Improving robustness against common corruptions with frequency biased models (Supplementary material)

1. Details on different corruption types

1.1. Fourier spectrum visualization

For visualizing the Fourier spectrum, we always shift low-frequency components to the center of the spectrum. In Figure 1, we visualize the Fourier spectrum of different corruption types in the ImageNet-C test set. We denote $\mathcal{F} : \mathbb{R}^{H\times W} \rightarrow \mathbb{C}^{H\times W}$ as the 2D discrete Fourier transform (DFT). Given a corruption function $C : \mathbb{R}^{H\times W} \rightarrow \mathbb{R}^{H\times W}$ which perturbs a clean image X, following Yin *et al*. [12], we plot $\mathbb{E}[|\mathcal{F}(C(X) - X)|]$. The quantity $\mathbb{E}[|\mathcal{F}(C(X) - X)|]$ is estimated over 5000 test images for each corruption type in the first severity level. We observe that noise and blur corruption types have relatively larger intensities in high-frequency regions (away from the center), compared to corruption types such as fog, frost, brightness, and contrast.

1.2. Ordering of corruption types

To visualize induced HF/LF biases, for example, in Figures 4 and 5 of the main paper, we ordered corruption types from low to high frequency. The ordering is done based on the fraction of high frequency energy in the corruption type. Given a clean image X and its corrupted version $C(X)$, the fraction of high frequency energy (F_{hf}) of the corruption can be computed as:

$$F_{hf} = \frac{||H(C(X) - X)||^2}{||C(X) - X||^2},$$

where $H(\cdot)$ represents a high-pass filter. We use a circular high-pass filter of size 56. $\mathbb{E}[F_{hf}]$ is computed over 5000 images of a given corruption type. Figure 2 shows the ordering of corruption types based on F_{hf} values.

2. Implementation details

2.1. Object classification

2.1.1 Training

We used AugMix data augmentation together with the JSD consistency loss [7]. We used the same hyperparameters as [7]. When training models from scratch we used the default augmentation operations of AugMix. The list of operations is: autocontrast, equalize, posterize, rotate, solarize, shear, translate. We used the standard 224×224 crop size for input images. For DeepAugment, we used the publicly available augmented images which were pre-computed by Hendryks *et al*. [5]. We used DeepAugment only for our large scale experiments on ImageNet.

For ImageNet-100, we trained our ResNet18 models for 75 epochs with AugMix. To train with TV regularization, we used a regularization factor $\lambda = 1e^{-5}$ for all experiments (a sensitivity analysis for λs see Figure 5). We observed that these models take longer to converge to a similar training loss as standard AugMix models. Therefore, we train these models for 150 epochs. On single GPUs, we use a batch size of 64 and an initial learning rate of 0.025 and decayed with the same schedule as [7].

For ImageNet, we used 8 Nvidia RTX 2080 Ti GPUs to train our ResNet50 models. We train models with AugMix and TV regularization for 330 epochs with a batch size of 256 and an initial learning rate of 0.1. ResNet50 models trained with AugMix are publicly available, hence we do not re-train these models. For stable distributed training, we follow recommendations of Goyal *et al*. [3] and perform a warm-up phase by training for 5 epochs. In this phase, the learning rate is linearly increased from 0 to the initial learning rate of 0.1. For training with AugMix and DeepAugment, we follow [5].

For BDD100k-cls, we finetuned our ResNet50 models (pretrained on ImageNet) for 75 epochs with a batch size of 64 and initial learning rate of 0.001.

2.1.2 Finetuning

For all datasets, to induce HF and LF robustness biases we finetuned with the relevant data augmentation operations. The AugMix approach is slightly modified to achieve this. We keep the JSD consistency loss but replace the default list of operations with either HF or LF augmentation operations to induce the required bias. We finetuned for 30 epochs with an initial learning rate of 0.001. Alternatively, one could also train from scratch with AugMix by sampling the chosen HF or LF augmentation operations with higher probabilities to induce desired biases. We found performance to be similar (see Table 1), but finetuning has lower training

