AUGMIX:A Simple Data Processing Method to Improve Robustness And Uncertainty

Dan Hendrycks*
DeepMind
hendrycks@berkeley.edu

Norman Mu*
Google
normanmu@google.com

Ekin D. Cubuk
Google
cubuk@google.com

Barret Zoph
Google
barretzoph@google.com

Justin Gilmer
Google
gilmer@google.com

Balaji Lakshminarayanan†
DeepMind
balajiln@google.com

ICLR 2020

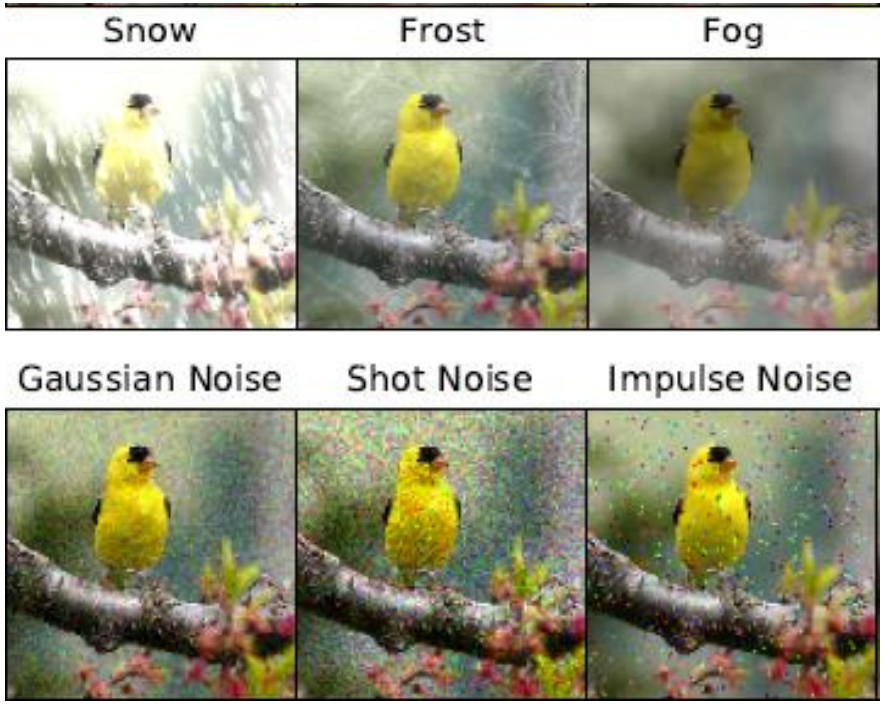
Background

Data shift

Training data

corruptions

Testing data



- Models do not robustly generalize across shifts in the data distribution.

Motivation

Data augmentation: Improver the accuracy of the model

- Mixup
- CutMix

....

$$\hat{x} = \lambda x_i + (1-\lambda)x_j,$$
$$\hat{y} = \lambda y_i + (1-\lambda)y_j,$$

Codes of Mixup

```
# y1, y2 should be one-hot vectors
for (x1, y1), (x2, y2) in zip(loader1, loader2):
    lam = numpy.random.beta(alpha, alpha)
    x = Variable(lam * x1 + (1. - lam) * x2)
    y = Variable(lam * y1 + (1. - lam) * y2)
    optimizer.zero_grad()
    loss(net(x), y).backward()
    optimizer.step()
```

