# COVARIATE SHIFT ADAPTATION FOR ADVERSARIALLY ROBUST CLASSIFIER

**Jay Nandy**[1], **Sudipan Saha**[2] **Wynne Hsu**[1] **Mong Li Lee**[1], **Xiao Xiang Zhu**[2]
[1]National University of Singapore, [2]Technical University of Munich
jaynandy@comp.nus.edu.sg, sudipan.saha@tum.de,
{whsu, leeml}@comp.nus.edu.sg, xiaoxiang.zhu@dlr.de

## ABSTRACT

We show that adaptive batch normalization (BN) technique that involves re-estimating the BN parameters during inference, can significantly improve the robustness of adversarially trained models for any random perturbations, including the Gaussian noise. This simple finding enables us to transform an adversarially trained model into a randomized smoothing classifier to provide certified robustness for $\ell_2$ norm. Moreover, we achieve $\ell_2$ certified robustness even for adversarially trained models, learned using $\ell_\infty$-bounded adversaries. Further, adaptive BN significantly improves robustness against common corruptions, without any detrimental effect on their performance against adversarial attacks. This enables us to achieve both adversarial and corruption robustness using the same classifier.

## 1 INTRODUCTION

Deep neural network (DNN) based models perform poorly in adverse real-world conditions when the test examples are obtained from a different distribution as the training examples. Recent studies have shown that a DNN classifier that correctly classifies an image $x$, can be easily fooled by an *adversarial attack* to misclassify $x + \delta$, where, $\delta$ is a minor *adversarial perturbation* such that the changes between $x$ and $x+\delta$ remain indistinguishable to the human eye (Szegedy et al., 2014). Further, DNN classifiers are also found to be sensitive to naturally occurred random corruptions(Hendrycks & Dietterich, 2019). Hence, building a robust classifier against both adversarial perturbations and common corruptions has emerged as an important research direction to enhance the reliability for sensitive real-world applications (Gilmer et al., 2019).

Among the successful defense mechanisms against adversarial attacks, adversarial training achieves the best empirical robustness performance for a specific perturbation type (such as a small $\ell_p$-noise) by training on adversarial examples of the same perturbation types (Madry et al., 2018; Tramèr & Boneh, 2019). While recently, a number of certification techniques is proposed for this frameworks to verify if the prediction of the test examples $x$ remain constant within its neighborhood (Wong & Kolter, 2018; Wang et al., 2018), they typically do not scale for large networks (e.g., ResNet50) and datasets (e.g., IMAGENET). In contrast, the randomized smoothing technique provides a scalable $\ell_2$-norm certification for any classification model that is robust against standard isotropic Gaussian noise (Cohen et al., 2019). However, this certification technique cannot be applied for adversarially trained models as they are not robust against Gaussian noise. Further, Gilmer et al. (2019) recently demonstrated a close relation between adversarial robustness and corruption robustness. However, in practice adversarially trained models often lead to poor performance even compared to the standard DNN based models when exposed to naturally occurring common corruptions, (e.g., IMAGENET-C (Hendrycks & Dietterich, 2019)), limiting the practical purview of them. Hence, it raises the question of whether adversarially trained models can be used for sensitive real-world applications?

Existing adversarially trained models typically perform inference on test examples independently from each other. This standard inference setup often underestimates their robustness in non-IID settings. For example, the distribution of test examples may change in the medical imaging settings, as we use a different data acquisition system or in autonomous cars, satellite image analysis as the weather condition changes. However, these external conditions do not change abruptly for most (but

not all) of the real-world applications. Hence, we can expect to receive potentially a large number of test examples from the same distribution during inference.

Unsupervised adaptation mechanisms are well studied in the field of domain adaptation (DA), where the aim is to modulate the model parameters for a different target domain (Li et al., 2016). Recent works have explored self-supervised training (Sun et al., 2020), batch normalization (BN) adaptation (Schneider et al., 2020; Nado et al., 2020) using test batches to improve the robustness against common corruptions for standard classifiers. In this paper, we investigate adapting BN statistics for adversarially trained models for both adversarial and common corruption robustness. In the standard inference setup, BN statistics of the DNN models are estimated during training and used without re-estimation for the test examples. However, activation statistics obtained during training time do not reflect the statistics of the test examples under covariate shifts. Here we apply adaptive BN, i.e., re-estimate the BN statistics using the test images to mitigate such covariate shifts (Li et al., 2016).

