PIXCALIBRATE: PIXEL-MIXING GUIDED CALIBRATION AND ROBUSTNESS FOR IMAGE CORRUPTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Reliable image classification requires models to remain accurate and calibrated under distribution shift. However, models trained on clean data are often prone to incorrect and overconfident predictions on corrupted or out-of-distribution test data. Existing augmentation frameworks partially address this issue, but they typically rely on global interpolation or introduce sharp region-level changes, neither of which simulates real-world corruptions. In this work, we propose PixCalibrate, a training-time pixel-level mixing approach for improving both robustness and calibration under corrupted settings. PixCalibrate generates augmented samples that preserve the semantic structure of an anchor image, while injecting a controlled fraction of pixels from a perturbation source—auxiliary image, producing fine-grained, spatially distributed perturbations that closely mimic real-world corruptions without disrupting the overall image structure. PixCalibrate consistently achieves strong accuracy and calibration across standard image corruption benchmarks and outperforms multiple prior augmentation frameworks.

1 INTRODUCTION

Deep neural networks (DNNs) (LeCun et al., 2015; Mathew et al., 2021) achieve impressive results on image classification benchmarks (Zagoruyko & Komodakis, 2016; He et al., 2016), but their success is often measured on clean images. In real-world settings, however, visual inputs are rarely ideal. For example, images captured on consumer devices may contain sensor noise under low light; medical scans can suffer from speckle or motion artifacts; satellite imagery may be degraded by clouds, haze, or compression; and autonomous vehicles must interpret scenes distorted by rain, fog, or occlusion. Since each image passes through an imperfect physical and digital acquisition pipeline, some degree of image corruption is thus unavoidable. As a result, models that perform well on clean benchmarks degrade in deployment (Hendrycks & Dietterich, 2019). Moreover, this degradation extends beyond accuracy: real-world reliability depends also on well-calibrated predictive confidence. For example, a medical diagnostic system may misclassify a lesion due to compression artifacts, or an autonomous vehicle may fail to detect a pedestrian through rain-streaked glass, but such errors can have serious consequences, especially when the model is overconfident. While modern DNNs are shown to be poorly calibrated even on clean distributions (Guo et al., 2017), with confidence scores that overestimate true accuracy, this miscalibration is further amplified under corrupted and perturbed inputs, as illustrated in Figure 1a.

Data augmentation (Shorten & Khoshgoftaar, 2019; Taylor & Nitschke, 2018; Yang et al., 2022; Mikołajczyk & Grochowski, 2018) has been widely used to reduce the mismatch between clean training data and corrupted (or out-of-distribution) real-world inputs. These approaches can be broadly categorized into global interpolation and patch-based augmentation. Global interpolation methods, including Mixup (Zhang et al., 2018) and CalibrateMix (Rahman et al., 2026), improve prediction and calibration (Thulasidasan et al., 2019) by mixing images and labels to create soft training targets. Other augmentation methods utilize external pattern mixing (fractals) (Hendrycks et al., 2022) or patch manipulation, such as Cutout (DeVries & Taylor, 2017), CutMix (Yun et al., 2019), ResizeMix (Qin et al., 2020), and SnapMix (Huang et al., 2021). These approaches have clear limitations. Global interpolation often produces unrealistic images that deviate from natural inputs, while patch-based methods can introduce sharp artificial boundaries or overlook important regions (see Figure 1e). In contrast, many real-world corruptions are typically fine-grained and occur at the pixel level, while preserving the overall structure. As a result, both approaches struggle in corrupted

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

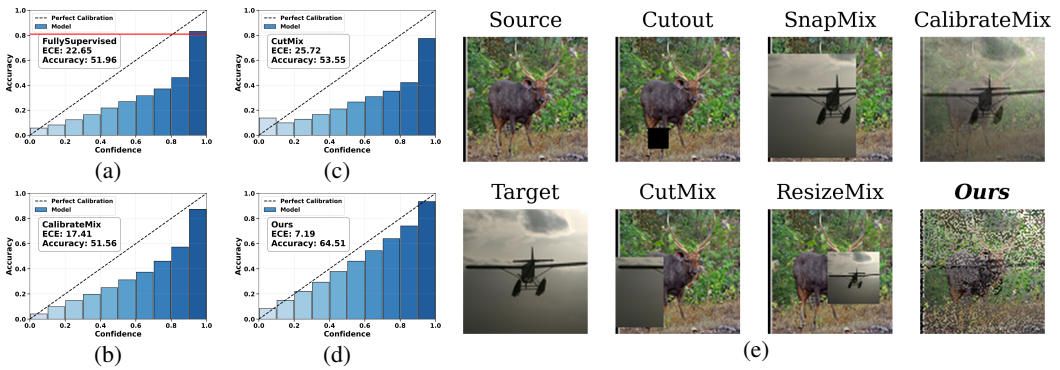


Figure 1: Performance (Expected Calibration Error (ECE) and Accuracy) on CIFAR-100-C for: (a) Fully-Supervised (the red line represents the accuracy of FullySupervised on test clean images of CIFAR-100), (b) CalibrateMix, (c) CutMix, (d) Ours; and (e) Augmentation strategies.

or distribution-shifted settings, often leading to degraded performance and overconfident predictions, as shown in Figures 1b and 1c (for global interpolation and patch-based, respectively).

To address these limitations, we propose PixCalibrate, a pixel-level mixing approach designed to improve both accuracy and calibration under corrupted data. Operating directly on clean training data without external inputs or auxiliary models, PixCalibrate avoids the unrealistic global interpolation of Mixup-style methods and the artificial boundaries of patch-based approaches. Instead, it injects a controlled fraction of pixels from a perturbation source image into an anchor image. This generates fine-grained, spatially distributed perturbations that closely mimic real-world corruptions while fully preserving the anchor's semantic structure. To ensure these augmented samples are informative rather than arbitrarily ambiguous, PixCalibrate employs a lightweight, difficulty-aware sample selection strategy, where sample difficulty is estimated using Area Under the Margin (AUM) (Pleiss et al., 2020), computed from the model's output margins (Bartlett et al., 2017; Elsayed et al., 2018) across training epochs. This integrated design offers three key advantages: First, the diffuse pixel-level mixing better reflects real-world corruptions (as we show in §6.1), directly enhancing robustness and preventing overconfident predictions (Figure 1d). Second, AUM-guided pairing successfully balances semantic stability with necessary uncertainty (Rahman et al., 2026; Park & Caragea, 2022). Finally, the method is highly computationally efficient, requiring only margin tracking and a random mask. We evaluate PixCalibrate on standard corrupted benchmarks (CIFAR-10-C, CIFAR-100-C (Hendrycks & Dietterich, 2019)) alongside our newly introduced SVHN-C and STL-10-C datasets (Appendix A.7). For large-scale experiments, we utilize ImageNet (Deng et al., 2009) along with ImageNet-C for corruption robustness (Hendrycks & Dietterich, 2019), and ImageNet-O and ImageNet-A for out-of-distribution and adversarial generalization (Hendrycks et al., 2021). Across all settings, PixCalibrate consistently improves both robustness and calibration; for example, it reduces Expected Calibration Error (ECE) from 22.18 to 8.88 on CIFAR-100-C and from 22.48 to 15.86 on ImageNet-C, while boosting accuracy from 51.76% to 65.03% and 26.93% to 32.53%, respectively.

Our contributions are as follows:

- We propose **PixCalibrate**, a novel targeted pixel-level mixing approach that enhances both predictive accuracy and calibration of supervised models under distribution shift, particularly in the presence of corrupted and out-of-distribution data.
- Through extensive experiments on different benchmark corrupted datasets, including our newly introduced **SVHN-C** and **STL-10-C** benchmarks, we demonstrate the effectiveness of PixCalibrate in terms of accuracy and calibration metrics in comparison to multiple state-of-the-art algorithms. We will make our code and dataset publicly available.
- We conduct extensive ablation studies to understand the contributions of various components to PixCalibrate's success, and perform error analysis to assess its performance in practice.

2 Related Works

Data Augmentation for Robustness. Data augmentation is a widely used strategy for improving model robustness (Shorten & Khoshgoftaar, 2019; Yang et al., 2022). Mixup (Zhang et al., 2018) trains on convex combinations of random image–label pairs, improving generalization, adversarial

robustness, and as later shown by Zhang et al. (2021) implicitly upper-bounding an adversarial training loss. Related global-mixing variants explore frequency-space masks (Harris et al., 2020), soft boundary blending (Lee et al., 2020), feature-space alignment (Venkataramanan et al., 2022), diversity-guided batch optimization (Kim et al., 2021), and mixing with complex fractal images to simultaneously improve corruption robustness and calibration (Hendrycks et al., 2022). AugMix (Hendrycks et al., 2020) enforces Jensen–Shannon consistency between clean and augmented views, achieving strong results on ImageNet-C/P. Patch-based methods instead operate on contiguous spatial regions: Cutout (DeVries & Taylor, 2017) and Random Erasing (Zhong et al., 2020) mask rectangular regions, while CutMix (Yun et al., 2019) and ResizeMix (Qin et al., 2020) cut and paste content between images with proportional label mixing. More sophisticated variants: Puzzle Mix (Kim et al., 2020), SaliencyMix (Uddin et al., 2021), SuperMix (Dabouei et al., 2021), GridMix (Baek et al., 2021), and TokenMix (Liu et al., 2022) use saliency maps, teacher networks, or token-level activations to guide mixup. These methods sample pairs at random and introduce either global mixup or contiguous-patch perturbations, neither of which captures the fine-grained, spatially distributed nature of real-world corruptions, and neither treats calibration under shift as a first-class objective.

Calibration and Calibration-Aware Augmentation. Guo et al. (2017) established that modern neural networks are systematically overconfident, introducing temperature scaling alongside vector scaling and matrix scaling as post-hoc recalibration baselines, which build upon classic non-parametric post-hoc methods like histogram binning and isotonic regression (Zadrozny & Elkan, 2001; 2002); however, such techniques degrade under dataset shift (Ovadia et al., 2019), motivating training-time alternatives. The confidence penalty (Pereyra et al., 2017) and label smoothing directly regularize output sharpness, while Thulasidasan et al. (2019) showed that Mixup-trained models are substantially better calibrated on Out-Of-Distribution (OOD) and noisy data. Subsequent analyses (Zhang et al., 2022) caution that this benefit depends heavily on interpolation strength α, where overly aggressive mixing can actually degrade calibration. A small but growing line of work treats calibration as an explicit objective of augmentation. RegMixup (Pinto et al., 2022) adds Mixup as an explicit regularizer on top of cross-entropy, improving both in-domain calibration and covariate-shift robustness. Most closely related to our work, CalibrateMix (Rahman et al., 2026) introduces difficulty-aware sample pairing via AUM (Pleiss et al., 2020) to improve calibration in semi-supervised image classification, but operates via global interpolation. PixCalibrate extends this line of work to the fully supervised setting with three complementary directions: (i) *pixel-level* mixing via a random binary mask, distributing perturbations diffusely to better approximate real-world corruptions; (ii) *difficulty-aware pairing* via AUM, balancing semantic stability with informative uncertainty; and (iii) an explicit *confidence penalty* that suppresses overconfidence on both clean and augmented samples, without requiring unlabeled data, auxiliary networks, or saliency signals.

3 METHODOLOGY

3.1 PROBLEM FORMULATION

Let $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{N}$ be a labeled dataset of clean examples. Each input $x \in \mathbb{R}^{H\times W\times C}$ denotes an image, where H, W, C represent height, width, and channels, and $y \in \{1, \ldots, K\}$ is its corresponding class label. The goal is to learn a model f_θ (parameterized by θ) that predicts y given an input x.

Training proceeds for T epochs. At each epoch, the dataset $\mathcal{D}$ is processed in batches of size B. For a given batch $\mathcal{B}$, the model minimizes the empirical supervised cross-entropy loss:

$$\mathcal{L}_{\text{sup}} = \frac{1}{B}\sum_{i=1}^{B} -\log p_\theta(y_i \mid x_i), \tag{1}$$

where $p_\theta(y_i \mid x_i)$ is the softmax probability assigned to the ground-truth class y_i by the model.

3.2 BACKGROUND

Modern deep neural networks (DNNs) are typically trained using one-hot labels, which assign all probability mass to a single class (as shown in Eq. 1). This encourages models to produce highly confident predictions, leaving little room to express uncertainty. As a result, model confidence often becomes misaligned with true accuracy, leading to overconfident predictions (Guo et al., 2017). To address this issue, Thulasidasan et al. (2019) explored Mixup (Zhang et al., 2018), a data augmentation

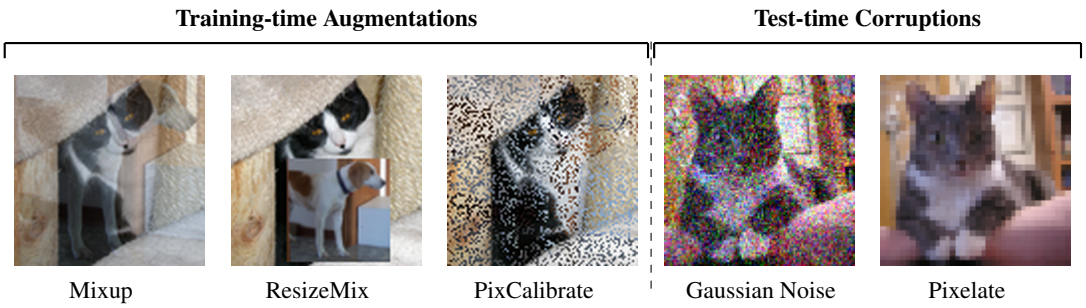


Figure 2: Visualization of training-time augmentation strategies and test-time corruption patterns on STL-10. The first three images show Mixup, ResizeMix, and PixCalibrate applied to the same clean cat image, while the two rightmost images show STL-10-C test samples with Gaussian Noise and Pixelate at severity level 3.

strategy that constructs synthetic examples by linearly interpolating pairs of training examples and their labels. Formally, given two examples (x_i, y_i) and (x_j, y_j), Mixup generates:

$$\tilde{x} = \lambda x_i + (1 - \lambda)x_j, \qquad \tilde{y} = \lambda y_i + (1 - \lambda)y_j \tag{2}$$

where x_i and x_j are two randomly selected training examples with corresponding one-hot labels y_i and y_j. The interpolation coefficient $\lambda \sim \mathrm{Beta}(\alpha, \alpha)$ controls the strength of mixing. Unlike standard one-hot targets, the mixed label $\tilde{y}$ assigns probability mass to multiple classes, increasing target entropy and introducing uncertainty during training. This helps reduce overconfident predictions and improves calibration. However, calibration under distribution shift requires more than soft targets. A model f_θ should also be locally consistent under small corruptions or perturbations such that

$$f_\theta(x + \delta_c) \approx f_\theta(x), \tag{3}$$

where δ_c denotes a corruption-induced perturbation. While Mixup encourages smoothness between training samples, its global linear interpolation fails to preserve the semantic structure of the original images and does not necessarily match the fine-grained, pixel-level distortions observed in real-world corrupted test data. This mismatch becomes problematic because standard empirical risk minimization assumes that train and test samples are drawn from the same distribution, whereas corrupted samples violate this assumption. Thus, models trained on clean inputs often fail to generalize under corruption, leading to low accuracy and poorly calibrated predictions.