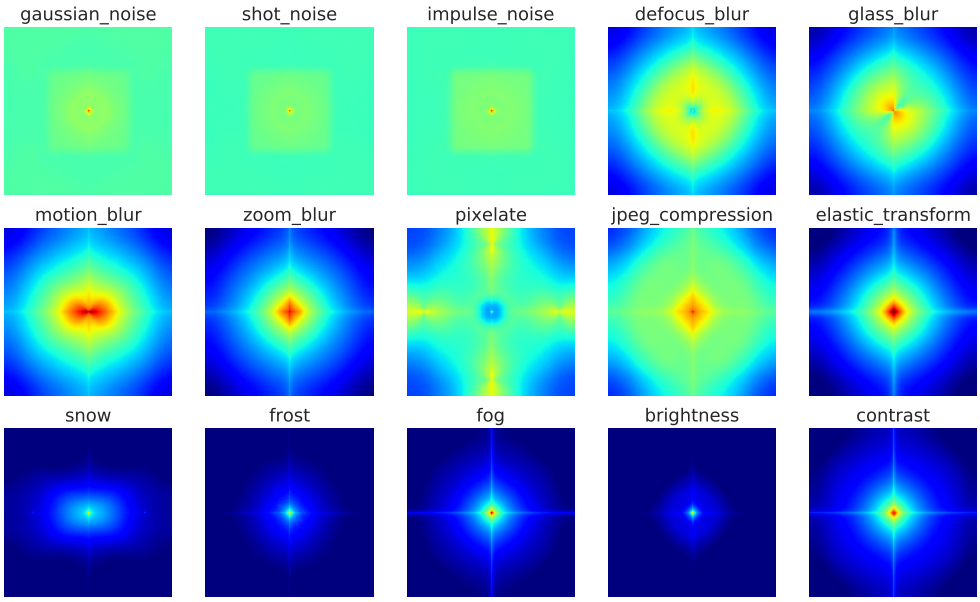


Figure 1: Visualizing Fourier spectrum of different corruption types. Given an image X and a corruption C we plot $\mathbb{E}[|\mathcal{F}(C(X) - X)|]$. $\mathcal{F}$ denotes the 2D discrete Fourier transform. The expectation is computed over 5000 test images of ImageNet-C for each corruption type. The center shows magnitudes for Fourier components with the lowest frequency. Points away from the center show magnitudes for — gradually increasing — higher frequency components. Note: the corrupted images are stored in JPEG format, therefore the visualizations can have some compression artefacts.

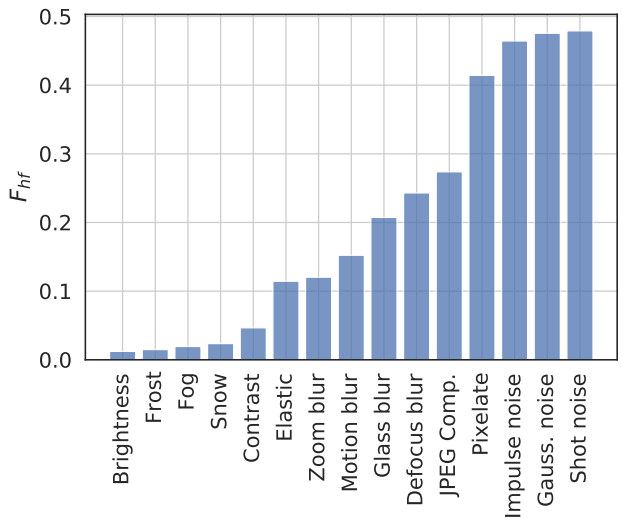


Figure 2: Corruption types and their F_{hf} (fraction of high-frequency energy).

cost and enables us to initialize with public models.

2.1.3 Combining predictions

To combine predictions for the baseline ensemble and RoHL, we always use outputs after softmax is applied.

Table 1: Robustness biases. Finetuning vs training from scratch. We show performance of ResNet18 models on ImageNet-100.

Bias	Finetune	Clean err.	mCE
HF	Yes	16.0	31.5
	No	18.2	33.6
LF	Yes	11.8	39.1
	No	12.7	38.0

2.2. Object detection

2.2.1 Training

We use the MMDetection framework [1] to train our Faster-RCNN architecture. To extract multi-scale feature representations, we used FPN (Feature pyramid networks). We trained using 8 GPUs with the default batch size and initial learning rate. The learning rate is decayed with the "1x" schedule [1]. The backbone was initialized with biased ResNet50 models finetuned on BDD100k-cls. We did not induce any further HF/LF biases during FasterRCNN's training.