Problem: previous methods never consider the robustness of the model

Method

Propose a data proposing method to improve robustness and accuracy of the model

Augmentations → Mix → Training

Algorithm AUGMIX Pseudocode

Input: Model $\hat{p}$, Classification Loss $\mathcal{L}$, Image x_{orig}, Operations $\mathcal{O} = \{\text{rotate}, \ldots, \text{posterize}\}$
function AugmentAndMix(x_{orig}, $k = 3, \alpha = 1$)
  Fill x_{aug} with zeros
  Sample mixing weights $(w_1, w_2, \ldots, w_k) \sim \text{Dirichlet}(\alpha, \alpha, \ldots, \alpha)$
  for $i = 1, \ldots, k$ **do**
    Sample operations $\text{op}_1, \text{op}_2, \text{op}_3 \sim \mathcal{O}$
    Compose operations with varying depth $\text{op}_{12} = \text{op}_2 \circ \text{op}_1$ and $\text{op}_{123} = \text{op}_3 \circ \text{op}_2 \circ \text{op}_1$
    Sample uniformly from one of these operations chain $\sim \{\text{op}_1, \text{op}_{12}, \text{op}_{123}\}$
    $x_{\text{aug}} \mathrel{+}= w_i \cdot \text{chain}(x_{\text{orig}})$ ▷ *Addition is elementwise*
  end for
  Sample weight $m \sim \text{Beta}(\alpha, \alpha)$
  Interpolate with rule $x_{\text{augmix}} = m x_{\text{orig}} + (1 - m) x_{\text{aug}}$
  return x_{augmix}
end function
$x_{\text{augmix1}} = \text{AugmentAndMix}(x_{\text{orig}})$ ▷ *$x_{augmix1}$ is stochastically generated*
$x_{\text{augmix2}} = \text{AugmentAndMix}(x_{\text{orig}})$ ▷ *$x_{augmix1} \neq x_{augmix2}$*
Loss Output: $\mathcal{L}(\hat{p}(y \mid x_{\text{orig}}), y) + \lambda \, \text{Jensen-Shannon}(\hat{p}(y \mid x_{\text{orig}}); \hat{p}(y|x_{\text{augmix1}}); \hat{p}(y|x_{\text{augmix2}}))$

Augmentations

Mix

Method

Augmentations

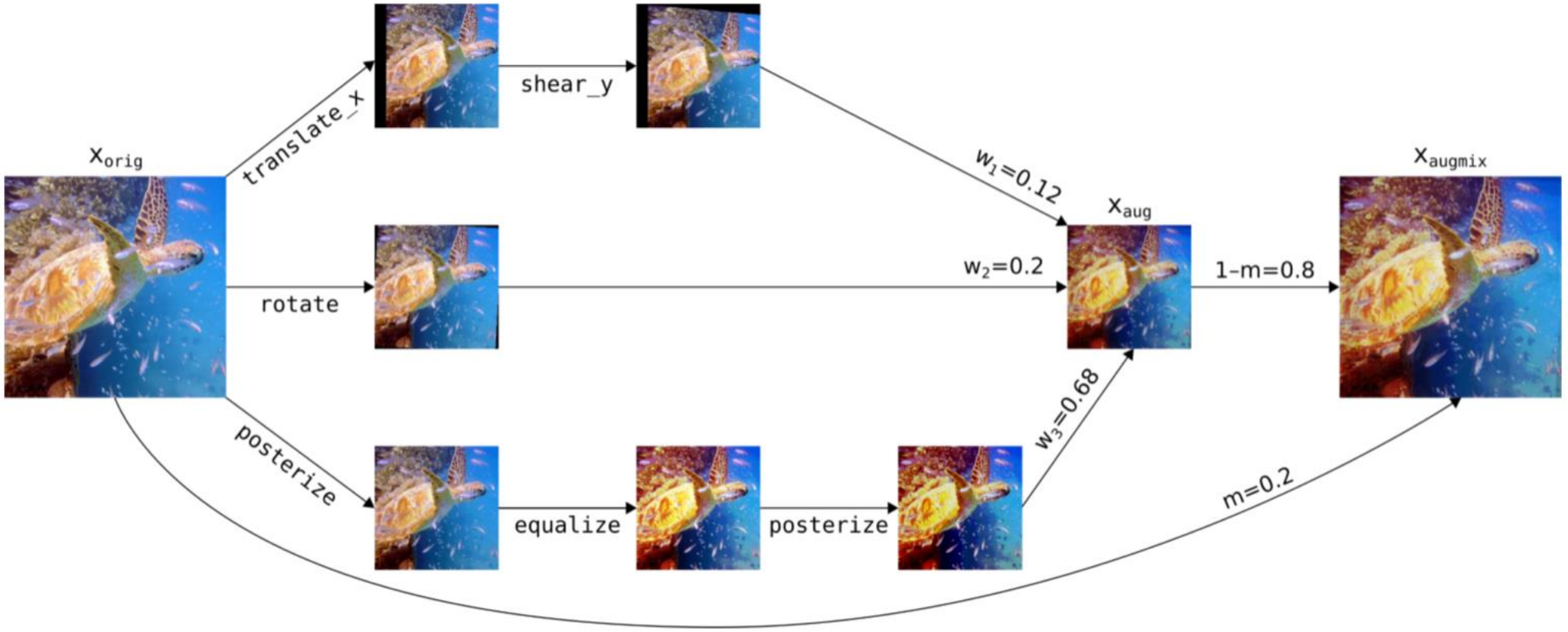


Figure 4: A realization of AUGMIX. Augmentation operations such as translate_x and weights such as m are randomly sampled. Randomly sampled operations and their compositions allow us to explore the semantically equivalent input space around an image. Mixing these images together produces a new image without veering too far from the original.