Unlike global interpolation, patch-based augmentation strategies (DeVries & Taylor, 2017; Qin et al., 2020; Yun et al., 2019) preserve partial semantic content, but they introduce coarse patches with sharp boundaries. Consequently, existing augmentation methods suffer from two limitations. First, in terms of spatial perturbation, neither global interpolation nor patch-based techniques capture the fine-grained, spatially distributed nature of real corruptions like Gaussian Noise or Pixelate (Fig. 2). Second, regarding sample selection, most variants rely on unguided, random sample pairing without considering sample learning difficulty. As a result, pairing easy samples yields minimal robustness gains, whereas pairing hard samples brings ambiguity and degrades both accuracy and calibration.

3.3 Proposed Approach: PixCalibrate

To overcome the limitations of global interpolation and patch-based augmentation, we introduce **PixCalibrate**, a difficulty-aware pixel-level mixing approach that generates semantically grounded, fine-grained perturbations through targeted sample pairing that better simulate real-world image corruptions. Given two samples (x_a, y_a) and (x_b, y_b), where x_a is the anchor image and x_b is the perturbation source image, and y_a and y_b are their corresponding one-hot labels, PixCalibrate constructs an augmented sample by fine-grained pixel-level mixing, replacing a subset of pixels from the anchor image x_a with pixels from the perturbation source x_b. We generate a binary mask $M \in \{0, 1\}^{H \times W}$ over spatial locations, where a fraction ρ of pixels are selected from the second sample:

$$\frac{\|M\|_0}{H \times W} = \rho. \tag{4}$$

Here, $\|M\|_0$ is the ℓ_0 pseudo-norm of the binary mask, which represents the number of non-zero entries. The binary mask $M \in \{0, 1\}^{H \times W}$ is sampled over spatial pixel locations and broadcast across the channel dimension, so all channels C at a selected pixel location are replaced together.

The augmented image and its target label are defined as:

$$\tilde{x} = (1 - M) \odot x_a + M \odot x_b, \tag{5}$$

$$\tilde{y} = (1 - \rho)y_a + \rho y_b. \tag{6}$$

The proposed formulation can be interpreted as introducing a structured perturbation to the input. Specifically, the augmented sample can be written as:

$$\tilde{x} = x_a + \delta, \quad \text{where} \quad \delta = M \odot (x_b - x_a). \tag{7}$$

Here, δ represents a localized and spatially distributed perturbation applied to image x_a. Unlike global interpolation or large patch replacement, this perturbation preserves the overall semantic structure of the input while introducing fine-grained variations. As a result, the model is encouraged to learn stable predictions under such perturbations, which closely resemble real-world corruptions.

The quality of the augmented sample depends on which two samples are mixed. Randomly pairing samples may combine two easy samples, which provide limited gains, or two hard samples, which can bring ambiguity. To obtain more reliable pairs, PixCalibrate uses learning difficulty estimated from Area Under the Margin (AUM) (Pleiss et al., 2020), which aggregates margin information over training epochs. By leveraging training dynamics over epochs, we obtain a more reliable estimate of sample difficulty. This is because confidence at a single epoch is often unreliable, as it may not faithfully capture the true margin or the correctness of predictions at a given epoch, as highlighted by (Sosea & Caragea, 2023). Formally, let $(x_i, y_i) \in \mathcal{D}$ be an input image x_i with ground-truth label y_i, and let $z_i^{(t)} \in \mathbb{R}^K$, where K is the number of classes, denote the logits produced by the model for this image at training epoch t. The margin of sample x_i with respect to y_i at epoch t is defined as:

$$m_i^{(t)} = z_{i,y_i}^{(t)} - \max_{j \neq y_i} z_{i,j}^{(t)}. \tag{8}$$

We accumulate this margin over the training epochs, up until the current epoch t to quantify the learning difficulty of each sample at epoch t. Specifically, the AUM for sample i is computed as:

$$\mathrm{AUM}_i^{(t)} = \frac{1}{t} \sum_{s=1}^{t} m_i^{(s)}. \tag{9}$$

Thus, $AUM_i^{(t)}$ represents the average classification margin of sample i observed up to epoch t (note that, in practice, we use warmup for the margins to stabilize). Samples with high AUM are treated as easy, while those with low or negative AUM are considered hard or ambiguous. PixCalibrate uses this distinction to form difficulty-aware pairs by splitting each batch $\mathcal{B}$ at its median AUM value τ, and randomly pairing samples from the easy subset (with AUM $\geq \tau$) with those from the hard subset (with AUM $< \tau$) using Eqs. 5 and 6. An easy sample x_e serves as the semantic anchor, while a hard sample x_h acts as the perturbation source. This pixel-level formulation also mitigates a potential limitation of selective Mixup: globally interpolating deliberately selected samples can shift the effective training distribution by introducing artificial features that may not resemble natural images. In PixCalibrate, the selected hard sample does not globally redefine the augmented example; it only contributes a masked perturbation, $\delta = M \odot (x_h - x_e)$, to the easy anchor. Therefore, the semantic identity of the augmented sample is primarily preserved by x_e, while x_h introduces localized uncertainty. The augmented samples are associated with soft targets (Eq. 6), which encourage smoother transitions in the predictive distribution. However, despite such regularization, neural networks tend to produce overconfident predictions due to the nature of cross-entropy optimization. To mitigate this effect, we incorporate a confidence penalty (Pereyra et al., 2017), which discourages overly sharp predictive distributions by encouraging higher output entropy. We apply it to both supervised and augmented samples. This serves complementary roles: it prevents overconfident predictions on clean data while appropriately regularizing the inherently ambiguous augmented samples.

Accordingly, the supervised loss for the supervised batch of size B is defined as:

$$\mathcal{L}_{\text{sup}} = \frac{1}{B} \sum_{i=1}^{B} \left[-\log p_\theta(y_i \mid x_i) - \beta \, \mathcal{H}(p_\theta(\cdot \mid x_i)) \right], \tag{10}$$

and the augmented loss for the augmented batch of size $\tilde{B}$ is given by:

$$\mathcal{L}_{\text{mix}} = \frac{1}{\tilde{B}} \sum_{i=1}^{\tilde{B}} \left[-\sum_{k=1}^{K} \tilde{y}_{i,k} \log p_\theta(k \mid \tilde{x}_i) - \beta \, \mathcal{H}(p_\theta(\cdot \mid \tilde{x}_i)) \right], \tag{11}$$

where $\mathcal{H}(p_\theta(\cdot \mid x)) = -\sum_{k=1}^{K} p_\theta(k \mid x) \log p_\theta(k \mid x)$ denotes the entropy of the predictive distribution, and β controls the strength of the confidence penalty.

The overall training loss is defined as:

$$\mathcal{L} = \mathcal{L}_{\text{sup}} + \mathcal{L}_{\text{mix}}. \tag{12}$$

The PixCalibrate algorithm is shown in Appendix A.1.

4 Experimental Setup

Adopting settings similar to prior image-classification studies (Wang et al., 2022; Rahman et al., 2026), we evaluate PixCalibrate on the corrupted versions of standard image classification datasets: CIFAR-10-C, CIFAR-100-C (Hendrycks & Dietterich, 2019), our newly introduced SVHN-C and STL-10-C (details of the dataset construction are provided in A.7), and the large-scale dataset ImageNet-C (Hendrycks & Dietterich, 2019), ImageNet-O and ImageNet-A (Hendrycks et al., 2021). All experiments are conducted in the FullySupervised setting, where models are trained on clean training data and evaluated on the corrupted test sets. We report results averaged across all 15 corruption types (spanning noise, blur, weather, and digital categories; we further provide per category performance analysis in Appendix A.6.1), and across all 5 severity levels. We first train the model with the standard supervised loss for ten warmup epochs (see Appendix A.6.3 for sensitivity analysis). Following this warmup phase, each sample's AUM is initialized to zero and updated at every subsequent epoch to separate easy and hard samples. We experiment with different values of the corruption ratio $\rho \in \{0.05, 0.2, 0.4, 0.45, 0.50\}$ and set $\rho = 0.4$ in our main experiments. Similarly, after experimenting with the confidence penalty over $\beta \in \{0.01, 0.1, 1\}$, we set $\beta = 0.1$. Experiments are repeated three times, and we report the mean Accuracy (Acc), Expected Calibration Error (ECE) (Guo et al., 2017), and Brier Score (Brier) in Table 1. For ease of readability, we provide the mean and standard deviation across 3 runs in Table 6 in Appendix A.3. The detailed experimental setup is provided in Appendix A.2 and a detailed hyperparameter analysis is provided in Appendix A.6.4.

5 Result and Observations

5.1 Performance on CIFAR-10-C, CIFAR-100-C, SVHN-C, STL-10-C

We evaluate PixCalibrate on corrupted small-scale benchmarks, including CIFAR-10-C, CIFAR-100-C, STL-10-C, and SVHN-C. Results are reported in Table 1. We compare against strong augmentation baselines, including CalibrateMix (Rahman et al., 2026), Mixup (Zhang et al., 2018), CutMix (Yun et al., 2019), ResizeMix (Qin et al., 2020), CutOut (DeVries & Taylor, 2017), SnapMix (Huang et al., 2021), and TdAttenMix (Wang et al., 2025). We consider two variants of our method, different-class mixing (DC), where pixel-level mixing is applied, combining samples based on their AUM values regardless of the class, and same-class mixing (SC), where mixing is applied, combining samples based on their AUM values but restricted to samples from the same class.

Across all datasets, PixCalibrate improves both calibration and robustness over the FullySupervised baseline and prior augmentation methods. On CIFAR-10-C, PixCalibrate achieves the strongest overall performance. In particular, DC improves accuracy from 73.48% to 84.70% (an increase of 11.22), while reducing ECE from 15.18 to 7.64 (a reduction of 7.54) and reducing Brier Score. These gains exceed those of CutMix and Mixup, indicating that AUM-guided pixel-level mixing is more effective than global interpolation and patch-based methods. On CIFAR-100-C, similar trends are observed. DC achieves an accuracy of 65.03%, corresponding to an improvement of 13.27 over the baseline, and attains the lowest ECE of 8.88, reducing calibration error by over 13. In contrast, methods such as CutMix and ResizeMix exhibit higher ECE despite small improvements in accuracy

Table 1: Performance comparison on CIFAR-10-C, CIFAR-100-C, STL-10-C, and SVHN-C. Mean values are reported across 3 runs. For standard deviations, refer to Table 6 in the Appendix. For Accuracy (Acc) higher is better, while for Expected Calibration Error (ECE) and Brier Score lower is better. Best scores are shown in **bold**. * denotes statistically significant improvement over the FullySupervised Base using a paired two-sided t-test ($p < 0.05$) computed over all corruption types and severity levels.

	CIFAR-10-C			CIFAR-100-C			STL-10-C			SVHN-C		
Model	Acc ↑	ECE ↓	Brier ↓	Acc ↑	ECE ↓	Brier ↓	Acc ↑	ECE ↓	Brier ↓	Acc ↑	ECE ↓	Brier ↓
FullySupervised Base	73.48	15.18	0.41	51.76	22.18	0.68	47.29	31.41	0.79	66.00	9.59	0.42
CalibrateMix	75.54	11.77	0.37	51.60	18.01	0.65	53.28	28.67	0.72	65.24	8.93	0.43
Mixup	75.41	11.15	0.36	51.67	18.35	0.66	54.30	27.68	0.71	66.55	8.91	0.42
CutMix	73.58	13.84	0.39	53.56	26.06	0.70	42.21	40.72	0.94	60.22	8.93	0.49
ResizeMix	73.84	12.65	0.38	54.11	25.46	0.68	42.84	38.79	0.90	59.87	8.55	0.49
CutOut	74.78	14.01	0.38	52.47	25.00	0.69	44.47	36.91	0.87	62.60	9.22	0.46
SnapMix	72.04	12.21	0.40	51.36	24.88	0.71	43.67	25.84	0.81	53.94	16.95	0.60
TdAttenMix	59.53	20.54	0.64	39.43	18.37	0.81	39.93	22.70	0.82	59.81	20.19	0.52
PixCalibrate DC (Ours)	**84.70***	**7.64***	**0.22***	65.03*	**8.88***	**0.47***	**59.75***	21.03*	0.60*	**68.17***	**8.10**	**0.41**
PixCalibrate SC	84.05*	9.56*	0.24*	**65.19***	10.27*	0.48*	59.38*	**20.33***	**0.58***	68.15*	8.67	**0.41**

Table 2: Performance comparison on ImageNet, ImageNet-C, ImageNet-A, and ImageNet-O. Best scores are shown in **bold**. For ImageNet-O, OOD-MSP represents the Mean OOD-MSP.

