelle et al. (2000). The approximate distribution in this case can be written as $P_v(\mathbf{x}, \mathbf{y}) = 1/n \sum_{i=1}^n P_{\mathbf{x}_i, \mathbf{y}_i}(\mathbf{x}, \mathbf{y})$[1]. Therefore, the vicinal risk boils down to

$$R_v(f) = \frac{1}{n} \sum_{i=1}^{n} \int \ell(f(\mathbf{x}), \mathbf{y}) dP_{\mathbf{x}_i, \mathbf{y}_i}(\mathbf{x}, \mathbf{y}). \tag{1}$$

In situations where the integral is intractable, Monte Carlo estimate with m samples can be used

$$\int \ell(f(\mathbf{x}), \mathbf{y}) dP_{\mathbf{x}_i, \mathbf{y}_i}(\mathbf{x}, \mathbf{y}) \approx \frac{1}{m} \sum_{j=1}^{m} \ell(f(\bar{\mathbf{x}}_j), \bar{\mathbf{y}}_j); \quad (\bar{\mathbf{x}}_j, \bar{\mathbf{y}}_j) \sim P_{\mathbf{x}_i, \mathbf{y}_i}(\mathbf{x}, \mathbf{y}). \tag{2}$$

Several approaches in the literature can be seen as a special instance of VRM. A widely used application is the procedure to obtain *robust base classifier* to design certifiable classifiers.

[1]Note, the original VRM paper used $P_{\mathbf{y}_i}(\mathbf{y}) = \delta_{\mathbf{y}_i}(\mathbf{y})$ which is simply a special case of this notation.

For example, if $P_{\mathbf{x}_i,\mathbf{y}_i}(\mathbf{x},\mathbf{y}) = P_{\mathbf{x}_i}(\mathbf{x})\delta_{\mathbf{y}_i}(\mathbf{y})$ and $P_{\mathbf{x}_i}(\mathbf{x})$ is a Gaussian distribution centered at $\mathbf{x}_i$, Eq. (2) can be computed by taking the average loss over gaussian perturbed inputs $\bar{\mathbf{x}}$ while keeping the target labels same. Minimizing such a risk would lead to a classifier that is robust to additive noise bounded within an ℓ_2 ball. This is exactly the procedure that has been widely adopted in *randomized smoothing* literature in order to obtain a base neural network for which a certifiable smooth classifier can be obtained (Lecuyer et al., 2019; Cohen et al., 2019). Below we discuss another highly effective use case of VRM called *Mixup* which is also the main focus of this work.

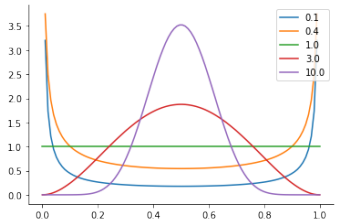

Figure 1: $\texttt{Beta}(\alpha,\alpha)$ pdf for varying α.

Mixup The vicinal distribution in Mixup is defined as

$$P_{\mathbf{x}_i,\mathbf{y}_i}(\mathbf{x},\mathbf{y}) = \mathbb{E}_\lambda[(\delta_{\bar{\mathbf{x}}_i}(\mathbf{x}), \delta_{\bar{\mathbf{y}}_i}(\mathbf{y}))],$$

where $\lambda \sim \texttt{Beta}(\alpha,\alpha)$, $\alpha \in (0,\infty)$, $\bar{\mathbf{x}}_i = \lambda\mathbf{x}_i + (1-\lambda)\mathbf{x}_j$ and $\bar{\mathbf{y}}_i = \lambda\mathbf{y}_i + (1-\lambda)\mathbf{y}_j$. Note that the vicinal distribution in this case not only depends on $(\mathbf{x}_i,\mathbf{y}_i)$ but also on another input-output pair $(\mathbf{x}_j,\mathbf{y}_j)$ drawn from the same training dataset. For a fixed α, implementing Mixup would require taking multiple Monte Carlo samples[2] for each datapoint (refer Eq. (2)) which can be computationally prohibitive. Therefore, in practice, only one interpolation ($m = 1$) per pair of samples from a batch is considered at a time. This procedure, though might look like a crude approximation to the original objective, has resulted in highly promising results in a variety of applications and is very well accepted in the research community. Without undermining the remarkable effectiveness of such a simple approach, we would like to bring attention on two of its potential limitations:

- **Small cross-validated $\alpha \ll 1$:** The shape of the vicinal distribution depends on the hyper-parameter of the Beta distribution, therefore, the values of α will decide the strength of the interpolation factor λ. Hence, how far the interpolated $(\bar{\mathbf{x}},\bar{\mathbf{y}})$ is from the true $(\mathbf{x},\mathbf{y})$ is something that is decided based on the cross-validation performance on a held out test dataset. Since high values of α would encourage $\lambda \approx 0.5$ resulting in $\bar{\mathbf{x}}$ that is very different from $\mathbf{x}$ (hence a mismatch between train and test distributions), the cross-validated value of α for Mixup always turns out to be very small ($\alpha \approx 0.1$) in order to provide good generalization. Note, $\alpha \approx 0.1$ leads to very sharp peaks at 0 and 1 (refer Figure 1). Therefore, effectively, Mixup ends up *slightly* perturbing a clean sample in the direction of another sample even though the vicinal distribution has the potential to explore a much larger portion of the interpolation space.
- **High-entropy behaviour:** Since $m = 1$ in practice (per batch), it is very unlikely that the interpolation factor $\lambda \in \{0,1\}$ even for small values of α. Thus, the model is never exposed to uninterpolated samples during training and hence it always learns to predict interpolated (or smoothed) labels $\bar{\mathbf{y}}$ for every input. Just like DNNs with cross-entropy loss are known to be overconfident because of their high capacity and target Delta distribution Guo et al. (2017), DNNs with Mixup turns out to be *relatively less confident* because the network retains its high capacity but observes only target smoothed labels. This underconfident behaviour results in high-entropy for both in-distribution and out-of-distribution samples. This is undesirable as it will not allow predictive uncertainty to reliably differentiate in-distribution samples from out-of-distribution ones, thus, leading to poor robustness.