Method

Propose a data proposing method to improve robustness and accuracy of the model

Augmentations → Mix → Training

Algorithm AUGMIX Pseudocode

Input: Model $\hat{p}$, Classification Loss $\mathcal{L}$, Image x_{orig}, Operations $\mathcal{O} = \{\text{rotate}, \ldots, \text{posterize}\}$
function AugmentAndMix(x_{orig}, $k = 3, \alpha = 1$)
  Fill x_{aug} with zeros
  Sample mixing weights $(w_1, w_2, \ldots, w_k) \sim \text{Dirichlet}(\alpha, \alpha, \ldots, \alpha)$
  for $i = 1, \ldots, k$ **do**
    Sample operations $\text{op}_1, \text{op}_2, \text{op}_3 \sim \mathcal{O}$
    Compose operations with varying depth $\text{op}_{12} = \text{op}_2 \circ \text{op}_1$ and $\text{op}_{123} = \text{op}_3 \circ \text{op}_2 \circ \text{op}_1$
    Sample uniformly from one of these operations chain $\sim \{\text{op}_1, \text{op}_{12}, \text{op}_{123}\}$
    $x_{\text{aug}} \mathrel{+}= w_i \cdot \text{chain}(x_{\text{orig}})$ ▷ *Addition is elementwise*
  end for
  Sample weight $m \sim \text{Beta}(\alpha, \alpha)$
  Interpolate with rule $x_{\text{augmix}} = m x_{\text{orig}} + (1 - m) x_{\text{aug}}$
  return x_{augmix}
end function
$x_{\text{augmix1}} = \text{AugmentAndMix}(x_{\text{orig}})$ ▷ *$x_{augmix1}$ is stochastically generated*
$x_{\text{augmix2}} = \text{AugmentAndMix}(x_{\text{orig}})$ ▷ *$x_{augmix1} \neq x_{augmix2}$*
Loss Output: $\mathcal{L}(\hat{p}(y \mid x_{\text{orig}}), y) + \lambda \, \text{Jensen-Shannon}(\hat{p}(y \mid x_{\text{orig}}); \hat{p}(y|x_{\text{augmix1}}); \hat{p}(y|x_{\text{augmix2}}))$

Augmentations

Mix

Method

Jensen-Shannon Divergence Consistency Loss

Since the *semantic content of an image is approximately preserved* with AUGMIX, we should like the model to embed x_{orig}, $x_{augmix1}$, $x_{augmix2}$ similarly.

$$p_{\text{augmix1}} = \hat{p}(y \mid x_{\text{augmix1}}), p_{\text{augmix2}} = \hat{p}(y|x_{\text{augmix2}})$$

Loss function: $$\mathcal{L}(p_{\text{orig}}, y) + \lambda \, \mathbf{JS}(p_{\text{orig}}; p_{\text{augmix1}}; p_{\text{augmix2}}).$$

$$\mathbf{JS}(p_{\text{orig}}; p_{\text{augmix1}}; p_{\text{augmix2}}) = \frac{1}{3}\Big(\mathbf{KL}[p_{\text{orig}} \| M] + \mathbf{KL}[p_{\text{augmix1}} \| M] + \mathbf{KL}[p_{\text{augmix2}} \| M]\Big).$$

$$M = (p_{\text{orig}} + p_{\text{augmix1}} + p_{\text{augmix2}})/3$$

Experiment

Datasets:

training data:

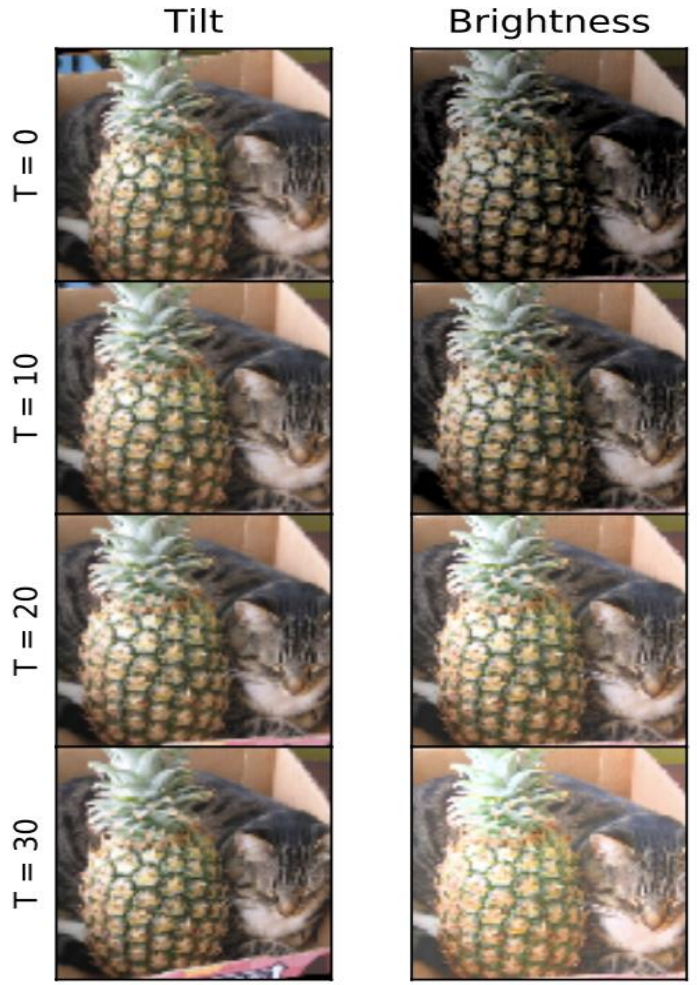
CIFAR-10, CIFAR-100, ImageNet

testing data:

CIFAR-10-C, CIFAR-100-C, ImageNet-C
(15 noise, blur, weather, and digital
corruption types)

CIFAR-10-P, CIFAR-100-P, ImageNet-P
(having perturbation sequences generated from each validation image)

ImageNet perturbation



Calibration Metrics

In order to assess a model's uncertainty estimates, we measure its miscalibration:

RMS Calibration Error: squared difference between the accuracy at a given confidence level and actual the confidence level.

$$\sqrt{\mathbb{E}_C[(\mathbb{P}(Y = \hat{Y}|C = c) - c)^2]}$$

Due to the finite size of empirical test sets, the RMS Calibration Error must be partitioning all n test set examples into b contiguous bins {B1, B2, . . . , Bb} ordered by prediction confidence. In this work we use bins which contain 100 predictions, so that we adaptively partition confidence scores on the interval [0, 1]

$$\sqrt{\sum_{i=1}^{b} \frac{|B_i|}{n} \left(\frac{1}{|B_i|} \sum_{k \in B_i} \mathbb{1}(y_k = \hat{y}_k) - \frac{1}{|B_i|} \sum_{k \in B_i} c_k \right)^2}.$$