	ImageNet			ImageNet-C			ImageNet-A			ImageNet-O		
Model	Acc ↑	ECE ↓	Brier ↓	Acc ↑	ECE ↓	Brier ↓	Acc ↑	ECE ↓	Brier ↓	AUROC ↑	OOD-MSP ↓	AUPR-OOD ↑
FullySupervised Base	71.76	5.46	0.39	26.93	22.48	0.91	1.40	46.67	1.28	47.8	81.26	3.42
PixCalibrate DC (Ours)	72.79	**3.74**	**0.37**	32.53	**15.86**	**0.83**	1.49	**44.75**	**1.23**	**49.1**	**76.5**	3.55
PixCalibrate SC	**73.14**	4.80	0.37	**33.22**	20.61	0.85	**1.69**	44.89	1.28	49.01	79.8	**3.6**

over FullySupervised, suggesting that patch-based augmentations are less effective under corruption. On STL-10-C, PixCalibrate improves accuracy and calibration, with DC boosting accuracy from 47.29 to 59.75 (a gain of 12.46), while maintaining strong calibration performance. On SVHN-C, PixCalibrate similarly improves calibration and accuracy, with DC reducing ECE from 9.59 to 8.10, while maintaining strong predictive performance. Overall, PixCalibrate consistently achieves higher accuracy, lower ECE, and Brier Scores across datasets compared to prior approaches.

5.2 Performance on ImageNet, ImageNet-C, ImageNet-O, ImageNet-A

We evaluate PixCalibrate on ImageNet, ImageNet-C, ImageNet-A, and ImageNet-O to assess performance across clean images, common synthetic corruptions, naturally occurring adversarial examples, and out-of-distribution inputs respectively. Similar to the small-scale benchmark experiments, we consider two variants of PixCalibrate - DC and SC. In Table 2 we report accuracy (Acc), Expected Calibration Error (ECE), and Brier Score for ImageNet, ImageNet-C, and ImageNet-A. For ImageNet-O, we report OOD detection metrics since its samples lie outside the ImageNet-1K label space, making standard top-1 accuracy, ECE and Brier scores unsuitable. Following standard practice (Hendrycks & Gimpel, 2017; Srivastava et al., 2022), we report the ID/OOD separability measure AUROC and AUPR-OOD measure, while the mean OOD MSP quantifies the model's confidence in OOD inputs using the maximum softmax probability baseline (Hendrycks & Gimpel, 2017). We observe that, on clean ImageNet, PixCalibrate improves calibration as well as accuracy. In particular, DC reduces ECE from 5.46 to 3.74. On ImageNet-C, DC increases accuracy from 26.93% to 32.53%, while reducing ECE from 22.48 to 15.86 and reducing the Brier Score. On the adversarial benchmark: ImageNet-A, DC reduces ECE from 46.67 to 44.75, indicating more reliable confidence estimates even in extremely challenging scenarios where accuracy remains low. On ImageNet-O, PixCalibrate improves OOD detection and confidence behavior, achieving higher AUROC and lower Mean OOD MSP. These results highlight that PixCalibrate is particularly effective at improving accuracy and calibration under adversarial, out-of-distribution and corrupted settings, marking its effectiveness.

6 Analysis

6.1 Quantitative Similarity to Corruption Perturbations

To quantify how closely our pixel-level augmentations resemble common corruptions, we compare the augmentation perturbation $\tilde{x} - x$ with the corresponding corruption perturbation $x_c - x$, where x is the original clean image, $\tilde{x}$ is the augmented image, and x_c is its corrupted counterpart. We measure structural similarity (SSIM) (Wang et al., 2004) and frequency-spectrum distance (Fuoli et al., 2021). As shown in Table 3, PixCalibrate achieves the highest SSIM (0.36), compared with Mixup (0.20) and CutMix (0.27), while also obtaining the lowest frequency distance. These results

Table 3: Similarity to CIFAR-10-C corruption perturbations.

Method	SSIM ↑	Freq. Dist. ↓
Mixup	0.20	0.1717
CutMix	0.27	0.1778
PixCalibrate	**0.36**	**0.1716**

Table 4: Performance on non-synthetic distribution shifts. Each entry reports Accuracy / ECE. Higher Accuracy and lower ECE are better.

Method	CIFAR-10.1	CIFAR-10.2	CINIC-10	CIFAR-10-W
FullySupervised	88.50/5.80	83.20/10.71	76.23/15.05	76.67/15.24
Mixup	88.75/**4.10**	83.75/9.98	76.49/15.20	77.60/15.10
CutMix	88.30/5.79	83.45/10.15	76.50/13.16	77.22/14.54
PixCalibrate	**88.95**/5.67	**83.80**/**9.95**	**77.20**/**12.62**	**77.79**/**13.42**

indicate that PixCalibrate generates perturbations that are more structurally aligned with common corruption patterns.

6.2 Generalization to Non-Synthetic Distribution Shifts

To determine whether the benefits of PixCalibrate extend beyond synthetic corruptions, we additionally evaluate CIFAR-10-trained models on four non-synthetic distribution-shift benchmarks: CIFAR-10.1 (Recht et al., 2018), CIFAR-10.2 (Lu et al., 2020), CINIC-10 (Darlow et al., 2018), and the real-image subset of CIFAR-10-W (Sun et al., 2024). These benchmarks contain independently collected or cross-source images rather than algorithmically corrupted versions of the original CIFAR-10 test images. The results are reported in Table 4. PixCalibrate achieves the highest accuracy on all four benchmarks. It also obtains the lowest ECE on CIFAR-10.2, CINIC-10, and CIFAR-10-W. These results indicate that the improvements of PixCalibrate are not restricted to synthetically corrupted test images and extend to independently collected and cross-source distribution shifts.

6.3 Ablation Study

We conduct ablation experiments to analyze the contribution of the key components in PixCalibrate: (1) Difficulty-aware pairing of images, (2) Confidence Penalty term with the training objective, (3) Effect of Semantic Anchor Selection, (4) Perturbation Construction Strategy.

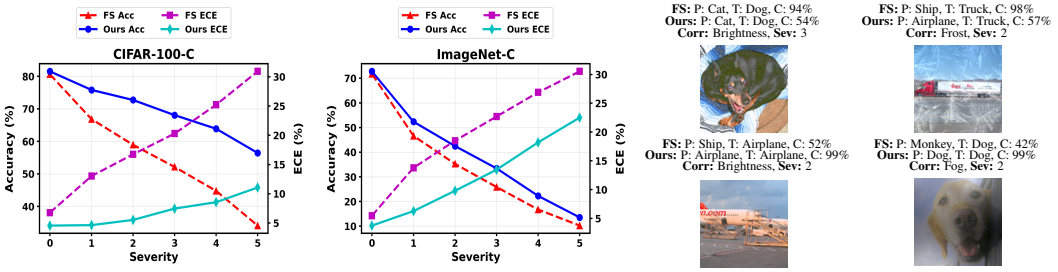
Difficulty-Aware Pairing of Images We compare four pairing strategies under the same pixel-level mixing operation: *Random Pairing*, which samples two images uniformly at random; *Easy-Easy Pairing*, where one easy sample serves as the anchor and another as the perturbation source; *Hard-Hard Pairing*, where one hard sample serves as the anchor and another as the perturbation source; *Ours*, where the easy sample serves as the anchor and the hard sample serves as the perturbation source. Sample difficulty is estimated using AUM. As shown in Table 5, *Ours* achieves the best calibration, reducing ECE overall on CIFAR-10-C (7.75 to 7.64) and CIFAR-100-C (8.92 to 8.88). These results suggest that AUM-guided easy-hard pairing provides clear benefits for calibration.

Confidence Penalty We also assess the effect of the confidence penalty by comparing PixCalibrate with and without confidence regularization. As shown in the middle section of Table 5, the confidence penalty consistently improves calibration, reducing ECE from 2.48 to 1.76 on CIFAR-10 (Krizhevsky et al., 2009), 8.43 to 7.64 on CIFAR-10-C, 5.93 to 4.55 on CIFAR-100 (Krizhevsky et al., 2009), and 10.15 to 8.88 on CIFAR-100-C. It also improves accuracy under corruption, indicating that confidence regularization reduces overconfident predictions and improves robustness under distribution shift. We perform a detailed analysis of the confidence penalty in Appendix A.6.2.

Semantic Anchor Selection We analyze the role of the semantic anchor in easy-hard pixel-level mixing by comparing three variants: using the easy sample as the anchor and the hard sample as the perturbation source, using the hard sample as the anchor and the easy sample as the perturbation source, and combining both configurations. As shown in Table 5, using the easy sample as the anchor and the hard sample as the perturbation source performs best overall, achieving 84.70% accuracy and

Table 5: Ablations of PixCalibrate on CIFAR-10, CIFAR-10-C, CIFAR-100, and CIFAR-100-C. A and P indicate the semantic anchor and perturbation source, while superscripts E and H denote easy-to-learn and hard-to-learn samples, respectively.

	CIFAR-10		CIFAR-10-C		CIFAR-100		CIFAR-100-C	
Model	Acc ↑	ECE ↓	Acc ↑	ECE ↓	Acc ↑	ECE ↓	Acc ↑	ECE ↓
Random Pairing	95.59	1.89	84.63	8.48	**81.94**	5.28	64.63	8.92
Easy-Easy Pairing	94.90	1.79	83.42	7.75	81.26	5.91	63.48	9.89
Hard-Hard Pairing	94.81	2.23	84.47	7.93	81.03	4.72	62.98	8.94
Ours	**95.60**	**1.76**	**84.70**	**7.64**	81.51	**4.55**	**65.03**	**8.88**
Ours w/o Confidence Penalty	95.37	2.48	83.23	8.43	**81.76**	5.93	63.01	10.15
Ours w/ Confidence Penalty	**95.60**	**1.76**	**84.70**	**7.64**	81.51	**4.55**	**65.03**	**8.88**
$A^{\mathrm{H}} + P^{\mathrm{E}}$	95.45	2.09	84.41	7.76	80.99	4.71	63.76	8.95
$A^{\mathrm{E}} + P^{\mathrm{H}}$, $A^{\mathrm{H}} + P^{\mathrm{E}}$	95.31	2.19	84.28	7.91	81.47	4.60	64.58	8.91
$A^{\mathrm{E}} + P^{\mathrm{H}}$ (Ours)	**95.60**	**1.76**	**84.70**	**7.64**	**81.51**	**4.55**	**65.03**	**8.88**
AugMix	94.35	2.22	84.13	7.82	80.08	7.44	61.82	11.86
PixMix	95.49	**1.65**	83.58	**3.57**	80.26	4.76	62.47	9.03
PixCalibrate (Ours)	**95.60**	1.76	**84.70**	7.64	**81.51**	**4.55**	**65.03**	**8.88**



(a) (b) (c)

Figure 3: Performance across corruption severity on (a) CIFAR-100-C and (b) ImageNet-C, and (c) qualitative error analysis on STL-10-C comparing Fully Supervised (FS) training with our method. P, T, C, Corr, and Sev denote predicted class, true class, confidence, corruption type, and corruption severity, respectively.

7.64 ECE on CIFAR-10-C, and 65.03% accuracy and 8.88 ECE on CIFAR-100-C. This suggests that easy samples provide more stable semantic anchors, while hard samples are more effective perturbation sources. In contrast, using hard samples as anchors can introduce greater ambiguity due to their less stable margins. These results support our design choice of preserving easy samples as semantic anchors while injecting pixels from hard samples as perturbations.

Perturbation Construction Strategy A central design choice in PixCalibrate is how perturbations are introduced into the training samples. To examine whether the observed improvements are specific to our pixel-level perturbation construction, we compare PixCalibrate with AugMix (Hendrycks et al., 2020) and PixMix (Hendrycks et al., 2022). All methods are evaluated using the same backbone and training protocol. As shown in Table 5, PixCalibrate achieves the highest corruption accuracy on both CIFAR-10-C (84.70%) and CIFAR-100-C (65.03%). While PixMix obtains lower ECE on CIFAR-10-C, PixCalibrate provides stronger accuracy and competitive calibration across both corrupted benchmarks. These results suggest that the manner in which perturbations are constructed is important: rather than relying on predefined transformation chains as in AugMix or auxiliary mixing images as in PixMix, PixCalibrate generates fine-grained perturbations directly from training samples through difficulty-aware pixel-level mixing.

6.4 Error Analysis

To qualitatively assess PixCalibrate, we compare predictions from the FullySupervised (FS) baseline and PixCalibrate in Figures 3a and 3b on CIFAR-100-C and ImageNet-C, respectively. As corruption severity increases, PixCalibrate remains robust under increasingly severe distribution shifts, exhibiting a gradual decline in accuracy and a slower increase in ECE than standard baselines. This indicates substantially slower degradation in both robustness and calibration as the corruption level increases. We also compare predictions from the FullySupervised (FS) baseline and PixCalibrate in Figure 3c. The figure shows representative STL-10-C samples across different corruption types and severities, along with predicted labels, confidence scores, and ground-truth labels. In the top row, both models misclassify the images; however, FS is highly overconfident in its incorrect predictions, while PixCalibrate assigns substantially lower confidence, indicating better uncertainty estimation under challenging corruptions. This is also reflected across all other corruption types and severities. In the bottom row, FS produces incorrect predictions, whereas PixCalibrate correctly classifies the same samples with high confidence. These examples suggest that PixCalibrate improves calibration by reducing overconfident errors while preserving strong confidence when predictions are correct under corrupted test distributions.