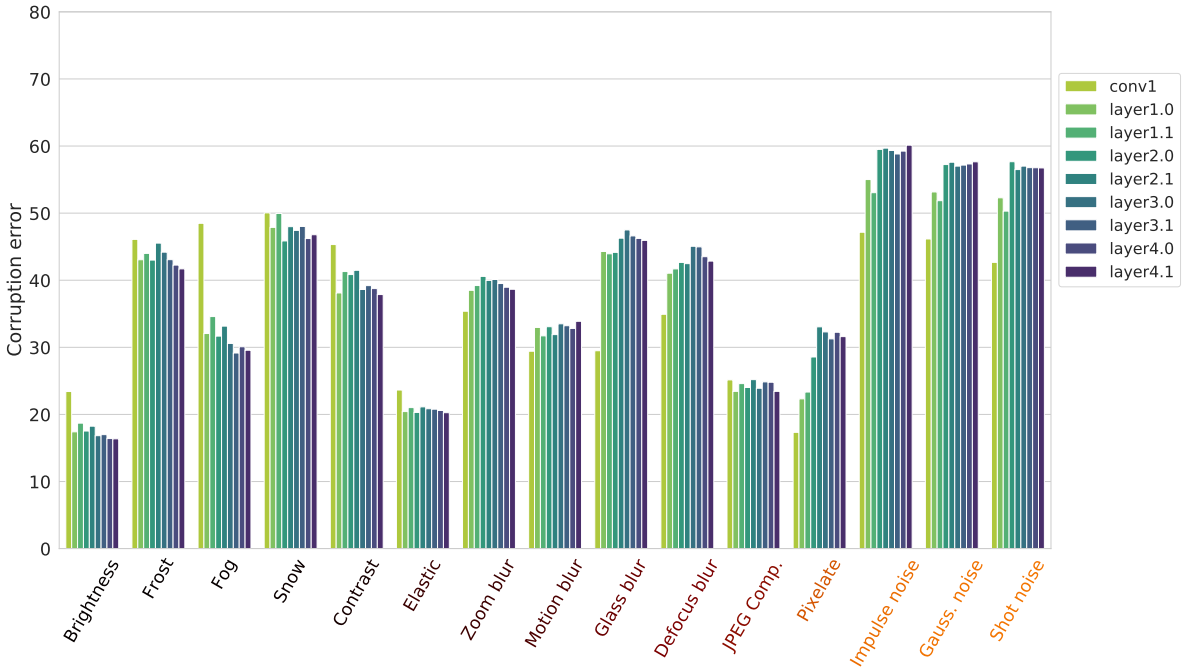


Figure 3: Impact of TV regularization applied to different layers (errors on ImageNet-C-100). Y-axis: mean error for a given corruption type over all severities. X-axis: corruption types ordered from low to high frequency. The legend on the right shows models trained with TV regularization applied to a specific layer of the ResNet18 architecture. The layer names are ordered sequentially along the network depth. We observe that applying TV regularization on conv1 — the first layer that processes the input image — leads to optimal high-frequency robustness. The effect slowly diminishes as we shift the application of TV regularzation deeper into the network.

2.2.2 Combining predictions

In addition to class probabilities, object detectors predict bounding box coordinates for each class. FasterRCNN [8] performs this in two stages. In the first stage, a region proposal (RPN) head predicts object proposals (rough bounding box estimates irrespective of the object's class) and objectness scores (probability of a proposal containing an object). After non-max suppression, these proposals are refined in the second stage (like Fast-RCNN) where the final bounding box coordinates and class probabilities are predicted. We combine the model predictions also in two stages. In the first stage, each model's object proposals and objectness scores are combined by averaging. Again, in the second stage, we average class predictions and bounding box predictions estimated by each model.

2.3. Unsupervised domain adaptation

For experiments on unsupervised domain adaptation we followed Schneider *et al*. [10] to adapt batch normalization statistics. Schneider *et al*. have shown that multiple unlabeled examples of the corruptions can be used for unsupervised adaptation. Updating the activation statistics estimated by batch normalization at training time with those of corrupted samples improves performance on ImageNet-C. Before evaluation on a corrupted test set, we used all samples to update the batch normalization statistics. Table 7. of the main paper shows results after adaptation. Note: we are able to preserve state-of-the-art performance on ImageNet-C even after adaptation.

3. Additional results on TV regularization

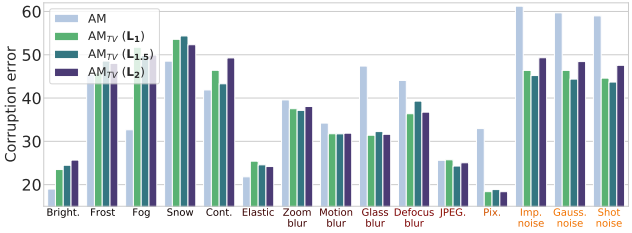


Figure 4: Comparing norms for TV reg (ImageNetC-100).

3.1. Other norms for TV

We also experimented with three values of p ($\{1, 1.5, 2\}$) in the generic L_p norm. For each p we trained with AugMix (AM) and L_p regularization. Compared to the AM baseline, all norms show similar behaviour (see Figure 4). On the targeted high-frequency corruptions, p=2 is a bit worse.

3.2. Layer-wise application of TV regularization

As we have discussed in Sec. 4.3 of the main text, standard CNN models are biased towards using high-frequency information, such as textures. Such a biased model contains filters that fire erratically whenever high-frequency information is present in the input image, resulting in large, noisy activations. This causes downstream layers — which rely on the first convolutional feature maps — to behave in unpredictable ways. We hypothesized that removing spatial outliers in the first conv feature maps will yield more stable representations and, thus, improves robustness to high-frequency corruptions. We verify this hypothesis empirically by applying this regularization to different layers of a ResNet18 architecture along the network depth. Results are shown in Figure 3. We observe that applying TV regularization to the first conv layer's activation maps leads to optimal high-frequency robustness.

3.3. Results on other architectures

Besides the ResNet family, we evaluated for two additional architectures, MNasNet_0.75 and DenseNet121. Table 2 shows results on IN-100 with the same hyperparameters as ResNet. We observe a significant decrease, similar to ResNet18 for AugMix models (see Table 1 in the main text).

Table 2: Performance of AM_{TV} with other architectures on IN-100.

Model	Arch.	clean err.	mCE
AM	MNasNet_0.75	11.8	45.2
AM_{TV}	MNasNet_0.75	11.6	39.0
AM	DenseNet121	9.8	36.6
AM_{TV}	DenseNet121	12.7	30.4

3.4. The TV regularization factor λ

The hyperparameter λ controls the strength for the TV regularization term. For all experiments in the main paper, we used a value of $1e^{-5}$. Here we study how different values of λ affect the clean and corruption error. To this end, we first sampled 50 random values for λ in the range $[1e^{-6}, 5e^{-4}]$. For each λ we trained ResNet18 models on ImageNet-100 with TV regularization. We plot the clean vs corruption error for each model in Figure 5. We observe that models trained with $\lambda \in [1e^{-6}, 9e^{-5}]$ have a good clean vs corruption error trade-off. Larger values of λ degrade both clean and corruption errors.