Experiment

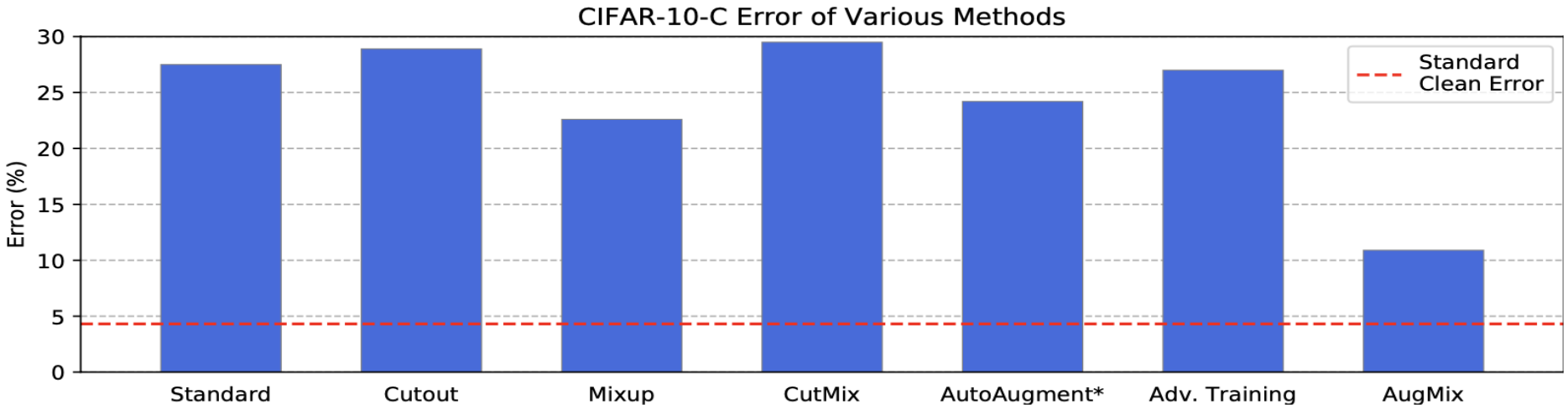
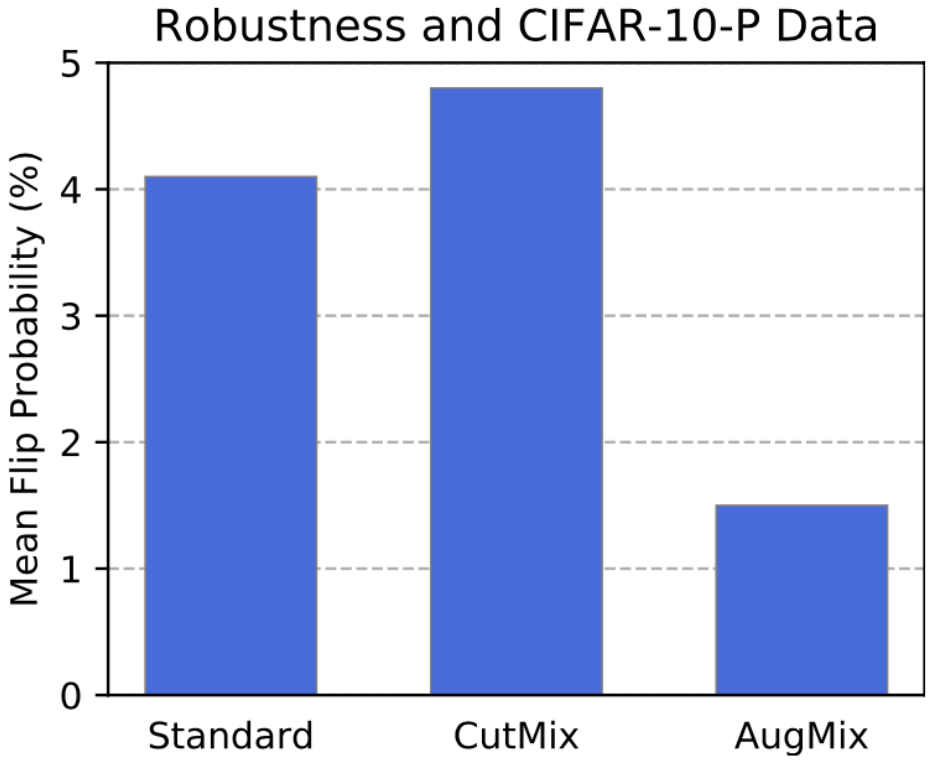


Figure 5: Error rates of various methods on CIFAR-10-C using a ResNeXt backbone. Observe that AUGMIX halves the error rate of prior methods and approaches the clean error rate.

		Standard	Cutout	Mixup	CutMix	AutoAugment*	Adv Training	AUGMIX
CIFAR-10-C	AllConvNet	30.8	32.9	24.6	31.3	29.2	28.1	**15.0**
	DenseNet	30.7	32.1	24.6	33.5	26.6	27.6	**12.7**
	WideResNet	26.9	26.8	22.3	27.1	23.9	26.2	**11.2**
	ResNeXt	27.5	28.9	22.6	29.5	24.2	27.0	**10.9**
Mean		29.0	30.2	23.5	30.3	26.0	27.2	**12.5**
CIFAR-100-C	AllConvNet	56.4	56.8	53.4	56.0	55.1	56.0	**42.7**
	DenseNet	59.3	59.6	55.4	59.2	53.9	55.2	**39.6**
	WideResNet	53.3	53.5	50.4	52.9	49.6	55.1	**35.9**
	ResNeXt	53.4	54.6	51.4	54.1	51.3	54.4	**34.9**
Mean		55.6	56.1	52.6	55.5	52.5	55.2	**38.3**

Table 1: Average classification error as percentages. Across several architectures, AUGMIX obtains CIFAR-10-C and CIFAR-100-C corruption robustness that exceeds the previous state of the art.