7 Conclusion

In this paper, we introduced PixCalibrate, a training-time difficulty-aware pixel-level mixup framework designed to improve the robustness and calibration under corrupted settings. PixCalibrate preserves the semantic structure of the original image while introducing fine-grained pixel-level perturbations. Our new method is lightweight and achieves state-of-the-art performance in terms of accuracy, ECE, and Brier score in multiple corrupted benchmarks. This contributes to reliable AI systems in real-world applications. For future work we aim to explore test time strategies and extend our method to tasks such as object detection, segmentation and multimodal learning under corrupted settings.

AI USE STATEMENT

In this work, we used generative AI tools to assist with gathering, sourcing, and searching for relevant literature, as well as summarizing existing literature during the research and writing process. We did not use generative AI tools for any tasks requiring mandatory disclosure such as, the methodology, experiments, citations, formulating mathematical claims, developing proofs, or interpreting experimental results. We reviewed and verified all the outputs. The authors take full responsibility for the final content of this work, including any text or claims.

ETHICS STATEMENT

Our work focuses on improving the robustness and calibration of image classification models under distribution shift and does not involve human subjects. We conduct our experiments using established, publicly available image-classification benchmarks and their corrupted or distribution-shifted variants. We additionally construct corrupted versions of SVHN and STL-10 by applying standard synthetic corruption procedures to existing benchmark images and do not introduce sensitive data. We report our methodology and experimental results and will release the corresponding code to facilitate reproducibility and responsible reuse. To the best of our knowledge, our work does not raise discrimination/bias/fairness concerns, privacy and security issues, legal compliance issues, research integrity issues, or other ethical concerns.

REPRODUCIBILITY STATEMENT

We provide detailed information to support reproduction of our results. The complete PixCalibrate procedure, including AUM-based difficulty estimation, easy-hard sample pairing, pixel-level mask generation, and the training objective, is described in Section 3 and Algorithm A.1 in the Appendix. Detailed implementation and training settings, including model architectures, optimizers, learning-rate schedules, batch sizes, training duration, EMA settings, corruption ratio ρ, and confidence-penalty weight β, are provided in Appendix A.2. We additionally provide the construction procedure for the newly introduced SVHN-C and STL-10-C benchmarks, hyperparameter analyses, ablation studies, and per-corruption results in the Appendix. Experiments on the small-scale benchmarks are repeated three times, with mean and standard deviation reported. We will release the source code upon the acceptance of the manuscript.

REFERENCES

Kyungjune Baek, Duhyeon Bang, and Hyunjung Shim. Gridmix: Strong regularization through local context mapping. *Pattern Recognition*, 109:107594, 2021. ISSN 0031-3203. doi: https://doi.org/10.1016/j.patcog.2020.107594. URL https://www.sciencedirect.com/science/article/pii/S0031320320303976.

Peter Bartlett, Dylan J Foster, and Matus J Telgarsky. Spectrally-normalized margin bounds for neural networks. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper_files/paper/2017/file/b22b257ad0519d4500539da3c8bcf4dd-Paper.pdf.

Adam Coates, Andrew Ng, and Honglak Lee. An analysis of single-layer networks in unsupervised feature learning. In Geoffrey Gordon, David Dunson, and Miroslav Dudík (eds.), *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, volume 15 of *Proceedings of Machine Learning Research*, pp. 215–223, Fort Lauderdale, FL, USA, 11–13 Apr 2011. PMLR. URL https://proceedings.mlr.press/v15/coates11a.html.

Ali Dabouei, Sobhan Soleymani, Fariborz Taherkhani, and Nasser M. Nasrabadi. Supermix: Supervising the mixing data augmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 13794–13803, June 2021.

Luke N Darlow, Elliot J Crowley, Antreas Antoniou, and Amos J Storkey. Cinic-10 is not imagenet or cifar-10. *arXiv preprint arXiv:1810.03505*, 2018.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 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.

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

Gamaleldin Elsayed, Dilip Krishnan, Hossein Mobahi, Kevin Regan, and Samy Bengio. Large margin deep networks for classification. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. URL https://proceedings.neurips.cc/paper_files/paper/2018/file/42998cf32d552343bc8e460416382dca-Paper.pdf.

Dario Fuoli, Luc Van Gool, and Radu Timofte. Fourier space losses for efficient perceptual image super-resolution. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 2360–2369, October 2021.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger. On calibration of modern neural networks. In *International conference on machine learning*, pp. 1321–1330. PMLR, 2017.

Ethan Harris, Antonia Marcu, Matthew Painter, Mahesan Niranjan, Adam Prügel-Bennett, and Jonathon Hare. Fmix: Enhancing mixed sample data augmentation. *arXiv preprint arXiv:2002.12047*, 2020.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

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

Dan Hendrycks and Kevin Gimpel. A baseline for detecting misclassified and out-of-distribution examples in neural networks. In *International Conference on Learning Representations*, 2017. URL https://openreview.net/forum?id=Hkg4TI9xl.

Dan Hendrycks, Norman Mu, Ekin Dogus Cubuk, Barret Zoph, Justin Gilmer, and Balaji Lakshminarayanan. Augmix: A simple method to improve robustness and uncertainty under data shift. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=S1gmrxHFvB.

Dan Hendrycks, Kevin Zhao, Steven Basart, Jacob Steinhardt, and Dawn Song. Natural adversarial examples. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 15262–15271, 2021.

Dan Hendrycks, Andy Zou, Mantas Mazeika, Leonard Tang, Bo Li, Dawn Song, and Jacob Steinhardt. Pixmix: Dreamlike pictures comprehensively improve safety measures. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 16783–16792, June 2022.

Shaoli Huang, Xinchao Wang, and Dacheng Tao. Snapmix: Semantically proportional mixing for augmenting fine-grained data. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pp. 1628–1636, 2021.

Jang-Hyun Kim, Wonho Choo, and Hyun Oh Song. Puzzle mix: Exploiting saliency and local statistics for optimal mixup. In *International conference on machine learning*, pp. 5275–5285. PMLR, 2020.

JangHyun Kim, Wonho Choo, Hosan Jeong, and Hyun Oh Song. Co-mixup: Saliency guided joint mixup with supermodular diversity. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=gvxJzw8kW4b.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436–444, 2015.

Jin-Ha Lee, Muhammad Zaigham Zaheer, Marcella Astrid, and Seung-Ik Lee. Smoothmix: a simple yet effective data augmentation to train robust classifiers. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pp. 3264–3274. IEEE, 2020.

Jihao Liu, Boxiao Liu, Hang Zhou, Hongsheng Li, and Yu Liu. Tokenmix: Rethinking image mixing for data augmentation in vision transformers. In *European conference on computer vision*, pp. 455–471. Springer, 2022.

Shangyun Lu, Bradley Nott, Aaron Olson, Alberto Todeschini, Hossein Vahabi, Yair Carmon, and Ludwig Schmidt. Harder or different? a closer look at distribution shift in dataset reproduction. In *ICML Workshop on Uncertainty and Robustness in Deep Learning*, volume 5, pp. 15, 2020.

Amitha Mathew, P. Amudha, and S. Sivakumari. Deep learning techniques: An overview. In Aboul Ella Hassanien, Roheet Bhatnagar, and Ashraf Darwish (eds.), *Advanced Machine Learning Technologies and Applications*, pp. 599–608, Singapore, 2021. Springer Singapore. ISBN 978-981-15-3383-9.

Agnieszka Mikołajczyk and Michał Grochowski. Data augmentation for improving deep learning in image classification problem. In *2018 international interdisciplinary PhD workshop (IIPhDW)*, pp. 117–122. IEEE, 2018.

Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y. Ng. Reading digits in natural images with unsupervised feature learning. In *NIPS Workshop on Deep Learning and Unsupervised Feature Learning 2011*, 2011. URL http://ufldl.stanford.edu/housenumbers/nips2011_housenumbers.pdf.

Yaniv Ovadia, Emily Fertig, Jie Ren, Zachary Nado, D. Sculley, Sebastian Nowozin, Joshua Dillon, Balaji Lakshminarayanan, and Jasper Snoek. Can you trust your model's uncertainty? evaluating predictive uncertainty under dataset shift. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper_files/paper/2019/file/8558cb408c1d76621371888657d2eb1d-Paper.pdf.

Seo Yeon Park and Cornelia Caragea. On the calibration of pre-trained language models using mixup guided by area under the margin and saliency. In Smaranda Muresan, Preslav Nakov, and Aline Villavicencio (eds.), *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 5364–5374, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.acl-long.368. URL https://aclanthology.org/2022.acl-long.368/.

Gabriel Pereyra, George Tucker, Jan Chorowski, Lukasz Kaiser, and Geoffrey Hinton. Regularizing neural networks by penalizing confident output distributions, 2017. URL https://openreview.net/forum?id=HkCjNI5ex.

Francesco Pinto, Harry Yang, Ser Nam Lim, Philip Torr, and Puneet Dokania. Using mixup as a regularizer can surprisingly improve accuracy & out-of-distribution robustness. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 14608–14622. Curran Associates, Inc., 2022. doi: 10.52202/068431-1062. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/5ddcfaad1cb72ce6f1a365e8f1ecf791-Paper-Conference.pdf.

Geoff Pleiss, Tianyi Zhang, Ethan Elenberg, and Kilian Weinberger. Identifying mislabeled data using the area under the margin ranking. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 17044–17056. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/c6102b3727b2a7d8b1bb6981147081ef-Paper.pdf.

Jie Qin, Jiemin Fang, Qian Zhang, Wenyu Liu, Xingang Wang, and Xinggang Wang. Resizemix: Mixing data with preserved object information and true labels. *arXiv preprint arXiv:2012.11101*, 2020.

Mehrab Mustafy Rahman, Jayanth Mohan, Tiberiu Sosea, and Cornelia Caragea. On the calibration of image semi-supervised learning models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 25073–25081, 2026.

Benjamin Recht, Rebecca Roelofs, Ludwig Schmidt, and Vaishaal Shankar. Do cifar-10 classifiers generalize to cifar-10? *arXiv preprint arXiv:1806.00451*, 2018.

Connor Shorten and Taghi M Khoshgoftaar. A survey on image data augmentation for deep learning. *Journal of big data*, 6(1):60, 2019.

Tiberiu Sosea and Cornelia Caragea. Marginmatch: Improving semi-supervised learning with pseudo-margins. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 15773–15782, June 2023.

Anugya Srivastava, Shriya Jain, and Mugdha Thigle. Out of distribution detection on imagenet-o. *arXiv preprint arXiv:2201.09352*, 2022.

Xiaoxiao Sun, Xingjian Leng, Zijian Wang, Yang Yang, Zi Huang, and Liang Zheng. CIFAR-10-warehouse: Broad and more realistic testbeds in model generalization analysis. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=pw2ssoOTpo.

Luke Taylor and Geoff Nitschke. Improving deep learning with generic data augmentation. In *2018 IEEE symposium series on computational intelligence (SSCI)*, pp. 1542–1547. IEEE, 2018.

Sunil Thulasidasan, Gopinath Chennupati, Jeff A Bilmes, Tanmoy Bhattacharya, and Sarah Michalak. On mixup training: Improved calibration and predictive uncertainty for deep neural networks. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper_files/paper/2019/file/36ad8b5f42db492827016448975cc22d-Paper.pdf.

A F M Shahab Uddin, Mst. Sirazam Monira, Wheemyung Shin, TaeChoong Chung, and Sung-Ho Bae. Saliencymix: A saliency guided data augmentation strategy for better regularization. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=-M0QkvBGTTq.

Shashanka Venkataramanan, Ewa Kijak, Laurent Amsaleg, and Yannis Avrithis. Alignmixup: Improving representations by interpolating aligned features. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 19174–19183, June 2022.

Yidong Wang, Hao Chen, Yue Fan, Wang SUN, Ran Tao, Wenxin Hou, Renjie Wang, Linyi Yang, Zhi Zhou, Lan-Zhe Guo, Heli Qi, Zhen Wu, Yu-Feng Li, Satoshi Nakamura, Wei Ye, Marios Savvides, Bhiksha Raj, Takahiro Shinozaki, Bernt Schiele, Jindong Wang, Xing Xie, and Yue Zhang. Usb: A unified semi-supervised learning benchmark for classification. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 3938–3961. Curran Associates, Inc., 2022. doi: 10.52202/068431-0285. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/190dd6a5735822f05646dc27decff19b-Paper-Datasets_and_Benchmarks.pdf.

Zhiming Wang, Lin Gu, and Feng Lu. Tdattenmix: top-down attention guided mixup. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 8232–8240, 2025.

Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004.

Suorong Yang, Weikang Xiao, Mengchen Zhang, Suhan Guo, Jian Zhao, and Furao Shen. Image data augmentation for deep learning: A survey. *arXiv preprint arXiv:2204.08610*, 2022.

Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019.

Bianca Zadrozny and Charles Elkan. Obtaining Calibrated Probability Estimates from Decision Trees and Naive Bayesian Classifiers. In *International Conference on Machine Learning*, pp. 609–616, 2001. URL https://mlanthology.org/icml/2001/zadrozny2001icml-obtaining/.

Bianca Zadrozny and Charles Elkan. Transforming classifier scores into accurate multiclass probability estimates. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 694–699, 2002.

Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. In Edwin R. Hancock Richard C. Wilson and William A. P. Smith (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.

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

Linjun Zhang, Zhun Deng, Kenji Kawaguchi, Amirata Ghorbani, and James Zou. How does mixup help with robustness and generalization? In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=8yKEo06dKNo.

Linjun Zhang, Zhun Deng, Kenji Kawaguchi, and James Zou. When and how mixup improves calibration. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato (eds.), *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 26135–26160. PMLR, 17–23 Jul 2022. URL https://proceedings.mlr.press/v162/zhang22f.html.

Zhun Zhong, Liang Zheng, Guoliang Kang, Shaozi Li, and Yi Yang. Random erasing data augmentation. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pp. 13001–13008, 2020.