We validate the above limitation of Mixup with a simple experiment. In Figure 3, we provide heat-maps that show how the entropy of the predictive distribution (softmax output) varies when interpolating between samples belonging to different classes. The heat-map is created as follows. We train a WideResNet28-10 (WRN) Zagoruyko & Komodakis (2016) using CIFAR-10 (C10) dataset. Then, we randomly choose 1K pairs of samples $\{\mathbf{x}_i,\mathbf{x}_j\}$ from the dataset such that $y_i \neq y_j$[3].

[2]We do not discuss how m would depend on α, however, it is clear that there exists a relation. For example, for small α, Beta distribution will have peaks at the extremes, therefore, relatively smaller m should be sufficient when compared to moderate values of α.

[3]Note, the probability that $\mathbf{y}_i \neq \mathbf{y}_j$ is very high given that we have many classes in the problem. Therefore, it is highly likely that the samples drawn will belong to different classes.

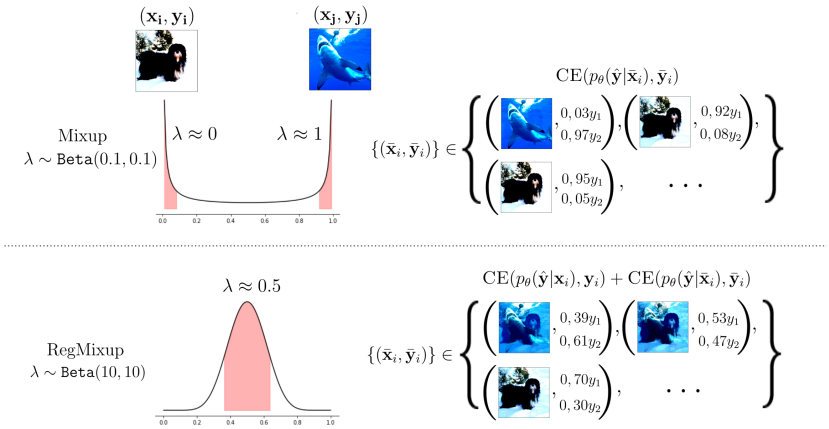

Figure 2: **Mixup vs RegMixup in practice**. Illustration of how the cross-validated α affects the shape of the `Beta` in both cases. Red regions represent 80% of the probability mass. Mixup typically samples $\lambda \approx 0$ or 1, while for RegMixup $\lambda \approx 0.5$. In the first case, one of the two interpolating images dominates the interpolated one; in the latter case, a wide variety of images are obtained.

For each pair, we synthesize 20 samples $\bar{\mathbf{x}}$s using equally spaced λs between 0 to 1. The heat-map is then created using all the 20K samples. The intensity of each (λ, H) bin in the heat-map indicates the number of samples in that bin. Note, DNN (trained with vanilla cross-entropy loss) shows low entropy (overconfidence) irrespective of where the interpolated sample lies. However, Mixup shows high entropy almost always (undercofindence). As shown in Table 1, though Mixup provides improved accuracy compared to DNN for in-distribution and covariate shift experiments, this high entropy behaviour makes it much worse than DNN when exposed to out-of-distribution detection task. For example, when SVHN is used as the OOD dataset, the performance of Mixup drops by nearly 8.47% compared to DNN. This clearly shows that the predictive entropy of Mixup is not discriminative enough. However, there is a clear improvement of nearly 5% for covariate shift experiments, implying, the Mixup augmentations do improve robustness in this aspect. Note that, in the context of calibration, Mixup's underconfident behaviour (which is equivalent to providing high predictive entropy) was also noticed by Wen et al. (2021).

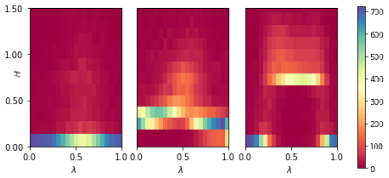

Figure 3: **Heatmaps of the entropy profiles** as the interpolation factor λ between samples of two classes varies. Left (**DNN**), Middle (**Mixup**), Right (**RegMixup**). RegMixup provides high entropy barrier separating in-distribution from out-of-distribution samples.

RegMixup We now provide a very simple yet effective change to the way Mixup has been used in the literature that not just avoids the aforementioned limitations, but also improves in-distribution and covariate-shift accuracies of Mixup. We use the following approximation to the data-distribution

$$P(\mathbf{x}, \mathbf{y}) = \frac{1}{2n} \sum_{i=1}^{n} (\delta_{\mathbf{x}_i}(\mathbf{x}) \delta_{\mathbf{y}_i}(\mathbf{y}) + P_{\mathbf{x}_i, \mathbf{y}_i}(\mathbf{x}, \mathbf{y})),$$

where $P_{\mathbf{x}_i, \mathbf{y}_i}(\mathbf{x}, \mathbf{y})$ is the mixup vicinal distribution. The above approximation is simply an explicit assemble of ERM and VRM based approximate distributions (one could trivially use different mixture weights as well). Though, in theory, VRM subsumes ERM, however, we argue that because of the compute limitations in realizing its full capacity, explicitly combining them might result in a *practically* more expressive models. We provide extensive experimental evidence to support this hypothesis. Implementation wise, for each sample $\mathbf{x}_i$ in a batch, another sample $\mathbf{x}_j$ is drawn from the same batch to obtain interpolated $\bar{\mathbf{x}}_i$ and $\bar{\mathbf{y}}_i$, and then the following loss is minimized

$$\mathrm{CE}(p_\theta(\hat{\mathbf{y}}|\mathbf{x}_i), \mathbf{y}_i) + \mathrm{CE}(p_\theta(\hat{\mathbf{y}}|\bar{\mathbf{x}}_i), \bar{\mathbf{y}}_i), \tag{3}$$

where CE(.,.) denotes the standard cross-entropy loss and $p(.)$ the softmax output of a neural network parameterized by θ.

What does this simple modification bring to the model? (1) A better empirical approximation to the underlying vicinal data distribution. This is because of the fact that irrespective of the values of m and α, the model will always be exposed to the clean training samples as well. (2) The interpolation factor λ can now explore a much wider space without impacting the generalization performance as train/test distribution shifts will not be a problem anymore. Therefore, if $\alpha \ll 1$ was actually the optimal solution, the cross validation would automatically find it.