The contributions of our paper are as follows: We first demonstrate that this simple adaptation alone greatly improves the overall robustness of the adversarially trained models against Gaussian noise. This simple finding allows us to transform them into a smoothed classifier to provide certified robustness in $\ell_2$ norm without degrading their empirical robustness against adversarial attacks. Hence, we can provide test-time flexibility to obtain empirically robust predictions and also verify whether we can certify the prediction for sensitive real-world applications. Further, we show that adversarially trained models with adaptive BN significantly improves the performance against common perturbations and mitigates the performance gap between common corruption images and clean images, improving the generalization for real-world image classification tasks. To the best of our knowledge, existing frameworks only provide either empirical adversarial robustness (Madry et al., 2018; Rice et al., 2020; Nandy et al., 2020) or certified robustness (Cohen et al., 2019; Salman et al., 2019), or corruption robustness (Hendrycks & Dietterich, 2019; Sun et al., 2020). In contrast, our proposed framework of using adaptive BN at test-time on adversarially trained models offers all of these three benefits without any additional training and architectural overhead.

## 2 ADAPTIVE BATCH-NORMALIZATION

Using BN statistics from the training dataset generalizes well in the standard IID settings (Ioffe & Szegedy, 2015). However, in non-IID settings, the changes in the test distribution lead to different activation statistics for the test examples. Hence, impacted by covariate shift, the statistics estimated using the training batches cannot effectively normalize the activation tensors, breaking the crucial assumption for the subsequent hidden layers to work.

More formally, let $x \in \mathcal{X}$ are inputs and $y \in \mathcal{Y}$ are the class labels. We denote the training distribution as $P_T : \mathcal{X} \times \mathcal{Y} \rightarrow \mathbb{R}^+$ and test distribution as $P_t : \mathcal{X} \times \mathcal{Y} \rightarrow \mathbb{R}^+$. Then, there exists covariate shift between training and test distribution if: $P_T(y|x)$=$P_t(y|x)$ and $P_T(x) \neq P_t(x)$. Further, if the covariate shift only leads to change in the first and second order moments of the feature activations $f_h(x)$, we can remove it by re-estimating these statistics using the test batches, followed by normalization (Schneider et al., 2020): $P_T\Big(\frac{f_h(x)-\mathbb{E}_T[f_h(x)]}{\sqrt{\mathbb{V}_T[f_h(x)]}}\Big)P_T(x) \approx P_t\Big(\frac{f_h(x)-\mathbb{E}_t[f_h(x)]}{\sqrt{\mathbb{V}_t[f_h(x)]}}\Big)P_t(x)$. We can remove the effect of covariate shift by correcting first and second-order moments as long as the change in the test distribution only leads the feature activations to scale and translate. In the context of domain adaptation (Li et al., 2016) and corruption robustness (Nado et al., 2020; Schneider et al., 2020), adaptive BN to correct the BN statistics are found to be effective as long as the semantics in the test images does not change.

Adaptive BN adapts the BN statistics using the unlabelled test batches in an unsupervised fashion. Given a set of test examples, we compute the BN statistics using the feature activations of the test batch, denoted as $\mu_t$, $s_t^2$ and adapt them with the existing statistics, $\mu_T$, $s_T^2$ that are already obtained using the training batches as: $\overline{\mu} = \rho \cdot \mu_t + (1-\rho) \cdot \mu_T; \quad \overline{s} = \rho \cdot s_t + (1-\rho) \cdot s_T$; where, $\rho \in [0, 1]$ is the *momentum*. The choice of $\rho = 0$ rejects the statistics of the test examples. This is equivalent to the deterministic DNN classifier in the standard inference setup. In contrast, $\rho = 1$ completely ignores the pre-calculated statistics from the training batches. As we receive a larger test batch, we can get a better estimate of the test distribution. Hence, we should choose a larger value for $\rho$. In contrast, if the test-batch size is too small, the estimated statistics would be unreliable.