A Appendix

A.1 Algorithm

We present the algorithm of PixCalibrate in this section. After the warmup period, samples within each mini-batch are partitioned into easy and hard subsets based on their AUM scores, and pixel-level mixed samples are generated by using easy samples as semantic anchors and hard samples as perturbation sources. The model is then optimized jointly using the supervised and mixed-sample losses, while the AUM statistics are updated throughout training.

Algorithm 1 PixCalibrate

Require: Dataset $\mathcal{D}$; model f parametrized by θ; Batch size B; epochs T; corruption ratio ρ; confidence weight β; warmup epoch T_w

Initialize Area Under Margin (AUM) values after T_w
for $t = 1$ to T **do**
  for each batch $\mathcal{B} = \{(x_i, y_i)\}_{i=1}^{B}$ in $\mathcal{D}$ **do**
    Retrieve $\mathrm{AUM}_i^{(t-1)}$ for samples in $\mathcal{B}$
    Compute the median AUM value τ within $\mathcal{B}$ and split:
      $\mathcal{B}_{easy} = \{(x_i, y_i) \mid \mathrm{AUM}_i^{(t-1)} \geq \tau\}$, $\mathcal{B}_{hard} = \{(x_i, y_i) \mid \mathrm{AUM}_i^{(t-1)} < \tau\}$
    for $k = 1$ to $|\mathcal{B}_{easy}|$ **do**
      Sample $(x_e, y_e) \sim \mathcal{B}_{easy}$, $(x_h, y_h) \sim \mathcal{B}_{hard}$
      Generate mask M_k with $\frac{\|M_k\|_0}{H \times W} = \rho$
      $\tilde{x}_k = (1 - M_k) \odot x_e + M_k \odot x_h,\ \tilde{y}_k = (1 - \rho) y_e + \rho y_h$
    end for
    $\tilde{\mathcal{B}} = \{(\tilde{x}_k, \tilde{y}_k)\}_{k=1}^{|\mathcal{B}_{easy}|}$
    Compute $\mathcal{L}_{sup}$ on $\mathcal{B}$ and $\mathcal{L}_{mix}$ on $\tilde{\mathcal{B}}$ using Eqs. 10, 11
    Compute margin $m_i^{(t)}$ using Eq. 8
    Update: $\mathrm{AUM}_i^{(t)}$ using Eq. 9
    $\mathcal{L} \leftarrow \mathcal{L}_{sup} + \mathcal{L}_{mix}$, update θ
  end for
end for

A.2 Detailed Experimental Setup & Implementation Details

We evaluate PixCalibrate on the corrupted versions of standard image classification benchmark datasets, including the small-scale datasets: CIFAR-10-C, CIFAR-100-C (Hendrycks & Dietterich, 2019), SVHN-C, STL-10-C (details of the dataset construction are provided in A.7), and the large-scale dataset ImageNet-C (Hendrycks & Dietterich, 2019). We additionally report results on ImageNet-O and ImageNet-A (Hendrycks et al., 2021) to assess robustness under Out-of-domain and adversarial scenarios. All experiments are conducted in the fully supervised setting, where models are trained on clean training data and evaluated on the corrupted test sets. We report results averaged across all corruption types and severity levels. All experiments are repeated three times, and we report the mean and standard deviation for Accuracy (Acc), Expected Calibration Error (ECE) (Guo et al., 2017), and Brier Score (Brier).

We adopt architectures and optimization settings similar to prior image-classification studies (Wang et al., 2022; Rahman et al., 2026). We use Wide Residual Networks (Zagoruyko & Komodakis, 2016) for small-scale experiments. Specifically, we use WRN-28-2 for CIFAR-10 (Krizhevsky et al., 2009) and SVHN, WRN-37-2 for STL-10, and WRN-28-8 for CIFAR-100 (Krizhevsky et al., 2009); and use ResNet-50 (He et al., 2016) for the large-scale ImageNet experiments. Models are trained using the SGD optimizer with momentum 0.9, weight decay 5×10^{-4}. We use a cosine annealing learning-rate scheduling to dynamically adjust the learning rate over 1000 epochs with an initial learning rate of 0.03. To improve training stability and computational efficiency, we apply an Exponential Moving Average (EMA) to the weights with a decay momentum of 0.999 while utilizing mixed-precision training. For each dataset, the labeled batch size is set to 64, and the augmented batch size is set to 32. We set the corruption ratio ρ as 0.4 and the confidence penalty β as 0.1. All small-scale experiments are run for 1000 epochs.

For ImageNet training, we use a learning rate schedule with a short linear warmup phase followed by step-wise decay, where the learning rate is gradually increased at the start of training and reduced at several predefined epochs. Following Rahman et al. (2026), the initial learning rate is set to 0.4. The labeled batch is set to be 1024, and the augmented batch size is set to be 512. All ImageNet experiments are run for 300 epochs.

The same-class (SC) variant follows the same PixCalibrate pipeline as the primary different-class (DC) setting, but restricts the hard perturbation source to the same semantic class whenever possible. Given an easy sample (x_e, y_e), we first search for hard samples from the same class within the current mini-batch:

$$\mathcal{H}_{y_e} = \{(x_h, y_h) \in \mathcal{B}_{\text{hard}} \mid y_h = y_e\}.$$

If $\mathcal{H}_{y_e}$ is non-empty, we sample (x_h, y_h) from this set; otherwise, we fall back to sampling from $\mathcal{B}_{\text{hard}}$ regardless of class. No external memory bank, dataset-level pool, or class-conditional batching is used. Although DC is our default configuration, we include SC in the main experiments to examine whether class-consistent perturbation sources provide additional benefits.

A.3 Results on Small Scale Datasets

We provide the mean and standard deviation of the models in terms of Accuracy, ECE, and Brier Score on CIFAR-10-C, CIFAR-100-C, STL-10-C, SVHN-C in Table 6.

Table 6: Performance comparison on CIFAR-10-C, CIFAR-100-C, STL-10-C, and SVHN-C. Values are reported in the format X_Y where X is the mean and Y is the standard deviation across 3 runs. For Accuracy (Acc) higher is better, while for Expected Calibration Error (ECE) and Brier Score lower is better. Best scores are shown in **bold**. * denotes statistically significant improvement over the FullySupervised Base using a paired two-sided t-test ($p < 0.05$) computed over all corruption types and severity levels.

	CIFAR-10-C			CIFAR-100-C		
Model	Acc ↑	ECE ↓	Brier ↓	Acc ↑	ECE ↓	Brier ↓
FullySupervised Base	$73.48_{0.25}$	$15.18_{0.53}$	$0.41_{0.01}$	$51.76_{0.35}$	$22.18_{0.81}$	0.68_{0}
CalibrateMix	$75.54_{0.67}$	$11.77_{0.58}$	$0.37_{0.01}$	$51.60_{0.03}$	$18.01_{0.53}$	0.65_{0}
Mixup	$75.41_{0.51}$	$11.15_{1.36}$	$0.36_{0.00}$	$51.67_{0.23}$	$18.35_{0.46}$	0.66_{0}
CutMix	$73.58_{0.59}$	$13.84_{0.96}$	$0.39_{0.01}$	$53.56_{0.18}$	$26.06_{0.36}$	0.70_{0}
ResizeMix	$73.84_{0.30}$	$12.65_{0.65}$	$0.38_{0.00}$	$54.11_{0.29}$	$25.46_{0.45}$	0.68_{0}
CutOut	$74.78_{0.55}$	$14.01_{0.47}$	$0.38_{0.01}$	$52.47_{0.45}$	$25.00_{1.46}$	0.69_{0}
SnapMix	$72.04_{1.24}$	$12.21_{1.90}$	$0.40_{0.03}$	$51.36_{0.45}$	$24.88_{1.15}$	0.71_{0}
TdAttenMix	$59.53_{0.74}$	$20.54_{0.97}$	$0.64_{0.00}$	$39.43_{0.91}$	$18.37_{0.81}$	0.81_{0}
PixCalibrate DC (Ours)	$\mathbf{84.70}^{*}_{0.21}$	$\mathbf{7.64}^{*}_{0.73}$	$\mathbf{0.22}^{*}_{0.00}$	$65.03^{*}_{0.45}$	$\mathbf{8.88}^{*}_{1.79}$	$\mathbf{0.47}^{*}_{0}$
PixCalibrate SC	$84.05^{*}_{0.07}$	$9.56^{*}_{1.40}$	$0.24^{*}_{0.00}$	$\mathbf{65.19}^{*}_{0.15}$	$10.27^{*}_{0.79}$	0.48^{*}_{0}

	STL-10-C			SVHN-C		
Model	Acc ↑	ECE ↓	Brier ↓	Acc ↑	ECE ↓	Brier ↓
FullySupervised Base	$47.29_{0.41}$	$31.41_{0.52}$	$0.79_{0.00}$	$66.00_{0.32}$	$9.59_{0.29}$	$0.42_{0.00}$
CalibrateMix	$53.28_{0.89}$	$28.67_{1.71}$	$0.72_{0.00}$	$65.24_{1.27}$	$8.93_{0.43}$	$0.43_{0.01}$
Mixup	$54.30_{2.20}$	$27.68_{2.36}$	$0.71_{0.04}$	$66.55_{0.51}$	$8.91_{0.33}$	$0.42_{0.00}$
CutMix	$42.21_{1.18}$	$40.72_{3.48}$	$0.94_{0.04}$	$60.22_{0.93}$	$8.93_{0.17}$	$0.49_{0.01}$
ResizeMix	$42.84_{0.48}$	$38.79_{1.29}$	$0.90_{0.02}$	$59.87_{1.00}$	$8.55_{0.45}$	$0.49_{0.01}$
CutOut	$44.47_{1.17}$	$36.91_{0.19}$	$0.87_{0.01}$	$62.60_{1.49}$	$9.22_{0.14}$	$0.46_{0.01}$
SnapMix	$43.67_{1.08}$	$25.84_{1.43}$	$0.81_{0.02}$	$53.94_{0.43}$	$16.95_{2.43}$	$0.60_{0.01}$
TdAttenMix	$39.93_{1.55}$	$22.70_{1.45}$	$0.82_{0.02}$	$59.81_{1.95}$	$20.19_{1.51}$	$0.52_{0.01}$
PixCalibrate DC (Ours)	$\mathbf{59.75}^{*}_{1.17}$	$21.03^{*}_{1.61}$	$0.60^{*}_{0.02}$	$\mathbf{68.17}^{*}_{0.20}$	$\mathbf{8.10}_{0.59}$	$\mathbf{0.41}_{0.00}$
PixCalibrate SC	$59.38^{*}_{0.35}$	$\mathbf{20.33}^{*}_{0.36}$	$\mathbf{0.58}^{*}_{0.00}$	$68.15^{*}_{0.14}$	$8.67_{0.05}$	$\mathbf{0.41}_{0.00}$

A.4 Additional Results on the Small-Scale Datasets Across Corruptions

For compactness and ease of comparison, we report per-corruption ECE and accuracy for CIFAR-10-C, CIFAR-100-C, STL-10-C, and SVHN-C in Tables 7 and 8, respectively. Results are reported from a representative run.

Table 7: ECE across corruption types on CIFAR-10-C, CIFAR-100-C, STL-10-C, and SVHN-C. Lower is better. Best average ECE within each dataset is shown in **bold**.