How does this simple modification impact the behaviour and the performance of the model in practice?

- **Large cross-validated** $\alpha \gg 1$: The model is now able to explore strong interpolations because of the additional cross-entropy term over the unperturbed training data. Interestingly, the cross-validated α that we obtained in fact is very high ($\alpha = 10$) leading to $\lambda \approx 0.5$. Therefore, as opposed to the standard Mixup, RegMixup prefers having strong diverse interpolations during training. Refer Figure 2 for visualizations. Clearly, interpolated samples in RegMixup are diverse and contain features from both the images in the pair.
- **Well behaved entropy (knowingly unknowing the unknown):** It is straightforward to notice that a value of $\lambda \approx 0.5$ would lead to $\bar{\mathbf{x}}$ that is a heavy mix of two samples (mimicking OOD samples, refer Fig. 2), and the corresponding target label vector $\bar{\mathbf{y}}$ would have almost equal masses corresponding to the labels of the interpolating samples. Therefore, minimizing $\mathrm{CE}(p_\theta(\mathbf{y}|\bar{\mathbf{x}}_i), \bar{\mathbf{y}}_i)$ in this case is equivalent to maximizing a soft proxy to entropy defined over the label support of $\mathbf{y}_i$ and $\mathbf{y}_j$ (exact entropy maximization would encourage equal probability masses of 0.5 for these labels). We find this observation intriguing as RegMixup naturally obtains a cross-validated α that leads to a maximum likelihood solution which is uncertain outside the data distribution. This is an extremely desirable property as it allows models to differentiate between in and out distribution samples. Thus, Mixup automatically acts as a **regularizer** in this case.

The entropy heat-map in Figure 3 clearly shows that as opposed to DNN and Mixup, the entropy for RegMixup is very low for λ close to either 0 or 1, however, it increases and remains high for all other intermediate interpolation factors, *practically creating an entropy barrier*. Also, as shown in Table 1, RegMixup provides improvement in accuracy (both in-distribution and covariate-shift) and out-of-distribution robustness. An interesting observation is that Mixup's performance on OOD (T-ImageNet) dropped by 2.4% compared to DNN whereas RegMixup performed 3.75% better than DNN. Thus, RegMixup effectively improved the effectiveness of Mixup by nearly 6.15%.

		Cov. Shift	OOD Detection		
Models	**C10** (Test) **Accuracy** (↑)	**C10-C Accuracy** (↑)	**C100 AUROC** (↑)	**SVHN AUROC** (↑)	**T-ImageNet AUROC** (↑)
DNN	96.14	76.60	88.61	96.00	86.44
Mixup	97.01	81.68	83.17	87.53	84.02
RegMixup (Our)	**97.46**	**83.13**	**89.63**	**96.72**	**90.19**

Table 1: In-distribution, covariate shift, and out-of-distribution detection experiments using WideResNet28-10 trained on C10. C10-C represents the corrupted version of CIFAR-10.

3 Related Works

Covariate-Shift Robustness It is well known that neural networks performance can severely degrade when exposed to covariate shift (Taori et al., 2016; Hendrycks & Dietterich, 2019). While several techniques assume the availability of sets of samples presenting such variations at training time (Wang et al., 2021), other techniques suggest leveraging augmentations to mimic a set of variations that would allow the model to learn features that generalize better (Hendrycks et al., 2020b,a) or self-supervision (Hendrycks et al., 2019a).

Out-Of-Distribution Detection While some authors suggest introducing an OOD reject option (Geifman & El-Yaniv, 2019), the most recent literature tends to focus on leveraging a threshold classifier on an uncertainty measure computed on top of the predictive distribution of a network. For the latter approach, it is therefore convenient for a model to produce high uncertainty predictions when facing OOD samples, and low uncertainty predictions when fed IND samples (Liu et al., 2020a). Within this family of approaches, the most popular ones are the expensive Deep Ensembles (Lakshminarayanan et al., 2017). There are several efficient variants of DE as well (Havasi et al., 2021;

Wen et al., 2020; Huang et al., 2017). Great attention is paid to Bayesian techniques as well (Chen et al., 2014; Zhang et al., 2020; Durmus et al., 2016) among which we focus particularly on SNGP (Liu et al., 2020a), and KFAC-LLLA (Kristiadi et al., 2020). Recently, techniques based on single deterministic networks have also been proposed (van Amersfoort et al., 2020).

Model calibration Modern NNs have been shown to be miscalibrated (Guo et al., 2017), i.e. they exhibit a mismatch between the model's confidence and its accuracy. While recent approaches have been proposed to fix this issue on in-distribution data (Guo et al., 2017; Mukhoti et al., 2020; Lakshminarayanan et al., 2017), it was shown that such calibrated models do not necessarily behave as well under domain-shift (Ovadia et al., 2019).

4 Experiments

Datasets and Network Architectures We employ the widely used WideResNet28-10 (WRN) (Zagoruyko & Komodakis, 2016) and ResNet50 (RN50) (He et al., 2016) architectures. We train them on CIFAR-10 (C10) and CIFAR-100 (C100) datasets. We employ RN50 to perform experiments on ImageNet-1K (Deng et al., 2009) dataset. We report the average of all the metrics computed on 5 seeds. For further details about the code base and the hyperparameters, refer to Appendix A.

For **Covariate Shift (CS)** experiments on models trained on C10 and C100, we resort to the widely used CIFAR10-C (C10-C) and CIFAR100-C (C100-C), corrupted versions of C10 and C100 (Hendrycks & Dietterich, 2019). These datasets are made by applying 15 synthetically generated but *realistic corruptions* at 5 degrees of intensity on the test sets of C10 and C100, respectively. For CIFAR-10, we also use the CIFAR-10.1 (C10.1) (Recht et al., 2018) and CIFAR-10.2 (C10.2) (Lu et al.) datasets designed to test the generalization of CIFAR-10 classifiers to *natural covariate shifts*. To the best of our knowledge there is no such analogous dataset for CIFAR-100. For ImageNet-1K experiments, we use the widely considered ImageNet-A (A) (Hendrycks et al., 2019b), ImageNet-R (R) (Hendrycks et al., 2021), ImageNetv2 (V2) (Recht et al., 2019), and ImageNet-Sketch (SK) (Wang et al., 2019) for covariate shift experiments.