## 3 EXPERIMENTS

We present four sets of experiments on IMAGENET (Deng et al., 2009) with ResNet50 (He et al., 2016a;b). The adversarially trained models, $\text{Adv}_\infty$ and $\text{Adv}_2$ are learned using $\ell_\infty \leq 4/255$ and $\ell_2 \leq 3$ threat models with early stopping criteria (Rice et al., 2020). We compare with Baseline and $\text{Rand}_{\sigma=0.5}$ that are respectively trained using clean images and by augmenting random Gaussian noise, sampled from $\mathcal{N}(0, \sigma^2 I)$ with $\sigma = 0.5$. See Appendix B for more details.

**Robustness against Gaussian Noise.** In Table 1, we present the performance of different models at different levels of Gaussian augmented noises. We observe that when the test images are sampled from IID settings (i.e., $\sigma = 0$ for Baseline, $\text{Adv}_\infty$, and $\text{Adv}_2$ and $\sigma = 0.5$ for $\text{Rand}_{\sigma=0.5}$), the performance of different models remain similar regardless of whether BN adaptation is used. However, as we move away from the IID settings by increasing (or decreasing) $\sigma$, the performance of these models significantly degrades in the standard inference setup. In contrast, by applying adaptive BN, we can improve the performance of all classifiers. In particular, $\text{Adv}_\infty$ and $\text{Adv}_2$ achieve significantly higher improvements by applying adaptive BN. We visualize the loss gradients to further investigate the effect of the adaptive BN in Appendix C.

| Model | $\sigma = 0$ | $\sigma = 0.25$ | $\sigma = 0.5$ | $\sigma = 0.75$ |
|---|---|---|---|---|
| Baseline | $75.2_{\pm 0.0}$ | $11.8_{\pm 0.22}$ | $0.3_{\pm 0.01}$ | $0.1_{\pm 0.0}$ |
| + adaptive BN | $74.4_{\pm 0.04}$ | $31.0_{\pm 0.27}$ | $7.7_{\pm 0.24}$ | $2.4_{\pm 0.01}$ |
| $\text{Adv}_\infty$ | $62.8_{\pm 0.0}$ | $3.9_{\pm 0.03}$ | $0.4_{\pm 0.01}$ | $0.2_{\pm 0.01}$ |
| + adaptive BN | $60.8_{\pm 0.16}$ | $53.4_{\pm 0.15}$ | $44.9_{\pm 0.08}$ | $33.7_{\pm 0.28}$ |
| $\text{Adv}_2$ | $59.8_{\pm 0.0}$ | $9.8_{\pm 0.08}$ | $0.9_{\pm 0.01}$ | $0.3_{\pm 0.0}$ |
| + adaptive BN | $58.3_{\pm 0.08}$ | $53.7_{\pm 0.14}$ | $47.3_{\pm 0.14}$ | $39.8_{\pm 0.18}$ |
| $\text{Rand}_{\sigma=0.5}$ | $22.0_{\pm 0.0}$ | $32.8_{\pm 0.11}$ | $60.9_{\pm 0.04}$ | $0.9_{\pm 0.06}$ |
| + adaptive BN | $62.7_{\pm 0.03}$ | $62.3_{\pm 0.18}$ | $59.5_{\pm 0.11}$ | $51.4_{\pm 0.27}$ |

Table 1: Effect of adaptive BN on Top-1 accuracy under Gaussian noises. We randomly shuffle the test set and sample the noises to report $(mean \pm 2 \times sd)$ of 5 different runs.

**$\ell_2$ Certification for Adversarially Trained Models.** A classifier, $f$ that achieves robustness against standard Gaussian noise, can be transformed into a smoothed classifier, $g$ to provide certified robustness against $\ell_2$-norm Cohen et al. (2019). For an input $x$, $g(x)$ labels $x$ as class $y$ that the base classifier is most likely to return under noisy corruption $x + \delta$ s.t.: $g(x) = arg\max_{y \in \mathcal{Y}} \mathbb{P}(f(x + \delta) == y)$ where, $\mathcal{Y}$ is the set of class labels and $\delta \sim \mathcal{N}(0, \sigma^2 I)$. Since modulating BN statistics using adaptive BN technique significantly improves robustness against Gaussian noise for both $\text{Adv}_2$ and $\text{Adv}_\infty$, we can transform both of these models into a smoothed classifier to provide robustness certification for $\ell_2$ norm.