3.5. Effect of λ on feature maps

In Figure 6 we visualize two examples to show the effect of increasing the TV regularization factor λ that is used for training. We observe that as we increase λ during training, the most active feature map for the conv1 layer is impacted less by noise at test time. We highlight that these models were not trained with any noise augmentation.

4. Low frequency robustness bias

Figure 7 shows an experiment with an augmentation that randomly perturbs the magnitude of LF components within patch windows centered at 0-frequency in the frequency domain. Note that this is different from Yin *et al*. [12], who perturbed *all* Fourier components with magnitudes sampled from the Fog corruption. Larger windows reduce LF robustness. For a small enough window (5×5), LF robustness improves over AugMix (AM) on all LF corruption types except Snow. The proposed LF expert based on contrast augmentation is even better in terms of mCE.

5. Detailed results on ImageNet

5.1. Robustness biases of expert models

We show the clean error and mCE for the high-frequency and low-frequency expert models in Table 3. The high-

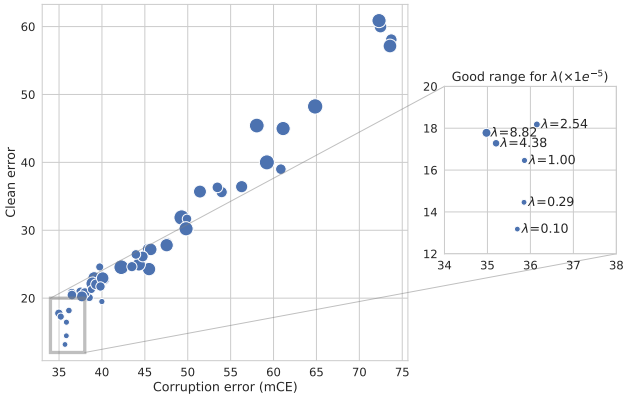
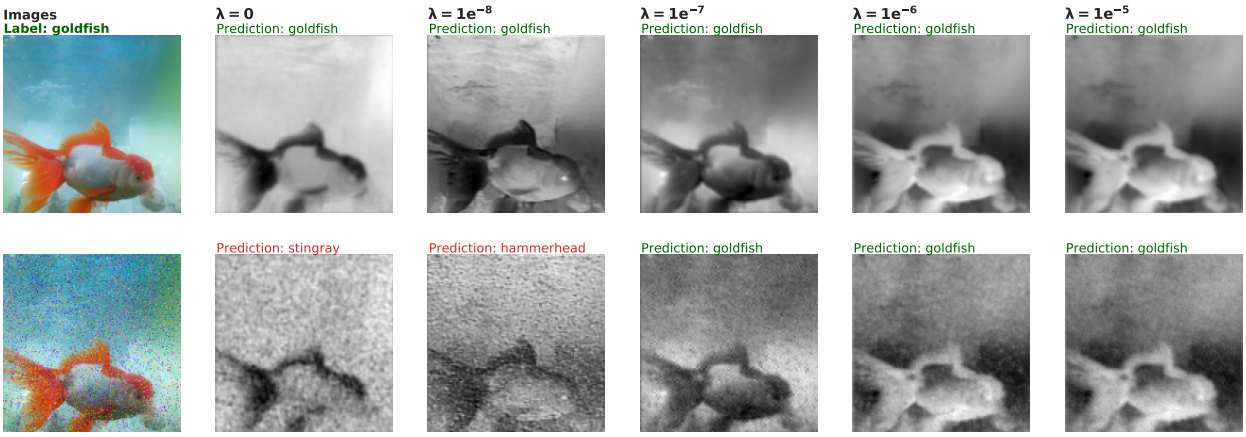
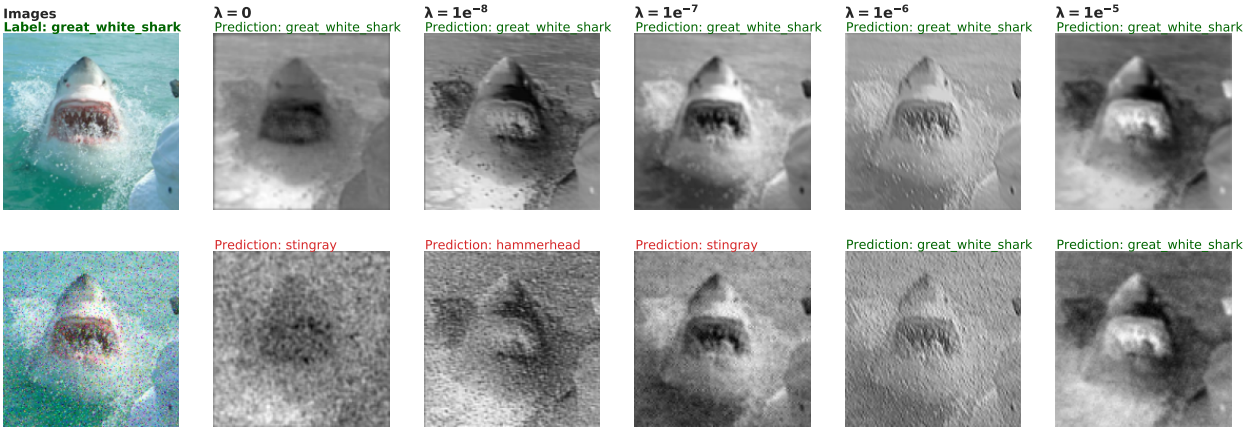