For **Out-of-Distribution (OOD)** detection, following SNGP (Liu et al., 2020a), we use C100 and SVHN Netzer et al. (2011) as OOD for models trained on C10. Similarly, for models trained on C100, we use C10 and SVHN as OOD. Additionally, we also consider the Tiny-ImageNet (T-ImageNet) dataset (Le & Yang, 2015) as OOD set in both the cases (filtering out common classes). For models trained on ImageNet-1K, we use ImageNet-O (O) (Hendrycks et al., 2019b) as the OOD dataset.

Methods considered for comparisons Besides the natural comparison of our method with Mixup (Zhang et al., 2018) and networks trained via ERM on the standard cross-entropy loss (which we will refer to as DNN), we consider several other methods from the OOD detection and CS literature. For models trained on C10 and C100, we consider:

- DNN-SN and DNN-SRN: taking inspiration from (Liu et al., 2020a), we consider DNN models trained with Spectral Normalization (SN) (Miyato et al., 2018a) and Stable Rank Normalization (SRN) (Sanyal et al., 2020) to control the bi-Lipschitz constant of the networks, which has been shown to affect the generalization properties of neural networks.
- SNGP: Spectrally Normalized Gaussian Process (Liu et al., 2020a).
- DUQ: Deterministic Uncertainty Quantification (van Amersfoort et al., 2020).
- KFAC-LLLA: KFAC-Laplace Last Layer Approximation (Kristiadi et al., 2020) makes a model Bayesian at test time by taking the Laplace approximation of the last layer Ritter et al. (2018). We provide a simple outline of this approach in Appendix D.
- AugMix (Hendrycks et al., 2020b): A data augmentation technique that applies randomized augmentations to an input while enforcing them to be consistent during training. With the recommended hyperparameters in the paper, it is almost $4\times$ slower than DNN during training while having the same inference requirements.
- DE: Deep Ensembles (Lakshminarayanan et al., 2017) with 5 members, requiring $5\times$ more compute than most single-model approaches such as DNN.

We would like to mention that our approach (RegMixup) is almost $1.5\times$ slower and Mixup is about $1.2\times$ slower than DNN training while having the same inference requirements. Due to the high computational requirements to train on ImageNet-1K, for ImageNet-1K experiments we consider

DNN, Mixup and the two other strongest baselines: AugMix and DE. We also cross-validate the hyperparameters on a 10% split of the test set, which is removed at test time.

Few missing experiments: Below we provide extensive experiments for proper benchmarking. The number of datasets and architectures we use lead to many combinations, a few of which we were not able to produce good results for (even after extensive hyperparameter search), hence, some of the entries in the tables are missing. For example, we could not make DUQ work on C100 as it exhibited unstable behaviour. We could not produce promising results for SNGP on RN50 CIFAR experiments using their official implementation. Similarly for AugMix RN50 experiments on C10 and C100. We chose not to report these suboptimal numbers. Further details can be found in Appendix A.

Table entries: Bold represents the best among all the single-model approaches, and underline represents the best among all including the expensive Deep Ensembles.

Note, we do not consider methodologies requiring access to an external dataset during training (either for CS or OOD) as not only this would be an unfair comparison[4], but we believe assuming such knowledge is against the goal of this work, which is to develop models robust to unknown scenarios.

4.1 RegMixup improves accuracy on IND and test samples

IND Small-scale CIFAR experiments In Table 2, we compare the accuracy of various approaches on the in-distribution test sets of C10 and C100, respectively. Clearly,

- RegMixup outperforms Mixup in all these experiments. In fact, RegMixup is the best performing one among all the single-model approaches.
- These improvements are non-trivial. For instance, on WRN trained on C100, it outperforms DNN and Mixup by 1.67% and 0.65%. It outperforms SNGP with a significant margin of 4.05%.

	IND Accuracy			
	WRN28-10		RN50	
Methods	C10 (Test) Accuracy (↑)	C100 (Test) Accuracy (↑)	C10 (Test) Accuracy (↑)	C100 (Test) Accuracy (↑)
DNN	96.14	81.58	95.19	79.19
Mixup	97.01	82.60	96.05	80.12
RegMixup (**our**)	**<u>97.46</u>**	**83.25**	**<u>96.71</u>**	**81.52**
DNN-SN	96.22	81.60	95.20	79.27
DNN-SRN	96.22	81.38	95.39	78.96
SNGP	95.98	79.20	-	-
DUQ	94.7	-	-	-
KFAC-LLLA	96.11	81.56	95.21	79.41
Augmix	96.40	81.10	-	-
DE (5×)	96.75	<u>83.85</u>	96.23	<u>82.09</u>

Table 2: Accuracies (%) on IND samples for models trained on C10 and C100

Note how the single-model approaches specifically designed to provide reliable predictive uncertainty estimations (for example, SNGP) underperform even compared to the vanilla DNN in terms of IND accuracy. In order to provide improved uncertainty estimates (as we will soon show), they trade clean data accuracy. This type of behaviour is not observed in RegMixup.

Covariate Shift Small-scale CIFAR experiments For C10-C and C100-C, as typical in the literature, we report the accuracy averaged over all the corruptions and degrees of intensities in Table 3. It is evident that our approach produces a remarkable improvement in the average accuracy compared to **all** the baselines (except AugMix, we discuss later why that might be the case). For instance, for C100-C experiments, our method achieves an accuracy improvement of **6.9%** over DNN, of **3.86%** over DE, and of **2.45%** over Mixup. Similarly, for C10-C WRN, our method achieves an improvement of almost **6.53%** over DNN, of **4.81%** over DE, and of **1.45%** over Mixup.

For natural covariate shift datasets C10.1 and C10.2 as well, RegMixup outperforms **all** the baselines (including AugMix). For instance, on C10.2, it obtains an improvement of **3.26%** over DNN, of **1.5%** over Mixup, and of **2.46%** over the expensive DE.