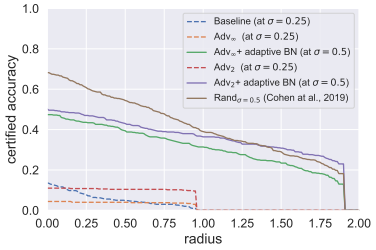



Figure 1: Certified accuracy at $\ell_2$ radii.

Here, we first modulate the BN statistics of the classifiers using adaptive BN by feeding the Gaussian noise augmented test batches. The noises are sampled from the same isotropic Gaussian distribution $\mathcal{N}(0, \sigma^2 I)$ as we use for certification. We then freeze the parameters for the whole certification process. That is, the base classifier $f$ remains the same for the process. Here, we use $\sigma = 0.5$ for $\text{Adv}_\infty$ and $\text{Adv}_2$ models with BN adaptation. We use $\sigma = 0.25$ for the Baseline, $\text{Adv}_\infty$ and $\text{Adv}_2$ models *without* BN adaption that are not robust against Gaussian noises (see Table 1). We also compare with the certification result of the standard randomized smoothing classifiers using $\text{Rand}_{\sigma=0.5}$ at $\sigma = 0.5$ as proposed in (Cohen et al., 2019). We use a subsampled IMAGENET test set of 500 samples, as in (Cohen et al., 2019) and certify them with $99.9\%$ probability. We can see in Figure 1, that both $\text{Adv}_\infty$ and $\text{Adv}_2$ models can be transformed using adaptive BN technique to provide certified robustness for $\ell_2$ norm. $\text{Adv}_2$ models consistently achieve better performance compared to $\text{Adv}_\infty$ in terms of certified accuracy. Further, $\text{Adv}_2$ produces similar certification as $\text{Rand}_{\sigma=0.5}$ beyond $\ell_2$-radii of 1.0 and outperforms $\text{Rand}_{\sigma=0.5}$ beyond $\ell_2$-radii of 1.5. Finally, we note that there exists improved randomized smoothing models (e.g., (Salman et al., 2019)) for superior robustness certification. However, they perform poorly against adversarial attacks or commonly occurring random perturbations, unlike our proposed framework.

**Under adversarial attacks.** Adversarially trained models provide the best empirical defense against adversarial attacks. Hence, we do not need to transform them into smoothed classifiers for practical use. This section evaluates the empirical performance of adversarially trained models as we apply the adaptive BN technique.

| Model | Threat Model | w/o adapt. BN | with adapt. BN | Diff |
|---|---|---|---|---|
| $\text{Adv}_\infty$ | $\|\|\delta\|\|_\infty \leq 4/255$ | 34.2 | $41.1_{\pm 1.14}$ | +6.9 |
| $\text{Adv}_2$ | $\|\|\delta\|\|_2 \leq 3.0$ | 35.4 | $37.0_{\pm 0.37}$ | +1.6 |

Table 2: Performance under adversarial attacks. For models with BN adaptation, we randomly shuffle the test images to report $(mean + 2 \times sd)$ of 3 different runs.

We use PGD attack with 100 iterations to evaluate the models without adaptive BN Madry et al. (2018). However, as we apply adaptive BN during inference, the BN parameters consistently change to accommodate the inputs at each forward pass. For this, we choose a set of $m = 10$ classifiers by applying adaptive BN technique on the original adversarially trained model using a test batch, perturbed with randomly chosen $\delta \in \Delta$ from the threat model (as a proxy for the adversarial perturbations). The adversarial examples are then produced using *expectation over transformation* (EoT) technique (Athalye et al., 2017; 2018). We present the results on a subset of 2000 images test examples. In Table 2, we observe a consistent improvement for the models using adaptive BN. To summarize, adaptive BN often improves performance against adversarial attacks without any detrimental effect. See Appendix E for additional details.

**Under Common Corruptions.** Hendrycks & Dietterich (2019) introduced IMAGENET-C by algorithmically generated corruptions from *noise, blur, weather, or digital* categories with 5 different severities for each corruptions. While adversarially trained models are effective against adversarial attacks, they often perform poorly against such common corruptions, even compared to the standard DNN models in the standard inference setup (Hendrycks et al., 2020; Gilmer et al., 2019). Here, we demonstrate that by applying the adaptive BN technique, we can greatly improve their performance.