Figure 5: Clean vs corruption error for different values of λ (ImageNet-100). Each dot shows the performance of a model trained with a certain λ. Values of λ are sampled uniformly at random from the range: $[1e^{-6}, 1e^{-3}]$. The size of each dot is directly proportional to the sampled value for λ (larger dots indicate larger values of λ). Left: Shows performance of all models. Right: A closer look at models with good clean vs corruption error trade-off. We observe that models trained with smaller regularization factors ($1e^{-6} < \lambda < 9e^{-5}$) perform better.



(a) Example: goldfish



(b) Example: great white shark

Figure 6: Effect of increasing TV regularization factor (λ). In Figure 6a and Figure 6b we visualize two examples to show effect of increasing the TV regularization factor λ that is used for training. First column: clean and noisy images. Remaining columns (left to right): the most active feature map (conv1) generated after forwarding a clean and noisy *test* image to a model trained with a certain λ (shown in the column header). Larger activation values have a lighter shade, while smaller values are darker. $\lambda = 0$ means no TV regularization was used. Models with no TV regularization fire erratically for the noisy test image. Increasing λ leads to smoother activation maps. With a larger λ ($\geq 1e^{-4}$), models face convergence issues and performance deteriorates.

Table 3: Performance of HF and LF experts (ImageNet). We show the clean error and mCE for ResNet50 models trained on ImageNet. High-frequency (HF) expert is AMDA$_{\text{TV}}$-ft_{Gauss}. Low-frequency (LF) expert is AMDA-ft_{Cont}.

Model	Clean err	mCE
AMDA	24.2	53.6
AMDA-ft_{Cont}	**23.4**	52.8
AMDA$_{\text{TV}}$-ft_{Gauss}	26.4	**52.6**

frequency expert (AMDA$_{\text{TV}}$-ft_{Gauss}) was first trained with AugMix and DeepAugment with TV regularization and then finetuned on Gaussian noise and blur. The low-frequency expert was obtained by finetuning the publicly available AMDA model with contrast augmentation.

Although the the results in Table 3 does not show much difference in terms of mCE, these expert models have very different robustness biases. This is shown in Figure 8. Compared to the baseline AMDA, the high-frequency expert AMDA$_{\text{TV}}$-ft_{Gauss} improves on most high-frequency corruptions while performing worse on low-frequency corruptions. AMDA-ft_{Cont} on the other hand improves on most

low-frequency corruptions and some high-frequency corruptions (noise). These observations are similar to the small scale ablation experiments on ImageNet-100 in the main paper (Section 5.3). Also we highlight that clean error improves for the low-frequency expert AMDA-ft_{Cont} (see Table 3).

5.2. Results of RoHL variants

In Table 4 we compare performance of RoHL variants with other approaches and an ensemble of two AMDA models. We also show errors on each corruption type. The trade-off between Clean vs Corruption error is shown in Figure 9. We observe that RoHL(AMDA$_{\text{TV}}$-ft_{Gauss}, AMDA-ft_{Cont}) outperforms the baseline Ensemble(AMDA, AMDA) on all high-frequency corruption types except Motion and Zoom blur. On low-frequency corruption types our approach performs the same or better. Also, we highlight that RoHL(AMDA$_{\text{TV}}$, AMDA) also improves that state-of-the-art without any additional data augmentation.

6. Results on BDD100k

Object classification. BDD100k is an object detection dataset containing multiple object instances per image and hence cannot be directly used in the classification setting. We extracted object images for each class using 2D bounding box annotations to first transform these datasets to the standard classification setting. The transformed variants are denoted as BDD100k-cls. We finetuned our ResNet50 models (pre-trained on ImageNet) on the "clear" split of BDD100k-cls. For RoHL, we finetune with the HF and LF biases. We evaluated on corrupted test sets of BDD100k-cls. We observed that weather distortions present in BDD100k are rather benign (see Figure 10). From Table 5 we observe that the corrupted test sets do not significantly impact performance of models trained even with standard data augmentation.

Object detection. Results for object detection are shown in Table 6. We can observe that performance gap between i.i.d and OOD is marginal.

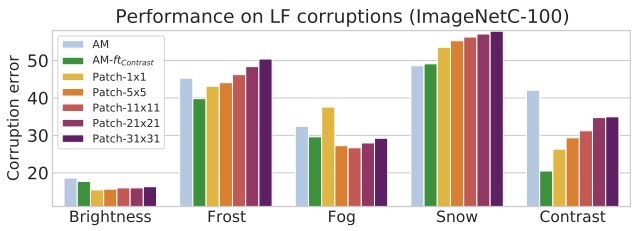


Figure 7: Performance of models trained with LF data aug. in frequency domain (randomly perturbed patch windows centered at 0-frequency). Image resolution is 224×224.