Experiment



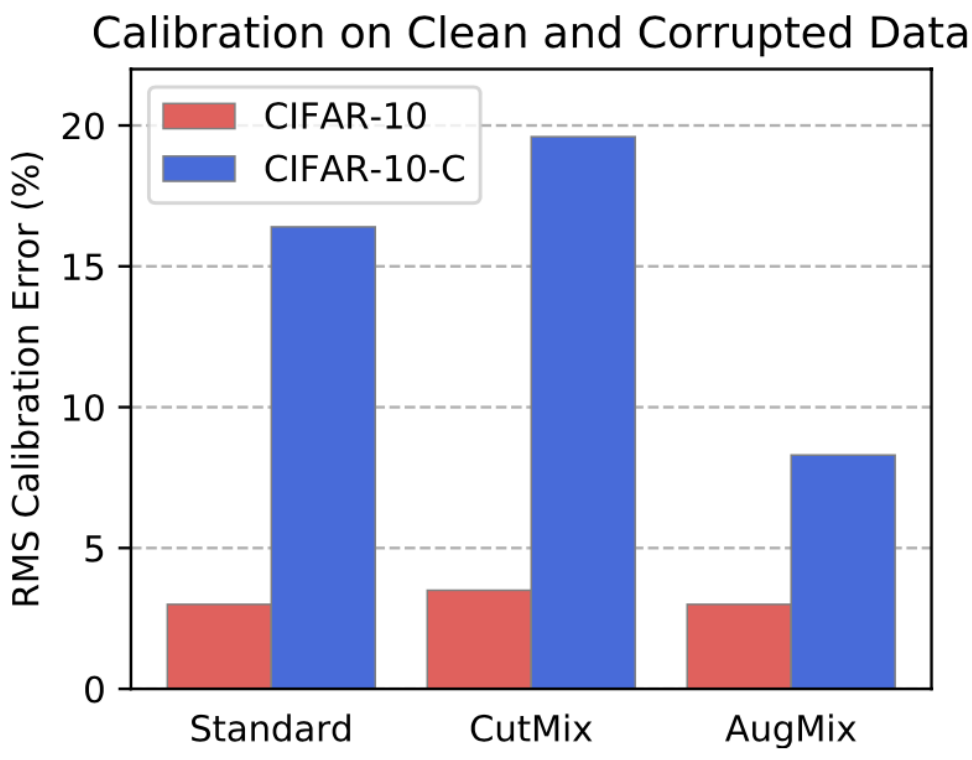


Figure 6: CIFAR-10-P prediction stability and Root Mean Square Calibration Error values for ResNeXt. AUGMIX simultaneously reduces flip probabilities and calibration error.

Experiment

		Noise			Blur				Weather				Digital				
Network	Clean	Gauss.	Shot	Impulse	Defocus	Glass	Motion	Zoom	Snow	Frost	Fog	Bright	Contrast	Elastic	Pixel	JPEG	**mCE**
Standard	23.9	79	80	82	82	90	84	80	86	81	75	65	79	91	77	80	80.6
Patch Uniform	24.5	67	68	70	74	83	81	77	80	74	75	62	77	84	71	71	74.3
AutoAugment* (AA)	22.8	69	68	72	77	83	80	81	79	75	64	**56**	70	88	**57**	71	72.7
Random AA*	23.6	70	71	72	80	86	82	81	81	77	72	61	75	88	73	72	76.1
MaxBlur pool	23.0	73	74	76	74	86	78	77	77	72	**63**	**56**	68	86	71	71	73.4
SIN	27.2	69	70	70	77	84	76	82	74	75	69	65	69	80	64	77	73.3
AUGMIX	**22.4**	65	66	67	70	80	66	**66**	75	72	67	58	58	79	69	69	68.4
AUGMIX+SIN	25.2	**61**	**62**	**61**	**69**	**77**	**63**	72	**66**	**68**	**63**	59	**52**	**74**	60	**67**	**64.9**

Table 2: Clean Error, Corruption Error (CE), and mCE values for various methods on ImageNet-C. The mCE value is computed by averaging across all 15 CE values. AUGMIX reduces corruption error while improving clean accuracy, and it can be combined with SIN for greater corruption robustness.