IND Large-scale ImageNet-1K experiments RegMixup scales to ImageNet-1K and exhibits improved accuracy with respect to both Mixup, DNN, and AugMix. In particular, it is **1.08%** better than DNN, **0.53%** better than Mixup and **0.80%** better than AugMix. DE in this case is the best performing one.

[4]since all the methods we consider do not leverage this information. RegMixup only relies on in-distribution training data, just like other approaches, and makes no assumption on the type of CS nor on the OOD inputs.

	WRN28-10						RN50					
	CIFAR10 (In-Distribution)			CIFAR100 (In-Distribution)			CIFAR10 (In-Distribution)			CIFAR100 (In-Distribution)		
Out-of-Distribution	**C100**	**SVHN**	**T-ImageNet**	**C10**	**SVHN**	**T-ImageNet**	**C100**	**SVHN**	**T-ImageNet**	**C10**	**SVHN**	**T-ImageNet**
Methods	**AUROC** (↑)			**AUROC** (↑)			**AUROC** (↑)			**AUROC** (↑)		
DNN	88.61	96.00	86.44	81.06	79.68	80.99	88.61	93.20	87.82	79.33	82.45	79.89
Mixup	83.17	87.53	84.02	78.37	78.68	80.61	84.24	89.40	84.89	77.02	76.86	80.14
RegMixup (**our**)	89.63	**96.72**	**90.19**	**81.27**	**89.32**	**83.13**	**89.63**	**95.39**	**90.04**	**79.44**	**88.66**	**82.56**
DNN-SN	88.56	95.59	87.71	81.10	83.43	82.26	88.19	93.46	87.55	79.20	80.78	79.90
DNN-SRN	88.46	96.12	87.43	81.26	85.51	82.41	88.82	93.54	87.82	78.77	82.39	79.70
SNGP	**90.61**	95.25	90.01	79.05	86.78	82.60	-	-	-	-	-	-
KFAC-LLLA	89.33	94.17	87.81	81.04	80.32	81.57	89.54	93.13	88.32	79.30	82.80	80.17
Aug-Mix	89.78	91.3	88.99	81.10	76.64	80.56	-	-	-	-	-	-
DE (5×)	91.25	97.53	89.52	83.26	85.07	83.40	91.38	96.90	90.5	81.93	85.08	82.15

Table 5: Out-of-distribution detection results (%) for WideResNet28-10 and ResNet50 for models trained on C10 and C100. See Appendix A for the cross-validated hyperparameters.

Covariate Shift Large-scale ImageNet-1K experiments In Table 4 we report the results for common ImageNet-1K covariate-shift test sets. As it can be seen, RegMixup is either the best performing among all the single-model approaches, or it is the absolute winner. For instance, on ImageNet-A RegMixup performs **2.67%** better than Mixup and **3.20%** better than DNN. Similarly, on ImageNet-V2 it performs **1.08%** better than Mixup and **1.13%** more than DNN. RegMixup also outperforms AugMix on all the considered datasets, while is outperformed by DE on ImageNet-V2 (by **1.02%**) and performs competitively on ImageNet-SK.

These experiments show the strong generalization performance of RegMixup under various CS scenarios. They also show that it does not trade clean data accuracy to do so.

	Covariate Shift Accuracy							
	WRN28-10				R50			
Methods	**C10-C**	**C10.1**	**C10.2**	**C100-C**	**C10-C**	**C10.1**	**C10.2**	**C100-C**
	Accuracy (↑)				**Accuracy** (↑)			
DNN	76.60	90.73	84.79	52.54	75.18	88.58	83.31	50.62
Mixup	81.68	91.29	86.55	56.99	78.63	90.03	84.61	53.96
RegMixup (**our**)	83.13	**92.79**	**88.05**	59.44	**81.18**	**91.58**	**86.72**	**57.64**
DNN-SN	76.56	91.01	84.72	52.61	74.88	88.26	82.96	50.55
DNN-SRN	77.21	90.88	85.24	52.54	75.40	88.61	83.49	50.48
SNGP	78.37	90.80	84.95	57.23	-	-	-	-
DUQ	71.6	-	-	50.4	-	-	-	-
KFAC-LLLA	76.56	90.68	84.68	52.57	75.18	88.34	83.50	50.85
AugMix	**90.02**	91.6	85.9	**68.15**	-	-	-	-
DE (×5)	78.32	92.17	85.59	55.58	77.63	90.05	85.00	53.91

Table 3: Accuracies (%) on covariate shifted samples for models trained on C10 and C100.

	IND Acc.	Covariate Shift Acc				OOD Det.
	ImageNet-1K	R	A	V2	Sk	O
	Acc (↑)	**Acc** (↑)	**Acc** (↑)	**Acc** (↑)	**Acc** (↑)	**AUROC** (↑)
DNN	76.60	36.41	2.76	64.53	24.72	55.97
Mixup	77.15	39.05	3.29	64.58	26.34	55.54
RegMixup (**our**)	**77.68**	**39.76**	**5.96**	**65.66**	**26.98**	**57.05**
AugMix	76.88	38.29	2.63	64.94	25.61	56.91
DE (5×)	78.22	38.94	2.11	66.68	27.03	53.29

Table 4: ImageNet accuracies (%) on IND and CS samples, and OOD detection performance.

Why AugMix performs extraordinarily well on synthetically corrupted C10-C and C100-C but not on natural distribution shift C10.1 and C10.2? Looking at Table 3 one can observe AugMix's extremely good performance on the C10-C and C100-C. However, at the same time, the model is underperforming with respect to RegMixup on C10.1 and C10.2. Similarly, AugMix is outperformed by RegMixup on all ImageNet CS experiments (and OOD as will be shown soon). This seems to suggest that although the augmentations used during the training of AugMix are not exactly same as that of the corrupted test dataset, they tend to benefit more synthetic forms of covariate shift, hence the dramatic improvement in these particular scenarios.

4.2 Out-of-Distribution detection experiments

Following the standard procedure (Liu et al., 2020a), we report the performance in terms of AUROC[5] for the binary classification problem between in- and out-distribution samples. The predictive uncertainty of the model is typically used to obtain these curves. Given an uncertainty measure (normally entropy, refer Appendix B for an extensive discussion), it is important for models to be more uncertain on OOD samples than on in-distribution samples to be able to distinguish them accurately. This behaviour leads to better AUROC.