7. Results on other distribution shifts

7.1. ImageNet-R

To measure performance on non-corruption based distribution shift we evaluate RoHL on ImageNet-R. We compare to other state-of-the-art approaches and a two-member ensemble of AMDA models. We note that ImageNet-R contains a subset of 200 classes from ImageNet. Therefore to evaluate models trained on ImageNet we follow Hendryks *et al*. [5] and mask out predictions for irrelevant class indices. We do not train or finetune new models. The results are shown in Table 7. RoHL improves on i.i.d and OOD test sets but the gains are diminished.

7.2. ObjectNet

We evaluate our ResNet50 models trained on ImageNet on ObjectNet's test images. We excluded non-overlapping classes between ImageNet and ObjectNet. Results are shown in Table 8. Considering the high baseline errors, the improvements are marginal.

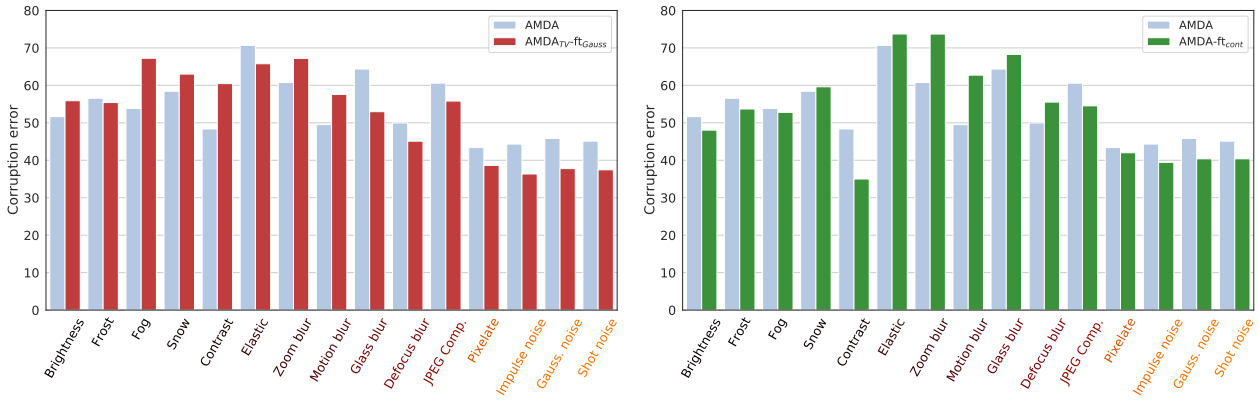
8. Dataset details

8.1. Visual examples of real image corruptions

Figure 10 shows example images of real image corruptions from BDD100k and DAWN. We can observe that corrupted images on BDD100k are mostly benign. DAWN on the other hand contains more severe samples.

8.2. ImageNet-100

The class ids for the ImageNet-100 dataset used in our ablation studies are listed below: n01443537, n01484850, n01494475, n01498041, n01514859, n01518878, n01531178, n01534433, n01614925, n01616318, n01630670, n01632777, n01644373, n01677366, n01694178, n01748264, n01770393, n01774750, n01784675, n01806143, n01820546, n01833805, n01843383, n01847000, n01855672, n01860187, n01882714, n01910747, n01944390, n01983481, n01986214, n02007558, n02009912, n02051845, n02056570, n02066245, n02071294, n02077923, n02085620, n02086240, n02088094, n02088238, n02088364, n02088466, n02091032, n02091134, n02092339, n02094433, n02096585, n02097298, n02098286, n02099601, n02099712, n02102318, n02106030, n02106166, n02106550, n02106662, n02108089, n02108915, n02109525, n02110185, n02110341, n02110958, n02112018, n02112137, n02113023, n02113624, n02113799, n02114367, n02117135, n02119022, n02123045, n02128385, n02128757, n02129165, n02129604, n02130308, n02134084, n02138441, n02165456, n02190166, n02206856, n02219486, n02226429,



(a) High frequency robustness bias

(b) Low frequency robustness bias

Figure 8: Robustness bias and its impact on performance across corruption types (ImageNet). Figure 8a and Figure 8b show corruption errors for models exhibiting high and low-frequency robustness biases, respectively. The y-axis shows the corruption error for different corruption types (averaged over severity levels) and the x-axis shows corruption types categorized into high-frequency (red text) and low-frequency (blue text). In Figure 8a, we see that AMDA$_{TV}$-ft_{Gauss} is more robust to high frequency corruptions compared to AMDA. Figure 8b shows that AMDA-ft_{Cont} improves on low-frequency corruption types. Surprisingly, it also improves performance on some noise corruptions. Comparing Figure 8a and Figure 8b, we see that these models have very different biases.

Table 4: Detailed results on ImageNet and ImageNet-C. We compare RoHL to other state-of-the-art approaches using a ResNet50 architecture. We also compare to an ensemble of two AMDA models with already improves the state-of-the-art. RoHL shows the best trade-off between clean error and mCE. Individual errors for different corruption types are also shown. Error for each corruption type is normalized by AlexNet's error [6] on that particular corruption. Therefore, values greater than 100 indicate worse performance compared to AlexNet.