The corruption robustness of a classifier, $f$ is typically measured using the mean corruption error ($mCE$) (Hendrycks & Dietterich, 2019). It is computed by normalizing the model's top-1 errors with the top-1 errors of AlexNet (Krizhevsky et al., 2012) across the set of corruptions, $K$ and corruption severities, $I$, as: $mCE(f) = \frac{1}{K}\sum_{k=1}^{K} \frac{\sum_{i=1}^{I} \text{err}_{k,i}^{f}}{\sum_{i=1}^{I} \text{err}_{k,i}^{Alex}}$; where, $\text{err}_{k,i}^{f}$ and $\text{err}_{k,i}^{Alex}$ denotes the top-1 error for corruption type $k$ at severity $i$ for $f$ and the AlexNet model respectively.

| Model | Top-1 Accuracy | | | | | | mCE | | relative-mCE | |
|---|---|---|---|---|---|---|---|---|---|---|
| | w/o adapt. BN | | | with adapt. BN | | | w/o | with | w/o | with |
| | Clean (↑) | Corrupt (↑) | Gaps (↓) | Clean (↑) | Corrupt (↑) | Gaps (↓) | adapt. BN (↓) | adapt. BN (↓) | adapt. BN (↓) | adapt. BN (↓) |
| Baseline | 75.2$_{\pm 0.0}$ | 39.0$_{\pm 0.0}$ | 36.2 | 74.4$_{\pm 0.14}$ | 49.0$_{\pm 0.01}$ | 25.4 | 77.0$_{\pm 0.0}$ | 64.3$_{\pm 0.01}$ | 103.3$_{\pm 0.0}$ | 70.8$_{\pm 0.45}$ |
| $\text{Adv}_\infty$ | 62.8$_{\pm 0.0}$ | 32.2$_{\pm 0.0}$ | 30.6 | 60.8$_{\pm 0.16}$ | 51.7$_{\pm 0.02}$ | 9.1 | 85.2$_{\pm 0.0}$ | 61.9$_{\pm 0.03}$ | 81.0$_{\pm 0.0}$ | 25.3$_{\pm 0.59}$ |
| $\text{Adv}_2$ | 59.8$_{\pm 0.0}$ | 31.3$_{\pm 0.0}$ | 28.5 | 58.3$_{\pm 0.08}$ | 50.3$_{\pm 0.02}$ | 8.0 | 86.5$_{\pm 0.0}$ | 63.9$_{\pm 0.03}$ | 75.7$_{\pm 0.0}$ | 22.6$_{\pm 0.28}$ |

Table 3: Effect of adaptive BN on adversarially trained models against corruption images. For models with BN adaptation, we randomly shuffle the test images to report ($mean + 2 \times sd$) of 3 different runs. For each column, down-arrow (↓) denotes the lower values are better and up-arrow (↑) denotes the higher values are better.

However, $mCE$ does not reflect the generalization gap between clean test images and the corruption images. For example, a classifier may withstand the corruption images, minimizing the generalization gaps from clean test images. However, it may produce a higher $mCE$ compared to a classifier with a low clean error rate while the error rate spike in the presence of corruption. Hence, (Hendrycks & Dietterich, 2019) proposed another metric, called relative mean corruption error ($rmCE$). It computes a normalized relative change of top-1 error rate of the classifier, $f$ from the clean error rate across different corruptions, $K$ and corruption severities, $I$, as: $rmCE(f) = \frac{1}{K}\sum_{k=1}^{K} \frac{\sum_{i=1}^{I} \text{err}_{k,i}^{f} - \text{err}_{Clean}^{f}}{\sum_{i=1}^{I} \text{err}_{k,i}^{Alex} - \text{err}_{Clean}^{Alex}}$; where, $\text{err}_{Clean}^{f}$ and $\text{err}_{Clean}^{Alex}$ denote the clean error rate for $f$ and the AlexNet model respectively. Since adversarially trained models already produce a high error rate for the clean test images to achieve adversarial robustness, $rmCE$ is a more appropriate metric for comparison.

In Table 3, we note that adversarially trained models already achieve lower $rmCE$ scores. Further, by applying the adaptive BN technique, these models significantly reduce $rmCE$ scores, leading to a better classification generalization. We also see that adaptive BN significantly reduces the gap in top-1 accuracy between clean and corrupted test images for adversarially trained models. In Appendix F.1, we also presents additional results.