		Noise			Blur				Weather				Digital				
Dataset	Model	Gauss	Shot	Imp.	Def.	Glass	Mot.	Zoom	Snow	Frost	Fog	Bright	Cont.	Elast.	Pixel	JPEG	Avg.
CIFAR-10-C	FullySupervised	37.83	27.76	22.42	7.33	26.46	11.12	10.61	9.52	11.16	7.26	2.71	10.57	6.97	16.05	10.73	14.57
	CalibrateMix	32.68	22.95	24.90	5.55	23.37	9.41	9.30	7.41	8.13	3.05	2.44	6.26	5.09	11.59	9.46	12.11
	Mixup	31.53	22.09	27.98	6.37	25.77	9.74	8.94	7.52	7.60	2.46	2.20	5.55	4.60	11.31	8.19	12.12
	CutMix	40.68	32.29	20.17	7.92	28.55	11.05	11.07	8.46	12.62	6.66	2.43	8.28	7.91	15.75	2.84	14.45
	ResizeMix	30.85	25.44	10.77	8.86	21.45	10.43	11.86	6.50	9.77	6.15	2.29	10.85	7.25	10.95	6.54	12.00
	Cutout	35.63	26.38	22.96	9.23	24.94	12.64	13.19	8.04	10.55	6.17	2.39	10.60	7.04	15.58	9.42	14.32
	SnapMix	56.09	44.33	40.89	4.02	12.14	4.94	3.94	2.83	5.85	5.15	3.13	7.58	4.40	15.50	3.96	14.32
	TDAttenMix	20.82	15.22	23.46	15.93	3.67	7.46	9.76	10.05	6.47	15.81	18.33	15.68	15.48	19.78	10.26	13.88
	PixCalibrate DC (Ours)	15.72	13.76	10.57	5.24	7.79	8.31	6.79	4.80	3.76	4.75	2.48	6.47	5.07	8.59	6.69	**7.39**
	PixCalibrate SC	13.96	12.92	15.55	5.65	15.50	9.39	7.84	5.12	4.74	5.10	2.72	6.78	6.04	9.64	7.98	8.60
CIFAR-100-C	FullySupervised	40.82	34.39	33.43	13.64	45.70	16.06	15.90	15.07	19.16	12.88	7.30	16.73	12.73	18.60	16.17	21.24
	CalibrateMix	34.95	28.73	27.60	12.46	40.94	12.14	13.49	11.34	16.04	9.13	5.16	13.70	9.15	14.51	11.74	17.41
	Mixup	37.66	31.04	36.20	12.12	39.80	12.47	13.50	11.34	16.16	8.54	4.99	12.28	9.05	14.65	11.26	18.07
	CutMix	56.46	51.08	43.93	15.68	46.79	19.35	17.90	16.78	22.90	15.29	9.41	20.55	16.43	20.02	13.26	25.72
	ResizeMix	52.31	49.20	33.84	15.70	46.45	18.65	17.94	16.13	21.98	14.87	9.33	21.70	15.54	22.63	20.37	25.11
	Cutout	44.32	38.46	43.43	14.68	46.78	16.96	15.95	16.18	21.34	13.37	8.73	17.46	13.71	20.02	18.42	23.32
	SnapMix	51.98	46.78	60.37	7.96	44.82	14.09	9.64	12.03	17.21	10.26	4.58	16.71	10.27	23.19	23.43	23.55
	TDAttenMix	38.95	33.31	45.25	7.79	41.88	5.53	5.77	7.83	19.05	11.31	13.35	11.40	10.20	13.83	7.69	18.21
	PixCalibrate DC (Ours)	2.13	1.98	4.86	9.53	5.15	10.12	9.95	6.54	7.22	9.61	5.12	13.03	7.02	6.55	9.08	**7.19**
	PixCalibrate SC	5.15	4.88	5.54	13.38	10.84	14.15	14.29	11.29	12.34	12.60	8.23	15.61	11.11	12.59	15.41	11.16
STL-10-C	FullySupervised	39.47	35.53	38.52	50.35	43.61	40.33	28.60	25.98	29.45	26.18	10.50	28.92	29.22	36.33	12.70	31.71
	CalibrateMix	30.38	29.97	31.08	72.88	61.55	39.24	23.18	21.41	19.72	7.66	9.24	13.32	28.18	44.10	12.91	29.65
	Mixup	24.88	25.39	25.80	60.00	51.14	40.17	22.97	19.31	18.72	12.86	8.47	16.02	29.21	40.60	13.61	27.28
	CutMix	52.57	53.33	53.32	81.20	62.27	62.57	36.94	21.71	28.73	22.86	10.36	34.36	32.72	57.24	15.56	41.72
	ResizeMix	48.50	45.97	38.06	79.16	66.07	61.01	36.13	21.50	27.41	21.74	9.61	39.37	29.83	48.85	19.57	39.52
	Cutout	37.76	33.55	37.37	76.40	57.00	52.10	38.76	24.14	34.55	23.88	9.49	32.06	31.26	50.60	15.22	36.94
	SnapMix	38.15	35.77	40.07	69.64	40.64	42.85	10.84	16.20	29.54	13.41	5.08	21.64	13.33	27.37	5.28	27.32
	TDAttenMix	24.14	23.06	24.11	64.04	42.51	47.15	23.25	9.09	13.88	9.02	9.14	25.52	15.08	27.36	7.83	24.35
	PixCalibrate DC (Ours)	10.49	8.49	11.97	33.06	34.84	35.75	17.64	20.19	22.25	22.13	10.89	30.46	22.54	12.05	9.84	**20.17**
	PixCalibrate SC	8.67	6.09	10.43	36.68	37.76	36.72	18.70	18.69	20.88	23.87	10.53	32.52	22.22	12.37	10.19	20.42
SVHN-C	FullySupervised	16.81	17.83	15.55	6.23	13.22	3.13	0.42	12.28	4.73	17.37	0.57	4.85	24.69	2.77	0.85	9.42
	CalibrateMix	14.95	16.20	18.41	6.61	9.75	3.16	0.14	7.85	1.41	21.98	0.54	8.35	23.58	3.25	0.62	9.12
	Mixup	17.10	18.42	16.27	5.61	8.48	3.17	0.16	8.42	2.09	24.44	0.45	4.97	22.18	3.34	0.58	9.05
	CutMix	10.24	9.79	14.75	5.57	10.35	3.95	0.63	3.96	2.24	14.31	0.55	3.73	25.12	17.43	13.49	9.07
	ResizeMix	14.00	13.98	16.89	6.94	12.72	2.86	0.32	6.58	1.99	14.53	0.65	3.42	22.63	8.39	3.06	8.60
	Cutout	13.27	13.49	14.65	6.27	15.46	2.19	0.45	9.81	4.75	17.07	0.67	3.06	23.82	8.00	3.43	9.09
	SnapMix	8.22	8.46	5.50	22.63	25.01	8.42	4.21	9.33	9.78	28.25	4.32	8.37	28.40	44.87	13.27	15.27
	TDAttenMix	24.14	24.26	21.90	19.94	13.45	17.54	27.14	12.99	13.74	14.57	25.61	21.01	19.86	18.74	23.18	19.87
	PixCalibrate DC (Ours)	14.05	13.74	17.58	4.51	10.25	2.53	0.28	6.25	2.04	23.29	0.84	4.84	23.77	2.81	0.47	**8.48**
	PixCalibrate SC	13.14	12.94	18.61	4.29	10.74	2.97	0.33	7.01	3.13	22.31	0.55	6.17	23.66	3.57	0.54	8.66

Table 8: Accuracy (%) across corruption types on CIFAR-10-C, CIFAR-100-C, STL-10-C, and SVHN-C. Higher is better. Best average accuracy within each dataset is shown in **bold**.

		Noise			Blur				Weather				Digital				
Dataset	Model	Gauss	Shot	Imp.	Def.	Glass	Mot.	Zoom	Snow	Frost	Fog	Bright	Cont.	Elast.	Pixel	JPEG	Avg.
CIFAR-10-C	FullySupervised	42.38	55.20	55.69	83.44	51.89	78.28	77.53	82.28	76.73	86.57	94.13	75.94	84.92	74.12	78.81	73.19
	CalibrateMix	48.29	60.35	56.66	85.93	56.09	80.66	80.76	84.29	80.23	88.89	94.31	80.47	85.69	76.56	79.79	75.93
	Mixup	50.02	61.89	55.35	84.58	54.34	79.34	80.22	83.83	80.68	89.19	94.31	79.33	85.81	76.33	80.32	75.70
	CutMix	36.00	48.93	57.80	83.67	53.22	81.04	78.37	85.77	79.62	89.27	95.33	80.80	85.68	71.64	75.49	73.51
	ResizeMix	37.29	49.36	59.20	82.54	54.05	81.34	77.76	87.37	81.94	89.34	95.28	77.92	86.22	72.12	78.67	74.03
	Cutout	40.82	53.90	55.86	82.75	56.55	77.64	76.41	84.95	79.33	88.86	95.02	78.88	85.76	75.46	80.01	74.15
	SnapMix	19.58	29.66	32.46	90.62	58.91	84.14	88.18	86.73	80.04	89.41	93.85	84.78	86.50	66.73	67.78	70.62
	TDAttenMix	24.31	30.91	32.62	62.91	36.81	55.55	53.99	59.62	51.28	71.19	75.62	64.50	64.97	40.80	53.02	51.87
	PixCalibrate DC (Ours)	75.83	81.61	80.44	87.56	83.93	83.04	84.15	88.86	88.45	89.25	94.22	78.72	88.16	84.10	85.60	**84.93**
	PixCalibrate SC	72.71	78.57	82.21	87.06	83.43	81.96	83.29	88.59	87.08	89.22	94.10	79.02	87.47	82.39	84.41	84.10
CIFAR-100-C	FullySupervised	23.03	32.50	26.65	64.24	22.31	57.94	57.92	60.08	52.77	66.71	77.67	55.14	63.30	57.22	52.86	51.36
	CalibrateMix	23.78	33.57	26.32	63.22	22.64	58.27	57.41	60.63	52.98	67.24	77.81	55.71	63.51	57.22	53.06	51.56
	Mixup	21.76	31.36	25.40	63.94	22.59	57.97	58.08	60.38	52.86	67.65	77.82	56.38	63.94	57.20	54.01	51.42
	CutMix	16.03	24.00	29.25	69.48	24.83	63.63	65.04	66.19	58.35	71.90	81.02	62.90	68.01	53.41	49.14	53.55
	ResizeMix	12.17	19.32	32.85	69.62	26.10	64.68	64.95	67.28	59.42	72.36	80.74	62.61	68.86	54.87	52.01	53.86
	Cutout	23.19	32.32	24.48	65.12	22.81	59.13	60.01	60.81	53.83	67.89	77.49	57.57	64.69	57.64	53.15	52.01
	SnapMix	10.37	17.40	15.34	73.40	21.31	63.66	70.00	63.95	57.03	69.83	77.99	61.71	68.03	50.36	42.30	50.85
	TDAttenMix	7.72	13.21	15.57	56.30	9.40	48.20	50.78	46.17	36.25	58.52	66.61	50.85	52.90	38.91	42.89	39.57
	PixCalibrate DC (Ours)	49.69	55.96	57.65	69.21	59.49	65.23	66.00	69.73	66.55	68.81	78.25	57.78	70.12	69.52	63.63	64.51
	PixCalibrate SC	49.11	56.17	71.21	68.66	58.20	64.90	64.88	69.36	66.20	70.15	78.41	60.73	69.67	66.65	61.13	**65.02**
STL-10-C	FullySupervised	38.98	44.85	39.42	16.17	23.31	31.81	52.74	58.29	50.11	52.84	81.22	38.75	56.45	49.87	78.04	47.52
	CalibrateMix	46.70	48.10	46.98	13.83	21.40	39.83	60.36	65.72	65.36	77.08	84.16	65.59	55.66	40.28	75.80	53.79
	Mixup	51.00	51.15	51.20	18.67	23.04	39.23	59.21	66.18	64.50	73.88	84.57	65.70	54.28	41.65	74.73	54.60
	CutMix	29.28	32.03	30.99	11.15	17.76	23.37	47.30	64.87	54.12	60.59	83.17	43.46	55.69	25.48	59.20	42.56
	ResizeMix	24.38	26.27	27.06	10.91	16.18	23.72	46.65	65.92	55.49	59.53	84.15	39.34	58.31	33.18	65.74	42.46
	Cutout	35.11	40.84	35.67	10.77	18.24	25.91	41.92	60.50	47.11	54.94	82.94	38.01	54.63	31.76	73.24	43.44
	SnapMix	31.89	33.56	31.30	13.37	22.58	24.00	55.81	52.12	42.04	56.41	75.78	45.81	56.89	45.70	64.42	43.45
	TDAttenMix	27.46	28.80	27.50	12.41	18.67	19.87	36.92	51.73	45.64	53.50	65.67	40.45	47.39	33.57	63.71	38.22
	PixCalibrate DC (Ours)	64.64	65.31	61.11	34.91	42.10	41.68	70.13	67.55	62.53	56.83	80.90	38.96	65.38	79.88	83.63	**61.04**
	PixCalibrate SC	61.66	62.44	58.99	33.86	40.38	42.60	69.73	68.80	64.38	56.78	81.36	38.83	65.68	79.47	83.27	60.55
SVHN-C	FullySupervised	52.94	50.96	56.95	53.49	45.46	72.27	97.85	65.92	72.87	36.68	94.21	71.87	36.19	86.09	93.53	65.82
	CalibrateMix	54.16	51.48	56.71	54.14	48.85	73.03	97.82	68.61	75.40	36.76	94.34	72.69	37.11	86.01	93.52	66.71
	Mixup	56.05	53.75	58.38	53.74	48.99	72.56	97.83	68.09	75.45	36.53	94.29	72.30	37.27	88.02	93.79	67.14
	CutMix	43.46	42.00	53.06	51.14	44.73	70.12	97.18	67.43	71.59	36.48	93.27	67.81	36.82	42.56	82.71	60.02
	ResizeMix	31.55	30.06	48.74	53.61	45.92	72.62	97.96	69.39	69.93	36.61	94.56	70.93	37.60	38.37	89.16	59.13
	Cutout	51.48	49.27	58.16	54.41	47.33	72.95	97.87	69.43	73.34	36.97	94.81	70.21	36.68	57.01	92.77	64.18
	SnapMix	20.27	19.23	28.47	53.49	44.04	75.93	98.00	61.84	63.01	38.51	95.25	80.27	39.27	27.05	57.17	53.45
	TDAttenMix	35.79	34.90	47.73	52.27	46.96	74.64	97.27	70.14	70.36	36.17	94.01	71.10	37.13	39.19	88.89	59.77
	PixCalibrate DC (Ours)	58.09	56.31	62.87	54.06	49.62	72.51	97.65	71.16	79.29	34.43	94.07	71.40	36.74	89.69	93.87	**68.12**
	PixCalibrate SC	56.36	54.78	63.23	53.76	49.58	73.45	97.70	70.69	79.23	33.65	94.44	70.77	37.39	90.60	94.15	67.99

A.5 Additional Results on the Large Scale Dataset Across Corruptions

We provide our experimental results on the corrupted test set of Imagenet-C. Table 9 reports the ECE and Table 10 reports the accuracy.

Table 9: ECE on ImageNet-C across corruption types. Lower is better. Best scores are shown in **bold**, second best scores are shown with underline.

	Noise			Blur				Weather				Digital				
Model	Gauss	Shot	Imp.	Def.	Glass	Mot.	Zoom	Snow	Frost	Fog	Bright	Cont.	Elast.	Pixel	JPEG	Avg ECE
FullySupervised Base	25.24	25.51	28.78	23.29	25.76	21.66	23.17	26.85	23.30	18.99	10.79	25.89	20.00	19.88	18.11	22.48
PixCalibrate DC, β=0.0	18.81	18.25	18.60	17.32	19.09	17.53	17.38	18.08	15.91	15.36	7.42	21.16	13.51	12.64	14.15	<u>16.35</u>
PixCalibrate DC (Ours), β=0.1	20.74	19.99	20.59	13.12	18.13	14.61	16.93	19.19	15.84	13.65	6.47	17.84	15.31	10.74	14.75	**15.86**
PixCalibrate SC	24.13	24.20	24.76	21.22	23.59	21.84	22.69	24.22	20.08	19.00	9.89	23.67	17.27	15.43	17.20	20.61

Table 10: Accuracy (%) on ImageNet-C across corruption types. Higher is better. Best scores are shown in **bold**, second best scores are shown with underline.