				Noise			Blurs				Weather				Digital			
	Model	Clean err.	mCE	Gauss.	Shot	Impulse	Defocus	Glass	Motion	Zoom	Snow	Frost	Fog	Bright.	Cont.	Elastic	Pix.	JPEG
	Standard-R50	23.9	76.7	80	82	83	75	89	78	80	78	75	66	57	71	85	77	77
SOTA approaches	IN-21K-Pretraining	22.4	65.8	61	64	63	69	84	68	74	69	71	61	53	53	81	54	63
	SE (Self-Attention)	22.4	68.2	63	66	66	71	82	67	74	74	72	64	55	71	73	60	67
	CBAM (Self-Attention)	22.4	70.0	67	68	68	74	83	71	76	73	72	65	54	70	79	62	67
	AdversarialTraining	46.2	94.0	91	92	95	97	86	92	88	93	99	118	104	111	90	72	81
	SpeckleNoise	24.2	68.3	51	47	55	70	83	77	80	76	71	66	57	70	82	72	69
	StyleTransfer	25.4	69.3	66	67	68	70	82	69	80	68	71	65	58	66	78	62	70
	AugMix	22.5	65.3	67	66	68	64	79	59	64	69	68	65	54	57	74	60	66
	DeepAugment	23.3	60.4	49	50	47	59	73	65	76	64	60	58	51	61	76	48	67
	AugMix+DeepAugment (AMDA)	24.2	53.6	46	45	44	50	64	50	61	58	57	54	52	48	71	43	61
Ours	Ens(AMDA, AMDA)	24.0	51.9	43	42	42	48	63	49	61	57	55	53	50	46	68	42	59
	RoHL(AM$_{TV}$, AM)	**22.2**	61.1	61	61	61	60	73	56	61	66	64	60	52	55	69	55	63
	RoHL(AMDA$_{TV}$, AMDA)	23.6	49.7	41	40	39	46	57	**47**	**58**	**57**	53	**53**	49	46	**64**	38	57
	RoHL(AMDA$_{TV}$-ft_{Gauss},AMDA-ft_{Cont})	22.7	**47.9**	**36**	**35**	**34**	**45**	**55**	56	66	**57**	**50**	**53**	**47**	**35**	**64**	**36**	**50**

Table 5: Object classification performance on BDD0100k-cls with ResNet50. We show errors on weather corruptions present in the BDD100k-cls test set. Corrupted samples are mostly benign and hence do not significantly degrade performance.

Model	Clear error	mCE	Rain errors	Snow errors
Standard data augmentation	5.8	7.4	8.1	6.8
AMDA	5.3	6.5	7.1	5.8
Ensemble(AMDA, AMDA)	5.1	6.4	7.0	5.8
RoHL (AMDA$_{TV}$-ft_{Gauss},AMDA-ft_{Cont})	**5.0**	**6.2**	**6.7**	**5.6**

Table 6: Object detection performance with different ResNet50 backbones used in FasterRCNN on BDD100k. We report AP scores on the "Clear" and corrupted test splits of BDD100k. Higher AP scores are better. mAPc denotes the mean AP over corruption types.

Pretrained backbone	Clear AP	mAPc	Rain AP	Snow AP
Standard data augmentation	27.8	25.6	27.6	23.6
AMDA	27.7	25.7	27.4	23.9
Ensemble(AMDA, AMDA)	28.6	26.6	28.5	24.7
RoHL (AMDA$_{TV}$-ft_{Gauss},AMDA-ft_{Cont})	**28.7**	**26.8**	**28.6**	**25.0**

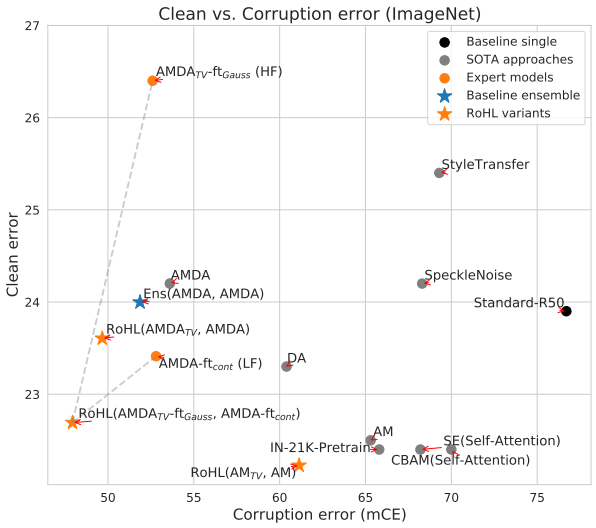


Figure 9: Clean vs. corruption error on full ImageNet. The Pareto-front shows that our approach scales well and improves the previous state of the art on ImageNet-C.

Table 7: Results on ImageNet-200 and ImageNet-R. ImageNet-200 (IN-200) uses the same 200 classes as ImageNet-R (IN-R). Here IN-200 and IN-R are the i.i.d and OOD test sets respectively. RoHL improves both i.i.d and OOD performance compared to the state-of-the-art AMDA.