We report the OOD detection results for models trained on C10 and C100 in Table 5. In the case of SVHN as the OOD dataset, our method either outperforms all the existing approaches (2.54% higher AUROC than the runner-up SNGP in C100 experiments) or it turns out to be a runner-up with the gap of 0.81% compared to the top performing and expensive DE for C10 experiments.

[5] Area Under Receiver Operating Characteristic curve.

In the case of CIFAR as the OOD dataset, DE turns out to be the best performing one. RegMixup, along with KFAC-LLLA, is the runner-up in the case of C100 experiments while SNGP is the runner-up in C10 experiments. In the case of Tiny-ImageNet as the OOD dataset, our method outperforms all single-network models and DE when the in-distribution set is C10, but it is slightly inferior to DE when the in-distribution set is C100. Similarly, on ImageNet-O detection, RegMixup outperforms all the baselines.

4.3 Calibration on In-Domain and Covariate Shifted Inputs

Additionally, we provide the calibration performance of various competitive approaches. Briefly, calibration quantifies how similar a model's confidence and its accuracy are (Osborne, 1991)). To measure it, we employ the recently proposed Adaptive ECE (AdaECE) (Mukhoti et al., 2020). For all the methods, the AdaECE is computed after performing temperature scaling (Guo et al., 2017) with a cross-validated temperature parameter. We also provide the AdaECE without temperature scaling in Appendix C.

	IND			
	WRN28-10		**RN50**	
Methods	**C10 (Test)**	**C100 (Test)**	**C10 (Test)**	**C100 (Test)**
	AdaECE (↓)		**AdaECE (↓)**	
DNN	1.34	3.84	1.45	2.94
Mixup	1.16	1.98	2.17	7.47
RegMixup (**our**)	**0.50**	**1.76**	**0.94**	**1.53**
SNGP	0.87	1.94	-	-
Augmix	1.67	5.54	-	-
DE (5×)	1.04	3.29	1.28	2.98

Table 6: CIFAR IND calibration performance (%).

	IND	**Covariate Shift**			
	ImageNet-1K (Test)	**ImageNet-R**	**ImageNet-A**	**ImageNet-V2**	**ImageNet-Sk**
	AdaECE (↓)	**AdaECE (↓)**	**AdaECE (↓)**	**AdaECE (↓)**	**AdaECE (↓)**
DNN	1.81	13.56	44.90	4.13	14.48
Mixup	**1.29**	12.08	44.63	4.28	15.26
RegMixup (**our**)	1.37	13.30	**41.18**	**3.38**	15.35
AugMix	2.05	**11.24**	42.83	3.94	**14.26**
DE (5×)	1.38	13.55	42.88	4.02	17.32

Table 7: ImageNet calibration performance on IND and CS datasets (%).

In terms of calibration on in-domain test sets (refer Tables 6 and 7), our method either remarkably improves the AdaECE with respect to Mixup and DNN or performs competitively (on ImageNet-1K).

Under covariate shift (refer Tables 7 and 8), on corrupted inputs, RegMixup underperforms with respect to Mixup on C10-C, but not on C100-C. On all other C10 covariate shift datasets RegMixup outperforms Mixup and DNN. Considering also the other baselines, except for the case of C10-C (in which AugMix significantly outperforms any other baseline on WRN28-10), our method provides the best calibration in all other cases. For example, on C100-C experiments on WRN28-10, in terms of AdaECE, RegMixup obtains a **4.43%** improvement over DE, **2.52%** over Mixup, and **2.47%** over SNGP. Though RegMixup outperformed all other approaches in 12 scenarios out of total 17 scenarios presented here, it is clear there is no single method that outperforms any other in all considered settings.

	Covariate Shift							
	WRN28-10				**R50**			
Methods	**C10-C**	**C10.1**	**C10.2**	**C100-C**	**C10-C**	**C10.1**	**C10.2**	**C100-C**
	AdaECE (↓)				**AdaECE (↓)**			
DNN	12.62	4.13	8.81	9.94	12.29	4.36	8.89	19.76
Mixup	7.93	4.39	7.44	10.45	**10.75**	5.72	10.59	12.63
RegMixup (**our**)	9.08	**2.57**	**6.83**	**7.93**	11.37	**2.89**	**6.74**	**11.47**
SNGP	11.34	4.36	8.33	10.43	-	-	-	-
AugMix	**4.56**	3.23	8.33	12.15	-	-	-	-
DE (5 ×)	10.31	2.60	7.50	12.36	12.68	4.10	6.94	12.36

Table 8: CIFAR CS calibration performance (%).

5 Conclusion

We proposed RegMixup, an extremely simple approach that combines Mixup with the standard cross-entropy loss over unperturbed training data. We conducted a wide range of experiments and showed that RegMixup significantly improved the reliability of uncertainty estimates of deep neural networks, while also provided a notable boost in the accuracy. We showed that RegMixup did not just outperform Mixup, it also outperformed most recent state-of-the-art approaches in providing reliable uncertainty estimates while improving the in-distribution accuracy. A potential extension of our work would be to explore more situations where ERM and VRM can be combined for practical benefits.

References

Behrmann, J., Grathwohl, W., Chen, R. T. Q., Duvenaud, D., and Jacobsen, J.-H. Invertible residual networks. In *ICML*, November 2019.

Bishop, C. M. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag, Berlin, Heidelberg, 2006. ISBN 0387310738.

Chapelle, O., Weston, J., Bottou, L., and Vapnik, V. Vicinal risk minimization. In *Proceedings of the 13th International Conference on Neural Information Processing Systems*, NIPS'00, pp. 395–401, Cambridge, MA, USA, 2000. MIT Press.

Chen, T., Fox, E., and Guestrin, C. Stochastic gradient hamiltonian monte carlo. In Xing, E. P. and Jebara, T. (eds.), *Proceedings of the 31st International Conference on Machine Learning*, volume 32 of *Proceedings of Machine Learning Research*, pp. 1683–1691, Bejing, China, 22–24 Jun 2014. PMLR. URL http://proceedings.mlr.press/v32/cheni14.html.