## 4 CONCLUSION

We demonstrate that the test-time adaptive BN technique for adversarially trained models allows us to produce robustness certification and significantly improves the robustness against common corruptions without degrading their performance against adversarial attacks. These experimental findings allow us to obtain robust predictions as well as verify whether we can certify the prediction for sensitive real-world applications.

**ACKNOWLEDGMENT:** This research is supported by the National Research Foundation Singapore under its AI Singapore Programme (AISG-GC-2019-001 and AISG-RP-2018-008).

## REFERENCES


Anish Athalye, Logan Engstrom, Andrew Ilyas, and Kevin Kwok. Synthesizing robust adversarial examples. *arXiv*, 2017.

Anish Athalye, Nicholas Carlini, and David Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. In *ICML*, 2018.

Jeremy M Cohen, Elan Rosenfeld, and J Zico Kolter. Certified adversarial robustness via randomized smoothing. *ICML*, 2019.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009.

Logan Engstrom, Andrew Ilyas, Hadi Salman, Shibani Santurkar, and Dimitris Tsipras. Robustness (python library), 2019. URL `https://github.com/MadryLab/robustness`.

Christian Etmann, Sebastian Lunz, Peter Maass, and Carola-Bibiane Schönlieb. On the connection between adversarial robustness and saliency map interpretability. In *ICML*, 2019.

Justin Gilmer, Nicolas Ford, Nicholas Carlini, and Ekin Cubuk. Adversarial examples are a natural consequence of test error in noise. In *ICML*, 2019.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *IEEE CVPR*, 2016a.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. In *ECCV*, 2016b.

Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. In *ICLR*, 2019.

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.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*, 2015.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *NeurIPS*, 2012.

Alexey Kurakin, Ian J. Goodfellow, and Samy Bengio. Adversarial examples in the physical world. In *Workshop track, ICLR*, 2017.

Bai Li, Changyou Chen, Wenlin Wang, and Lawrence Carin. Certified adversarial robustness with additive noise. *NeurIPS*, 2019.

Yanghao Li, Naiyan Wang, Jianping Shi, Jiaying Liu, and Xiaodi Hou. Revisiting batch normalization for practical domain adaptation. *arXiv*, 2016.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *ICLR*, 2018.

Zachary Nado, Shreyas Padhy, D Sculley, Alexander D'Amour, Balaji Lakshminarayanan, and Jasper Snoek. Evaluating prediction-time batch normalization for robustness under covariate shift. *arXiv*, 2020.

Jay Nandy, Wynne Hsu, and Mong-Li Lee. Approximate manifold defense against multiple adversarial perturbations. In *International Joint Conference on Neural Networks (IJCNN)*, 2020.

Leslie Rice, Eric Wong, and Zico Kolter. Overfitting in adversarially robust deep learning. In *ICML*, 2020.

Hadi Salman, Jerry Li, Ilya Razenshteyn, Pengchuan Zhang, Huan Zhang, Sebastien Bubeck, and Greg Yang. Provably robust deep learning via adversarially trained smoothed classifiers. In *NeurIPS*, 2019.

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

Yu Sun, Xiaolong Wang, Zhuang Liu, John Miller, Alexei Efros, and Moritz Hardt. Test-time training with self-supervision for generalization under distribution shifts. In *ICML*, 2020.

Christian Szegedy et al. Intriguing properties of neural networks. In *ICLR*, 2014.

Florian Tramèr and Dan Boneh. Adversarial training and robustness for multiple perturbations. In *NeurIPS*, 2019.

Dimitris Tsipras, Shibani Santurkar, Logan Engstrom, Alexander Turner, and Aleksander Madry. Robustness may be at odds with accuracy. In *ICLR*, 2019.

Shiqi Wang, Kexin Pei, Justin Whitehouse, Junfeng Yang, and Suman Jana. Efficient formal safety analysis of neural networks. In *NeurIPS*, pp. 6367–6377, 2018.

Eric Wong and Zico Kolter. Provable defenses against adversarial examples via the convex outer adversarial polytope. In *ICML*, 2018.

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.