	Model	IN-200 error	IN-R error
	Standard ResNet50 [4]	7.9	63.9
SOTA approaches	IN-21K-Pretrain [5]	**7.0**	62.8
	CBAM(Self-Attention) [5]	**7.0**	63.2
	AdversarialTraining [11]	25.1	68.6
	SpeckleNoise [9]	8.1	62.1
	StyleTransfer [2]	8.9	58.5
	AM [7]	7.1	58.9
	DA [5]	7.5	57.8
	AMDA [5]	8.0	53.2
Ours	Baseline Ensemble (AMDA, AMDA)	8.0	52.3
	RoHL (AMDA$_{TV}$-ft_{Gauss},AMDA-ft_{Cont})	7.5	51.6

Table 8: Errors on ObjectNet with ResNet50. Lower is better.

Model	ObjectNet (error)
Standard ResNet50	72.3
AMDA	72.4
Ensemble(AMDA, AMDA)	72.3
RoHL (AMDA$_{TV}$-ft_{Gauss},AMDA-ft_{Cont})	**70.8**

n02233338, n02236044, n02268443, n02279972, n02317335, n02325366, n02346627, n02356798, n02363005, n02364673, n02391049, n02395406, n02398521, n02410509, n02423022

References

[1] Kai Chen, Jiaqi Wang, Jiangmiao Pang, Yuhang Cao, Yu Xiong, Xiaoxiao Li, Shuyang Sun, Wansen Feng, Ziwei Liu, Jiarui Xu, Zheng Zhang, Dazhi Cheng, Chenchen Zhu, Tianheng Cheng, Qijie Zhao, Buyu Li, Xin Lu, Rui Zhu, Yue Wu, Jifeng Dai, Jingdong Wang, Jianping Shi, Wanli Ouyang, Chen Change Loy, and Dahua Lin. MMDetection: Open mmlab detection toolbox and benchmark. *arXiv*, 2019. 2

[2] Robert Geirhos, Carlos RM Temme, Jonas Rauber, Heiko H Schütt, Matthias Bethge, and Felix A Wichmann. Generalisation in humans and deep neural networks. In *NeurIPS*, 2018. 8

[3] Priya Goyal, Piotr Dollár, Ross Girshick, Pieter Noordhuis, Lukasz Wesolowski, Aapo Kyrola, Andrew Tulloch, Yangqing Jia, and Kaiming He. Accurate, large minibatch sgd: Training imagenet in 1 hour. *arXiv*, 2017. 1

[4] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 8

[5] Dan Hendrycks, Steven Basart, Norman Mu, Saurav Kadavath, Frank Wang, Evan Dorundo, Rahul Desai, Tyler Zhu, Samyak Parajuli, Mike Guo, Dawn Song, Jacob Steinhardt, and Justin Gilmer. The many faces of robustness: A critical analysis of out-of-distribution generalization. *arXiv*, 2020. 1, 6, 8

[6] Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. *ICLR*, 2019. 7

[7] Dan Hendrycks, Norman Mu, Ekin D. Cubuk, Barret Zoph, Justin Gilmer, and Balaji Lakshminarayanan. AugMix: A simple data processing method to improve robustness and uncertainty. *ICLR*, 2020. 1, 8

[8] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *NeurIPS*, 2015. 3

[9] Evgenia Rusak, Lukas Schott, Roland S. Zimmermann, Julian Bitterwolf, Oliver Bringmann, Matthias Bethge, and Wieland Brendel. A simple way to make neural networks robust against diverse image corruptions. *ECCV*, 2020. 8

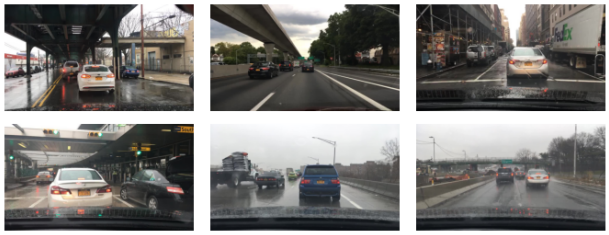
[10] Steffen Schneider, Evgenia Rusak, Luisa Eck, Oliver Bringmann, Wieland Brendel, and Matthias Bethge. Improving robustness against common corruptions by covariate shift adaptation. *NeurIPS*, 2020. 3

[11] Eric Wong, Leslie Rice, and J Zico Kolter. Fast is better than free: Revisiting adversarial training. *ICLR*, 2020. 8

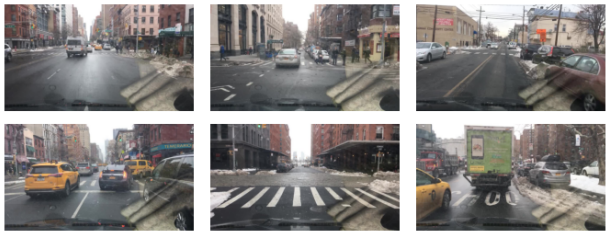
[12] Dong Yin, Raphael Gontijo Lopes, Jon Shlens, Ekin Dogus Cubuk, and Justin Gilmer. A fourier perspective on model robustness in computer vision. In *NeurIPS*, 2019. 1, 4

(a) DAWN: Rain

(b) DAWN: Snow



(c) BDD100K: Rain



(d) BDD100K: Snow

Figure 10: Example images of real image corruptions in BDD100k and DAWN. Images are randomly selected. DAWN contains more severe image corruptions and has a larger negative impact on OOD performance.