	Noise			Blur				Weather				Digital				
Model	Gauss	Shot	Imp.	Def.	Glass	Mot.	Zoom	Snow	Frost	Fog	Bright	Cont.	Elast.	Pixel	JPEG	Avg Acc
FullySupervised Base	17.12	16.25	13.33	22.27	20.47	26.94	25.64	22.54	25.77	33.83	54.85	25.34	35.53	28.69	35.33	26.93
PixCalibrate DC, β=0.0	21.79	21.72	19.67	27.01	27.58	29.98	30.33	27.17	31.72	35.84	56.33	27.21	41.24	43.05	38.64	31.95
PixCalibrate DC, β=0.1	22.82	22.72	21.09	28.33	28.31	31.23	29.90	26.75	31.54	38.08	55.68	30.46	40.11	44.13	36.80	<u>32.53</u>
PixCalibrate SC	23.41	23.06	21.30	28.38	28.58	31.24	30.21	27.12	32.93	38.40	57.96	30.57	41.45	43.66	39.97	**33.22**

A.6 Additional Experimental Analyses

We provide additional experiments to further evaluate PixCalibrate from different perspectives. Specifically, we analyze performance across different corruption families and after excluding pixel-level corruptions in section A.6.1, isolate the effect of the confidence penalty across augmentation methods in section A.6.2, and study sensitivity to the AUM warmup duration in section A.6.3. We use the same hyperparameters across all datasets unless otherwise specified. A detailed hyperparameter analysis is provided in section A.6.4.

A.6.1 Performance Across Corruption Families

We further investigate whether PixCalibrate's improvements are primarily driven by noise-like or pixel-level corruptions. We group the 15 CIFAR-10-C corruptions into four families. *Noise* contains Gaussian noise, shot noise, and impulse noise; *Blur* contains defocus blur, glass blur, motion blur, and zoom blur; *Weather* contains snow, frost, fog, and brightness; and *Digital* contains contrast, elastic transform, pixelate, and JPEG compression.

In addition, we define a pixel-level subset consisting of Gaussian noise, shot noise, impulse noise, glass blur, and pixelate. We exclude these five corruptions and separately average performance over the remaining ten corruptions: defocus blur, motion blur, zoom blur, snow, frost, fog, brightness, contrast, elastic transform, and JPEG compression. We denote this evaluation as *Non-Pixel-Level*. The results are shown in Table 11.

Table 11: CIFAR-10-C performance across corruption families and after excluding five pixel-level corruptions. Each entry reports Accuracy / ECE. Higher Accuracy and lower ECE are better. Best results are shown in **bold**.

Method	Noise	Blur	Weather	Digital	Non-Pixel-Level
FullySupervised	51.09/29.34	72.79/13.88	84.93/7.66	78.45/11.08	81.86/8.80
Mixup	55.75/27.20	74.62/12.71	87.00/4.95	80.45/7.41	83.76/6.32
CutMix	47.58/31.05	74.08/14.65	87.50/7.54	78.40/8.70	83.50/7.92
PixCalibrate	**79.29/13.35**	**84.67/7.03**	**90.20/3.95**	**84.15/6.71**	**86.80/5.44**

PixCalibrate achieves the highest accuracy and lowest ECE across all four corruption families. More importantly, after excluding all five pixel-level corruptions, PixCalibrate improves accuracy from 83.76% for the strongest baseline, Mixup, to 86.80%, corresponding to an improvement of 3.04 percentage points. At the same time, ECE decreases from 6.32 to 5.44. These results demonstrate that the improvements of PixCalibrate are not restricted to pixel-level or noise-like corruptions, but extend

to structured distortions including blur, weather effects, contrast changes, elastic transformations, and JPEG compression.

A.6.2 Effect of the Confidence Penalty Across Augmentation Methods

To determine whether the improvements of PixCalibrate can primarily be attributed to the confidence penalty (CP), we apply the same confidence regularization to FullySupervised training, Mixup, and CutMix. Table 12 compares these methods with and without CP on CIFAR-10-C and CIFAR-100-C.

Table 12: Effect of the confidence penalty (CP) across augmentation methods on CIFAR-10-C and CIFAR-100-C. Higher Accuracy is better and lower ECE is better. Best results are shown in **bold**.

	CIFAR-10-C		CIFAR-100-C	
Method	Acc ↑	ECE ↓	Acc ↑	ECE ↓
FullySupervised	73.48	15.18	51.76	22.18
FullySupervised + CP	73.65	11.44	51.43	17.22
CutMix	73.58	13.84	53.56	26.06
CutMix + CP	72.59	12.03	51.85	24.70
Mixup	75.41	11.15	51.67	18.35
Mixup + CP	75.14	10.47	51.95	16.34
PixCalibrate − CP	83.23	8.43	63.01	10.15
PixCalibrate	**84.70**	**7.64**	**65.03**	**8.88**

The confidence penalty generally improves calibration for the baseline methods, although its effect on accuracy varies. However, PixCalibrate without CP already outperforms FullySupervised, Mixup, and CutMix, both with and without CP, in terms of both accuracy and ECE on CIFAR-10-C and CIFAR-100-C. Adding CP to PixCalibrate provides a further improvement. These results indicate that pixel-level mixing and difficulty-aware pairing provide the primary gains, while the confidence penalty serves as a complementary calibration regularizer.

A.6.3 Sensitivity to AUM Warmup Duration

Since PixCalibrate relies on AUM to separate easy-to-learn and hard-to-learn samples, we further evaluate its sensitivity to the reliability of AUM estimates during the early stages of training. Specifically, we vary the number of warmup epochs before enabling difficulty-aware pairing. The results on CIFAR-10-C are shown in Table 13.

Table 13: Sensitivity to the number of AUM warmup epochs on CIFAR-10-C from a representative run. Higher Accuracy and lower ECE are better. Best results are shown in **bold**.

Warmup Epochs	Acc ↑	ECE ↓
0	84.51	7.92
5	84.74	8.10
10	**84.93**	**7.39**
20	83.81	8.34

Performance remains relatively stable across the tested warmup durations, indicating that PixCalibrate is not highly sensitive to early AUM estimates. Even without warmup, PixCalibrate achieves 84.51% accuracy and 7.92 ECE. Among the tested settings, a 10-epoch warmup provides the best accuracy (84.93%) and ECE (7.39). Increasing the warmup duration to 20 epochs slightly degrades both metrics. We therefore use a 10-epoch warmup as the default setting in our experiments.

A.6.4 Hyperparameter Analysis

To assess the sensitivity of PixCalibrate to its hyperparameters, we conduct experiments to capture: (1) the effect of the corruption ratio ρ, (2) the effect of confidence penalty multiplier β.

Effect of the corruption ratio ρ Since ρ directly controls the extent of pixel-level perturbation during augmentation, its effect is more clearly reflected under corrupted evaluation settings. Therefore, we report the accuracy and ECE for different values of ρ in Table 14 to better capture the model's behavior under realistic perturbations. We observe that increasing ρ improves both accuracy and calibration under corruption with $\rho = 40\%$ achieving the best performance. We also observe that increasing ρ beyond 40% degrades performance. Hence, for our experiments we chose $\rho = 40\%$.

Table 14: Hyperparameter Analysis.

Dataset	Metric	$\rho = 5\%$	$\rho = 20\%$	$\rho = 40\%$	$\rho = 45\%$	$\rho = 50\%$	$\beta = 1$	$\beta = 0.01$	$\beta = 0.1$
CIFAR-10-C	Acc ↑	78.93	83.09	**84.70**	83.84	84.52	83.56	83.29	**84.70**
	ECE ↓	7.93	8.03	**7.64**	7.97	7.72	11.37	8.35	**7.64**
CIFAR-100-C	Acc ↑	60.20	64.16	**65.03**	64.18	63.74	62.71	62.63	**65.03**
	ECE ↓	11.87	10.58	**8.88**	9.08	8.90	45.76	9.93	**8.88**

Effect of the confidence penalty multiplier β We further analyze the effect of the confidence penalty multiplier β, which controls the strength of entropy regularization during training. We report the accuracy and ECE in Table 14. We observe that a moderate confidence penalty is important for achieving well-calibrated predictions.

A.7 Corruption details on the new benchmark test sets SVHN-C, STL-10-C:

This section describes the construction of the corrupted evaluation benchmarks used in our experiments, which we denote as **SVHN-C** and **STL-10-C**. These datasets are designed to measure model robustness under common, non-adversarial input degradations. Our construction follows the general benchmark philosophy of Hendrycks & Dietterich (2019), namely that robustness should be evaluated under a diverse set of realistic corruptions spanning noise, blur, weather, and digital artifacts, rather than only under clean test accuracy or worst-case adversarial perturbations.

A.7.1 Dataset Construction

For each dataset, we begin with the standard test split and apply a collection of corruption operators independently to every image. Each corruption is instantiated at five severity levels, ranging from mild to severe degradation. Thus, if a test set contains N images (the test set of SVHN (Netzer et al., 2011) contains $26,032$ images and the test set of STL-10 (Coates et al., 2011) contains $8,000$ images), then each corruption produces $5N$ corrupted samples. The samples are stored in corruption-specific arrays, where the first N entries correspond to severity level 1, the next N entries correspond to severity level 2, and so forth up to severity level 5. The class labels are repeated five times so that the label array remains aligned with this stacked format.

The same corruption-generation pipeline is used for both SVHN and STL-10. However, the implementation is adapted to support arbitrary image sizes rather than assuming a fixed ImageNet-style resolution. This is important because the two source datasets differ significantly in visual scale and content: SVHN contains 32×32 street-view digit images, whereas STL-10 contains 96×96 natural images from ten semantic object classes. Using the same corruption protocol across both datasets yields a consistent evaluation procedure, while the underlying benchmark remains dataset-specific because the source distributions, image resolutions, and semantic structures differ.

Our corrupted test sets are intended strictly for *evaluation*. They are not used during training. This follows the common-corruption benchmark setup, where the goal is to test whether a model trained on the original dataset can generalize to degraded but semantically unchanged inputs.

A.7.2 Overview of Corruption Groups

We use the standard set of 15 corruption types from the common-corruption benchmark literature. These corruptions are grouped into four broad categories:

1. **Noise corruptions:** gaussian noise, shot noise, and impulse noise.
2. **Blur corruptions:** defocus blur, glass blur, motion blur, and zoom blur.
3. **Weather corruptions:** snow, frost, fog and brightness.
4. **Digital corruptions:** contrast, elastic transform, pixelation, and JPEG compression.

These corruptions were chosen to cover a range of failure modes that are commonly encountered in real-world imaging systems. Some corruptions act directly at the pixel level, some simulate optical or sensor effects, some model environmental conditions, and others imitate distortions introduced by image storage, transmission, or geometric deformation. In addition, several corruptions contain a

stochastic component, such as random noise realization, random local shuffling, or random texture cropping, which helps avoid overly deterministic benchmark behavior.

A.7.3 Detailed Description of the 15 Corruption Types

Gaussian noise. Gaussian noise is generated by adding i.i.d. (independent and identically distributed) zero-mean Gaussian perturbations to each pixel in the RGB image. Let $x \in [0,1]^{H\times W\times 3}$ denote the normalized image. The corrupted image is obtained as

$$\tilde{x} = \mathrm{clip}(x + \epsilon, 0, 1), \qquad \epsilon \sim \mathcal{N}(0, \sigma^2 I),$$

where the standard deviation σ increases with the severity level. This corruption models sensor noise and low-light acquisition artifacts, where all pixels are perturbed independently by additive fluctuations. Because the perturbation is dense and unstructured, it tends to affect fine details, edges, and low-contrast regions throughout the image.

Shot noise. Shot noise is generated using a Poisson process. Let $x \in [0,1]^{H\times W\times 3}$ denote the normalized image and let $c > 0$ be a severity-dependent parameter. The corrupted image is obtained as

$$\tilde{x} = \mathrm{clip}\left(\frac{\mathrm{Poisson}(cx)}{c}, 0, 1\right).$$

This corruption is motivated by photon-counting noise arising from the discrete nature of light, especially in dim imaging conditions. Compared with Gaussian noise, shot noise is signal-dependent: brighter pixels exhibit different variance characteristics than darker ones. In practice, this often produces more realistic noise patterns for low-light imagery and can be particularly disruptive for small objects or thin strokes, such as the digits in SVHN.

Impulse noise. Impulse noise, also known as salt-and-pepper noise, is produced by randomly replacing a fraction of pixels with extreme values. In the implementation, the corruption amount increases monotonically with severity. This corruption approximates transmission errors, faulty sensor readings, or isolated bit flips. Unlike Gaussian or shot noise, impulse noise is sparse but high magnitude, so it can erase local evidence entirely. For small images such as those in SVHN, even a modest amount of impulse noise may obscure critical structural components of a digit.

Defocus blur. Defocus blur simulates optical blur caused by an out-of-focus lens. The corruption is implemented by convolving each image channel with a disk-shaped blur kernel. The radius of the disk increases with severity, producing progressively stronger low-pass filtering. Defocus blur removes high-frequency detail in a structured way and tends to soften object boundaries. In natural images, this can eliminate texture cues; in digit images, it can blur thin strokes and reduce the separability of nearby shapes.

Glass blur. Glass blur is a more complex blur model intended to resemble the visual effect of viewing a scene through textured or frosted glass. The implementation first applies Gaussian smoothing, then locally shuffles pixels by randomly swapping nearby locations, and finally applies Gaussian smoothing again. This produces a corruption that combines spatial displacement with blur. Unlike ordinary defocus blur, glass blur does not merely remove detail; it also distorts local structure, making edges appear warped or scrambled. This corruption is often harder than simple blur because it disrupts both texture and local geometric consistency.

Motion blur. Motion blur is generated using directional blur with a randomly sampled angle. In the implementation, the image is encoded and processed with a motion-blur operation parameterized by a blur radius and standard deviation, both of which increase with severity. The random orientation introduces variation across images, reflecting the fact that motion in real scenes is not aligned to a fixed axis. Motion blur approximates the effect of rapid camera movement or object motion during exposure. For object-centric images, it smears boundaries along one direction; for digits, it may elongate or connect strokes in ways that alter the perceived class.