# A APPENDIX ORGANIZATION

We organize the appendix as follows:

- Section B presents the implementation details of training for our experiments and the hyper-parameter settings for adaptive BN technique during inference.
- Section C visualizes the loss gradients of adversarially trained models with and without BN adaptation as we augment Gaussian noise to the test images.
- Section D presents two additional experiments for $\ell_2$ certification.
- Section E provides additional ablation studies on the adversarial attacks.
- Section F presents additional results on corruption robustness.

# B EXPERIMENTAL SETUP

In this section, we first present the implementation details for training the classification models. Next, we discuss the choice of test-time hyper-parameter settings for using the adaptive batch-normalization (BN) technique.

## B.1 IMPLEMENTATION DETAILS

We present our experiments on IMAGENET Deng et al. (2009) dataset. We use ResNet50 models for IMAGENET He et al. (2016a). For Baseline and Rand$_{\sigma=0.5}$ models, we obtain the weights from Cohen et al. (2019)[1] that are trained using Gaussian augmented noises, sampled from isotropic Gaussian distribution $\mathcal{N}(0, \sigma^2 I)$ with $\sigma = 0.0$ (i.e., no noise) and $\sigma = 0.5$ respectively.

Here, the adversarially trained models i.e., Adv$_\infty$ and Adv$_2$ are learned for $\ell_\infty$ and $\ell_2$ threat models with threat boundary of $4/255$ and 3, respectively. We use the models provided by Rice et al. (2020) [2], which in turn is adopted from PGD-based adversarial training of the models provided by Engstrom et al. (2019) [3].

We resize the input images to $256 \times 265$ pixels and crop $224 \times 224$ pixels from the middle. Note that, ImageNet-C images are provided after resizing and cropping. So, we do not change those images. For our experiments on certification, we use a subsample of 500 test images by choosing at most 1 sample for each class. Also, we use a subsample of $2,000$ test images, with exactly 2 samples per class, for our experiments on the adversarial attack.

| $\rho$ | 0.0 | 0.1 | 0.3 | 0.5 | 0.7 | 0.9 | 1.0 |
|---|---|---|---|---|---|---|---|
| Adv$_\infty$ | 0.4$_{\pm 0.01}$ | 2.1$_{\pm 0.04}$ | 20.6$_{\pm 0.16}$ | 41.1$_{\pm 0.09}$ | 43.5$_{\pm 0.14}$ | 44.2$_{\pm 0.12}$ | 44.8$_{\pm 0.13}$ |
| Adv$_2$ | 0.9$_{\pm 0.01}$ | 7.7$_{\pm 0.09}$ | 36.6$_{\pm 0.09}$ | 45.5$_{\pm 0.13}$ | 46.7$_{\pm 0.13}$ | 46.8$_{\pm 0.13}$ | 47.2$_{\pm 0.14}$ |

Table 4: Top-1 accuracy using fixed test batch-size $= 512$ for adversarially trained models under Gaussian augmented noise with $\sigma = 0.5$ for different choices of momentum, $\rho$ during inference. We randomly shuffle the test images and randomly sampled Gaussian noises to report $(mean + 2 \times sd)$ of 5 different runs. $\rho = 0$ indicates the performance of models without BN adaptation.

| Batch Size | - | 8 | 16 | 32 | 64 | 128 | 256 | 512 |
|---|---|---|---|---|---|---|---|---|
| Adv$_\infty$ | 0.4$_{\pm 0.01}$ | 11.5$_{\pm 0.22}$ | 28.1$_{\pm 0.22}$ | 37.1$_{\pm 0.24}$ | 41.4$_{\pm 0.26}$ | 43.3$_{\pm 0.15}$ | 44.4$_{\pm 0.21}$ | 44.8$_{\pm 0.13}$ |
| Adv$_2$ | 0.9$_{\pm 0.01}$ | 9.1$_{\pm 0.15}$ | 26.7$_{\pm 0.14}$ | 37.6$_{\pm 0.2}$ | 42.9$_{\pm 0.12}$ | 45.4$_{\pm 0.13}$ | 46.7$_{\pm 0.07}$ | 47.2$_{\pm 0.14}$ |

Table 5: Top-1 accuracy using fixed $\rho = 1$ for adversarially trained models under Gaussian augmented noise with $\sigma = 0.5$ for different sizes of test batches during inference. We randomly shuffle the test images to report $(mean + 2 \times s.d.)$ of 5 different runs. '-' denotes the performance without any BN adaptation.