Cohen, J., Rosenfeld, E., and Kolter, Z. Certified adversarial robustness via randomized smoothing. In *International Conference on Machine Learning*, pp. 1310–1320. PMLR, 2019.

Dangel, F., Kunstner, F., and Hennig, P. Backpack: Packing more into backprop. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=BJlrF24twB.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 248–255, 2009. doi: 10.1109/CVPR.2009.5206848.

Durmus, A., Simsekli, U., Moulines, E., Badeau, R., and RICHARD, G. Stochastic gradient richardson-romberg markov chain monte carlo. In Lee, D., Sugiyama, M., Luxburg, U., Guyon, I., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 29, pp. 2047–2055. Curran Associates, Inc., 2016. URL https://proceedings.neurips.cc/paper/2016/file/03f5446139179452450 41ea1581df0c2-Paper.pdf.

Gal, Y. and Ghahramani, Z. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, ICML'16, pp. 1050–1059. JMLR.org, 2016.

Geifman, Y. and El-Yaniv, R. Selectivenet: A deep neural network with an integrated reject option. In *International Conference on Machine Learning*, pp. 2151–2159. PMLR, 2019.

Guo, C., Pleiss, G., Sun, Y., and Weinberger, K. Q. On calibration of modern neural networks. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, pp. 1321–1330. JMLR.org, 2017.

Havasi, M., Jenatton, R., Fort, S., Liu, J. Z., Snoek, J., Lakshminarayanan, B., Dai, A. M., and Tran, D. Training independent subnetworks for robust prediction. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=OGg9XnKxFAH.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, 2016. doi: 10.1109/CVPR.2016.90.

Hendrycks, D. and Dietterich, T. Benchmarking neural network robustness to common corruptions and perturbations. *Proceedings of the International Conference on Learning Representations*, 2019.

Hendrycks, D., Mazeika, M., Kadavath, S., and Song, D. Using self-supervised learning can improve model robustness and uncertainty. In Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019a. URL https://proceedings.neurips.cc/paper/2019/file/a2b15837edac15df90721968986f7f8e-Paper.pdf.

Hendrycks, D., Zhao, K., Basart, S., Steinhardt, J., and Song, D. Natural adversarial examples. July 2019b.

Hendrycks, D., Basart, S., Mu, N., Kadavath, S., and others. The many faces of robustness: A critical analysis of out-of-distribution generalization. *arXiv preprint arXiv*, 2020a.

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. *Proceedings of the International Conference on Learning Representations (ICLR)*, 2020b.

Hendrycks, D., Basart, S., Mu, N., Kadavath, S., Wang, F., Dorundo, E., Desai, R., Zhu, T., Parajuli, S., Guo, M., Song, D., Steinhardt, J., and Gilmer, J. The many faces of robustness: A critical analysis of out-of-distribution generalization. *ICCV*, 2021.

Hobbhahn, M., Kristiadi, A., and Hennig, P. Fast predictive uncertainty for classification with bayesian deep networks, 2021. URL https://openreview.net/forum?id=KcImcc3j-qS.

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

Kingma, D. P., Salimans, T., and Welling, M. Variational dropout and the local reparameterization trick. In Cortes, C., Lawrence, N., Lee, D., Sugiyama, M., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 28, pp. 2575–2583. Curran Associates, Inc., 2015. URL https://proceedings.neurips.cc/paper/2015/file/bc7316929fe1545bf0b98d114ee3ecb8-Paper.pdf.

Kristiadi, A., Hein, M., and Hennig, P. Being bayesian, even just a bit, fixes overconfidence in ReLU networks. In III, H. D. and Singh, A. (eds.), *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 5436–5446. PMLR, 13–18 Jul 2020. URL http://proceedings.mlr.press/v119/kristiadi20a.html.

Lakshminarayanan, B., Pritzel, A., and Blundell, C. Simple and scalable predictive uncertainty estimation using deep ensembles. In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 30, pp. 6402–6413. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper/2017/file/9ef2ed4b7fd2c810847ffa5fa85bce38-Paper.pdf.

Le, Y. and Yang, X. S. Tiny imagenet visual recognition challenge. 2015.

Lecuyer, M., Atlidakis, V., Geambasu, R., Hsu, D., and Jana, S. Certified robustness to adversarial examples with differential privacy. In *IEEE Symposium on Security and Privacy (SP)*, 2019.

Lee, K., Lee, K., Lee, H., and Shin, J. A simple unified framework for detecting out-of-distribution samples and adversarial attacks, 2018.

Liu, J. Z., Lin, Z., Padhy, S., Tran, D., Bedrax-Weiss, T., and Lakshminarayanan, B. Simple and principled uncertainty estimation with deterministic deep learning via distance awareness. In *NeurIPS*, June 2020a.

Liu, W., Wang, X., Owens, J. D., and Li, Y. Energy-based out-of-distribution detection, 2020b.

Lu, S., Nott, . B., and Olson, . A. Harder or different? a closer look at distribution shift in dataset reproduction. http://www.gatsby.ucl.ac.uk/~balaji/udl2020/accepted-papers/UDL2020-paper-101.pdf. Accessed: 2021-11-10.

Lu, Z., Ie, E., and Sha, F. Uncertainty estimation with infinitesimal jackknife, its distribution and mean-field approximation, 2020.

Miyato, T., Kataoka, T., Koyama, M., and Yoshida, Y. Spectral normalization for generative adversarial networks. In *ICLR*, February 2018a.

Miyato, T., Kataoka, T., Koyama, M., and Yoshida, Y. Spectral normalization for generative adversarial networks. In *International Conference on Learning Representations*, 2018b. URL https://openreview.net/forum?id=B1QRgziT-.

Mukhoti, J., Kulharia, V., Sanyal, A., Golodetz, S., Torr, P. H., and Dokania, P. K. Calibrating deep neural networks using focal loss. In *NeurIPS*, 2020.

Netzer, Y., Wang, T., Coates, A., Bissacco, A., Wu, B., and Ng, A. Reading digits in natural images with unsupervised feature learning. 2011.