Zoom blur. Zoom blur is generated by averaging multiple progressively zoomed-in versions of the same image with the original image. Operationally, the image is repeatedly center-cropped and rescaled at several zoom factors, and the resulting images are averaged together. The range of zoom factors expands with severity. This corruption is intended to mimic the effect of rapid forward camera motion or optical zoom during capture. In contrast to directional motion blur, zoom blur is radially structured and tends to smear image content outward from the center. It is especially disruptive when the class identity depends on small localized features.

Snow. Snow corruption is synthesized by generating a noisy snow layer, enlarging it, thresholding it to retain bright flakes, and applying directional motion blur so that the snow appears streaked, as in falling precipitation. This snow layer is then blended with an adjusted version of the original image that increases brightness in low-visibility regions. The corruption therefore combines partial occlusion, brightness change, and structured overlay noise. At higher severities, the density and visibility of snow increase substantially, making the scene progressively more obstructed. This corruption can be challenging because it both covers the image and changes the effective appearance of visible regions.

Frost. Frost corruption is generated by overlaying an external frost texture onto the image. A frost image is randomly selected from a small set of frost patterns, randomly cropped to the appropriate size, and linearly blended with the original image. The mixing coefficients vary with severity, increasing the prominence of the frost layer. This corruption approximates ice crystals or frost formation on a camera lens or window. Relative to snow, frost behaves more like a semi-transparent textured mask, often creating irregular high-frequency overlays that interfere with edge perception and color consistency.

Fog. Fog corruption is generated by creating a low-frequency fractal map using the diamond-square, or plasma-fractal, algorithm and then blending this map into the image. The effect is to add a haze-like veil that reduces local contrast and obscures distant or low-contrast structures. Unlike snow and frost, fog does not primarily create discrete occluders; instead, it degrades visibility through a smooth atmospheric layer. In the implementation, the fractal map is scaled to the image size and combined with the image using severity-dependent parameters. This makes fog particularly effective at suppressing weak contours and fine spatial detail.

Brightness. Brightness corruption is implemented in HSV color space by increasing the value channel while clipping to the valid range. This simulates global illumination change, such as brighter daylight or overexposure. Unlike contrast changes, brightness shifts the overall luminance level without directly altering local geometry. Nevertheless, strong brightness increases can wash out informative structure and reduce the visibility of edges or dark regions. For digit images, excessive brightness may erase stroke-background contrast; for natural images, it can saturate reflective regions and flatten local detail.

Contrast. Contrast corruption rescales each pixel's deviation from the per-image channel mean. Let μ denote the mean RGB value of the image. The corrupted image is formed by

$$\tilde{x} = \mathrm{clip}\big((x - \mu)\alpha + \mu,\, 0,\, 1\big),$$

where α decreases as severity increases. This corruption leaves the mean appearance roughly unchanged while compressing the dynamic range of local intensity differences. As a result, shapes and edges remain present but become visually weaker. Contrast reduction is especially important as a robustness test because many classifiers rely on strong edge or texture cues that may disappear when contrast is reduced.

Elastic transform. Elastic transform introduces non-rigid geometric deformation. In the implementation, the image first undergoes a small random affine perturbation and is then warped according to smooth displacement fields generated by Gaussian-filtered random noise. The resulting deformation stretches or compresses local regions while preserving overall continuity. This corruption is widely used to simulate natural non-rigid distortions and is particularly relevant for handwritten or printed symbols, where local stroke deformation can change the apparent shape of a character. On natural images, elastic transform perturbs object geometry without changing the semantic scene composition.

Pixelation. Pixelation is generated by downsampling the image to a lower spatial resolution and then upsampling it back to the original size using box interpolation. The downsampling factor becomes more aggressive with higher severity. This process removes high-frequency detail while preserving coarse global structure, thereby mimicking the appearance of low-resolution acquisition or digital zoom. Pixelation can be especially harmful to models that depend heavily on fine texture or boundary precision. In SVHN, it may destroy thin digit strokes; in STL-10, it may suppress informative object texture.

JPEG compression. JPEG compression is produced by re-encoding the image at progressively lower quality factors. This introduces block artifacts, ringing, and loss of fine detail, especially around edges and textured regions. Because JPEG compression is a common artifact in practical imaging pipelines, it serves as a particularly realistic test of robustness to storage and transmission effects. As the quality factor decreases with severity, the corruption becomes increasingly visible and can interfere with both local texture recognition and object boundary localization.

A.7.4 Role of Severity Levels

Each corruption is evaluated at five severity levels. These levels are corruption-specific, meaning that the numerical parameters controlling the corruption differ by corruption type, but are always chosen so that larger severity corresponds to stronger degradation. This design reflects the fact that different corruptions vary in how quickly they become visually destructive. For example, a small increase in Gaussian noise standard deviation has a different perceptual effect than a small decrease in JPEG quality or a small increase in disk blur radius.

Using multiple severity levels is important for two reasons. First, it avoids overfitting conclusions to a single arbitrary corruption strength. Second, it reveals whether a model degrades gracefully as conditions worsen, or whether performance collapses abruptly once corruption exceeds a certain threshold. In this sense, the severity axis provides a more informative picture of robustness than a single corrupted test set.

A.7.5 Shared Pipeline Across SVHN and STL-10

Although SVHN-C and STL10-C are generated using the same corruption operators and the same five-level severity protocol, the practical impact of these corruptions is not identical across datasets. SVHN images are lower resolution and contain compact digit-centric content, so blur and geometric distortions may remove class-defining information relatively quickly. In contrast, STL-10 images are larger and depict natural objects with richer texture and scene variation, so some corruptions may be tolerated differently. For this reason, the same nominal corruption severity should not necessarily be interpreted as having the same semantic difficulty across the two datasets.

Nevertheless, we intentionally keep the corruption-generation pipeline shared across datasets. This allows us to compare robustness trends under a consistent protocol rather than introducing additional confounds through dataset-specific redesign. The resulting benchmarks should therefore be understood as parallel corruption test sets constructed under the same procedure, rather than as severity-calibrated benchmarks whose levels are perceptually matched across datasets.

A.7.6 Visualization

Here, we visualize one sample each from the SVHN-C (Figure 4) and the STL-10-C (Figure 5) datasets.

A.8 Visualization of Difficulty Aware Pairing

We visualize representative CIFAR-100 training samples at the end of training in Figure 6. Our method uses difficulty-aware pairing, which splits images within each training batch into easy-to-learn and hard-to-learn groups based on their AUM values. The top row shows high-AUM samples that the model learns more easily, while the bottom row shows low-AUM samples that are harder to learn. We also report the model predictions and ground-truth labels for each image. Visually, the two groups exhibit clear differences: easy samples tend to contain more recognizable object structures, whereas hard samples are often ambiguous or visually challenging, sometimes even for humans. By pairing

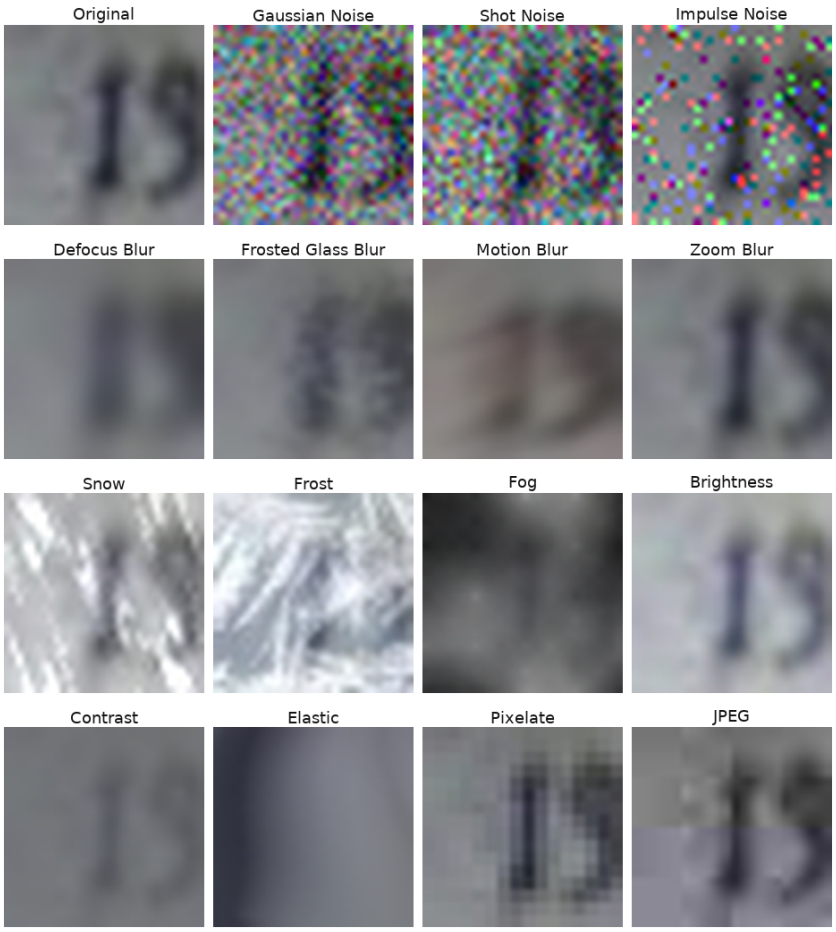


Figure 4: Visualization of a sample from the **SVHN-C** dataset of *severity level* 2

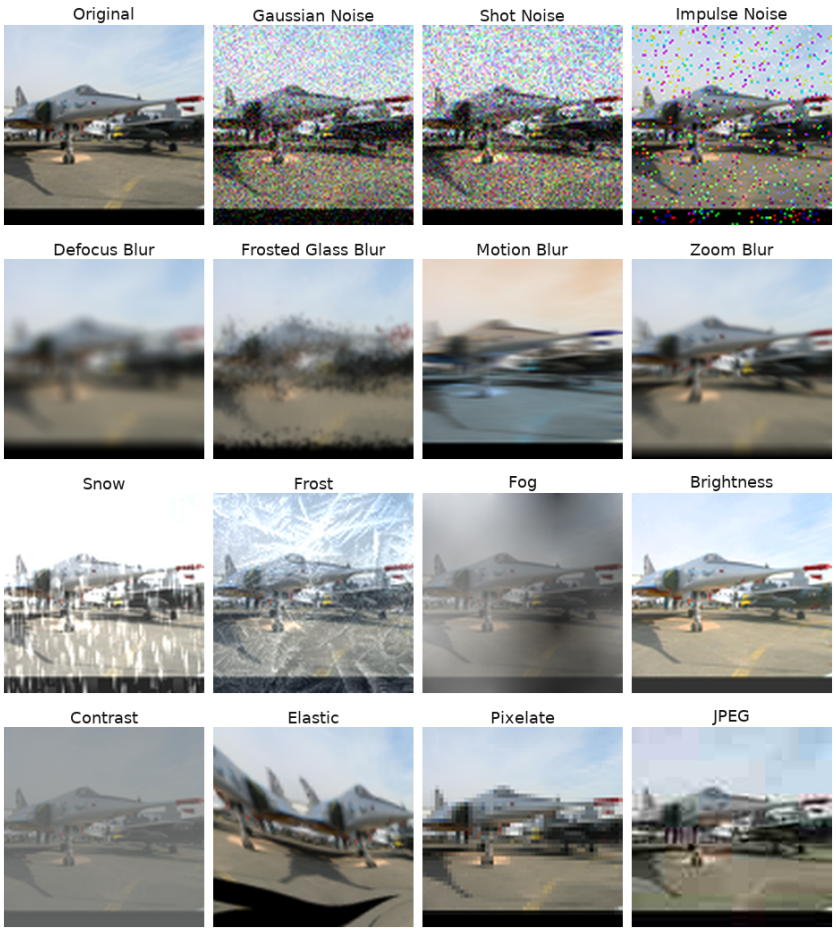


Figure 5: Visualization of a sample from the **STL-10-C** dataset of *severity level* 2

hard-to-learn images with easy-to-learn images during pixel-level mixing, our method generates challenging yet semantically grounded augmented samples that better reflect real-world corruptions.

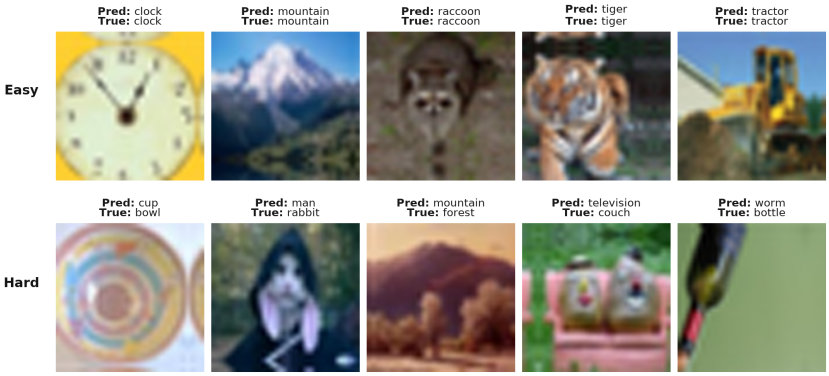


Figure 6: Visualization of sample from the **CIFAR-100** dataset based on the learning difficulty.

A.9 Computation Resources

All experiments are conducted on a machine equipped with an AMD EPYC 7V12 64-core CPU, 256 GB of DDR4 RAM (8×32 GB DIMMs), and four NVIDIA RTX A6000 GPUs, each with 48 GB of VRAM.

CIFAR-10, CIFAR-100, SVHN, and STL-10 experiments require $24 - 36$ hours to complete for a given seed for 1000 epochs. ImageNet experiments require approximately 100 hours for 300 epochs.