Experiment

		Noise		Blur		Weather		Digital				
Network	Clean	Gaussian	Shot	Motion	Zoom	Snow	Bright	Translate	Rotate	Tilt	Scale	**mFR**
Standard	23.9	57	55	62	65	66	65	43	53	57	49	57.2
Patch Uniform	24.5	**32**	**25**	50	52	54	57	40	48	49	46	45.3
AutoAugment* (AA)	22.8	50	45	57	68	63	53	40	44	50	46	51.7
Random AA*	23.6	53	46	53	63	59	57	42	48	54	47	52.2
SIN	27.2	53	50	57	72	51	62	43	53	57	53	55.0
MaxBlur pool	23.0	52	51	59	63	57	64	34	43	49	40	51.2
AUGMIX	**22.4**	46	41	**30**	**47**	38	**46**	**25**	**32**	**35**	**33**	**37.4**
AUGMIX+SIN	25.2	45	40	**30**	54	**32**	48	27	35	38	39	38.9

Table 3: ImageNet-P results. The mean flipping rate is the average of the flipping rates across all 10 perturbation types. AUGMIX improves perturbation stability by approximately 20%.

Experiment

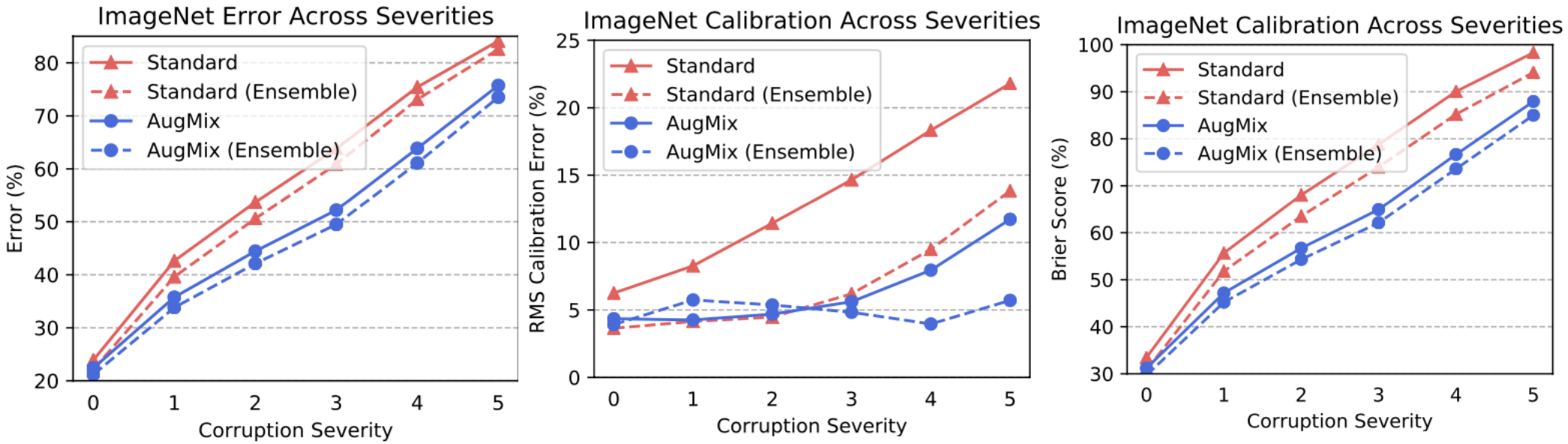


Figure 7: Uncertainty results on ImageNet-C. Observe that under severe data shifts, the RMS calibration error with ensembles and AUGMIX is remarkably steady. Even though classification error increases, calibration is roughly preserved. Severity zero denotes clean data.

Experiment

Method	CIFAR-10-C Error Rate	CIFAR-100-C Error Rate
Standard	26.9	53.3
AutoAugment*	23.9	49.6
Random AutoAugment*	17.0	43.6
Random AutoAugment* + JSD Loss	14.7	40.8
AugmentAndMix (No JSD Loss)	13.1	39.8
AUGMIX (Mixing + JSD Loss)	11.2	35.9

Table 4: Ablating components of AUGMIX on CIFAR-10-C and CIFAR-100-C. Variety through randomness, the Jensen-Shannon divergence (JSD) loss, and augmentation mixing confer robustness.

Thanks