Osborne, C. Statistical calibration: A review. *International Statistical Review / Revue Internationale de Statistique*, 59(3):309–336, 1991. ISSN 03067734, 17515823. URL http://www.jstor.org/stable/1403690.

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, 2019.

Quionero-Candela, J., Sugiyama, M., Schwaighofer, A., and Lawrence, N. D. *Dataset Shift in Machine Learning*. The MIT Press, 2009. ISBN 0262170051.

Recht, B., Roelofs, R., Schmidt, L., and Shankar, V. Do CIFAR-10 classifiers generalize to CIFAR-10? June 2018.

Recht, B., Roelofs, R., Schmidt, L., and Shankar, V. Do imagenet classifiers generalize to imagenet? *CoRR*, abs/1902.10811, 2019. URL `http://arxiv.org/abs/1902.10811`.

Ritter, H., Botev, A., and Barber, D. A scalable laplace approximation for neural networks. In *International Conference on Learning Representations*, 2018. URL `https://openreview.net/forum?id=Skdvd2xAZ`.

Sanyal, A., Torr, P. H. S., and Dokania, P. K. Stable rank normalization for improved generalization in neural networks and GANs. In *ICLR*, 2020.

Sensoy, M., Kaplan, L., and Kandemir, M. Evidential deep learning to quantify classification uncertainty. In Bengio, S., Wallach, H., Larochelle, H., Grauman, K., Cesa-Bianchi, N., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 31, pp. 3179–3189. Curran Associates, Inc., 2018. URL `https://proceedings.neurips.cc/paper/2018/file/a981f2b708044d6fb4a71a1463242520-Paper.pdf`.

Taori, R., Dave, A., Shankar, V., Carlini, N., Recht, B., and Schmidt, L. Measuring robustness to natural distribution shifts in image classification. July 2016.

Thulasidasan, S., Chennupati, G., Bilmes, J. A., Bhattacharya, T., and Michalak, S. On mixup training: Improved calibration and predictive uncertainty for deep neural networks. In Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 32, pp. 13888–13899. Curran Associates, Inc., 2019. URL `https://proceedings.neurips.cc/paper/2019/file/36ad8b5f42db492827016448975cc22d-Paper.pdf`.

van Amersfoort, J., Smith, L., Teh, Y. W., and Gal, Y. G. Uncertainty estimation using a single deep deterministic neural network. In *ICML*, 2020.

Vapnik, V. Principles of risk minimization for learning theory. In *Proceedings of the 4th International Conference on Neural Information Processing Systems*, NIPS'91, pp. 831–838, San Francisco, CA, USA, 1991. Morgan Kaufmann Publishers Inc. ISBN 1558602224.

Wang, H., Ge, S., Lipton, Z., and Xing, E. P. Learning robust global representations by penalizing local predictive power. In *Advances in Neural Information Processing Systems*, pp. 10506–10518, 2019.

Wang, J., Lan, C., Liu, C., Ouyang, Y., Zeng, W., and Qin, T. Generalizing to unseen domains: A survey on domain generalization. *arXiv preprint arXiv:2103.03097*, 2021.

Wen, Y., Tran, D., and Ba, J. Batchensemble: an alternative approach to efficient ensemble and lifelong learning. In *International Conference on Learning Representations*, 2020. URL `https://openreview.net/forum?id=Sklf1yrYDr`.

Wen, Y., Jerfel, G., Muller, R., Dusenberry, M. W., Snoek, J., Lakshminarayanan, B., and Tran, D. Combining ensembles and data augmentation can harm your calibration. In *International Conference on Learning Representations*, 2021. URL `https://openreview.net/forum?id=g11CZSghXyY`.

Wightman, R. Pytorch image models. `https://github.com/rwightman/pytorch-image-models`, 2019.

Winkens, J., Bunel, R., Roy, A. G., Stanforth, R., Natarajan, V., Ledsam, J. R., MacWilliams, P., Kohli, P., Karthikesalingam, A., Kohl, S., Cemgil, T., Ali Eslami, S. M., and Ronneberger, O. Contrastive training for improved Out-of-Distribution detection. July 2020.

Zagoruyko, S. and Komodakis, N. Wide residual networks. In Richard C. Wilson, E. R. H. and Smith, W. A. P. (eds.), *Proceedings of the British Machine Vision Conference (BMVC)*, pp. 87.1–87.12. BMVA Press, September 2016. ISBN 1-901725-59-6. doi: 10.5244/C.30.87. URL https://dx.doi.org/10.5244/C.30.87.

Zhang, H., Cisse, M., Dauphin, Y. N., and Lopez-Paz, D. mixup: Beyond empirical risk minimization. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=r1Ddp1-Rb.

Zhang, R., Li, C., Zhang, J., Chen, C., and Wilson, A. G. Cyclical stochastic gradient mcmc for bayesian deep learning. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=rkeS1RVtPS.

Checklist

The checklist follows the references. Please read the checklist guidelines carefully for information on how to answer these questions. For each question, change the default **[TODO]** to [Yes] , [No] , or [N/A] . You are strongly encouraged to include a **justification to your answer**, either by referencing the appropriate section of your paper or providing a brief inline description. For example:

- Did you include the license to the code and datasets? [Yes] See Section **??**.
- Did you include the license to the code and datasets? [No] The code and the data are proprietary.
- Did you include the license to the code and datasets? [N/A]

Please do not modify the questions and only use the provided macros for your answers. Note that the Checklist section does not count towards the page limit. In your paper, please delete this instructions block and only keep the Checklist section heading above along with the questions/answers below.

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] See the "Methods considered for Comparison" section
   (c) Did you discuss any potential negative societal impacts of your work? [No] Not expected
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A]
   (b) Did you include complete proofs of all theoretical results? [N/A]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [No] Code will be made available upon acceptance. The appendix contains an explicative pseudocode
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] They are specified either in the Experiments section of the main paper or the Appendix
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [N/A]
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] See Appendix
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]

(b) Did you mention the license of the assets? [No]

(c) Did you include any new assets either in the supplemental material or as a URL? [No]

(d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]

(e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]

5. If you used crowdsourcing or conducted research with human subjects...

(a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]

(b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]

(c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]