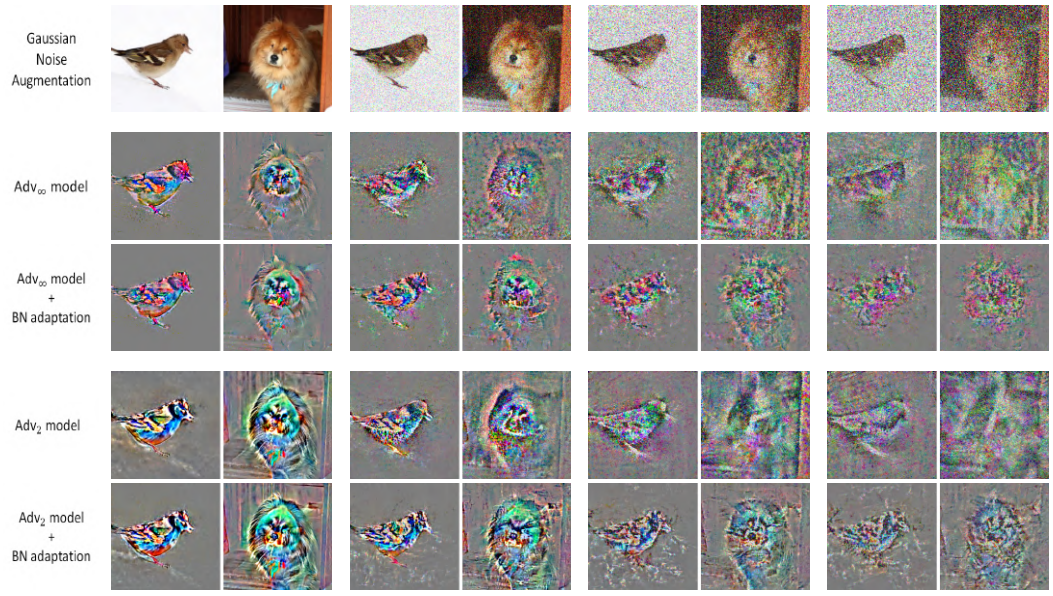
## B.2 CHOICE OF HYPER-PARAMETERS

During inference, adaptive BN technique is controlled using two hyper-parameters, i.e., the *test batch-size* and *momentum*, $\rho$ to update the statistics of the batch-normalization layers. We assume that the images in the test batches are obtained independently from the same test distribution. We compute the statistics corresponding

[1] https://github.com/locuslab/smoothing

[2] https://github.com/locuslab/robust_overfitting

[3] https://github.com/MadryLab/robustness




(a) $\sigma = 0$ (Clean) (b) $\sigma = 0.25$ (c) $\sigma = 0.5$ (d) $\sigma = 0.75$

Figure 2: Visualization of loss-gradient images produced by adversarially trained classifiers as we apply different levels of Gaussian noises in the test images.

to these images. The hyper-parameter $\rho \in [0, 1]$ controls the tread-off between existing training statistics and test statistics for updating the parameters. When a larger size of the test batch is available, we can get a better estimation of the test distribution. Hence, we can choose a larger value of $\rho$. However, the choice of test batch size is constrained by different factors in practice, such as available hardware resources.

Here, we analyze the effects of these hyper-parameters for adversarially trained models using adaptive BN technique during inference. For different choices, we compare the top-1 test accuracy under Gaussian augmented noise using $\sigma = 0.5$, to find the appropriate $\rho$ and the batch size. Since, the standard DNN classifier (i.e., Baseline) does not improve robustness under Gaussian noise at $\sigma = 0.5$ even after BN adaptation, we skip those models from our following analysis. Also, we refer to the previous work in Schneider et al. (2020); Nado et al. (2020) that analyzed the effects of these hyper-parameters for the standard DNN classifiers.

**Momentum, $\rho$.** We first investigate the effect of momentum, $\rho$ as we choose a large batch size of $512$. In Table 4, we present the performance of adversarially trained models as we choose different values of $\rho$. Recall that, $\rho = 1$ denotes *full-adaptation*. Here, we completely ignore the existing statistics, computed during training and use the test statistics. In contrast, $\rho = 0$ represents *no-adaptation*, where we completely ignore the test statistics. This leads to the standard deterministic classification model.

As we can see in Table 4, the performance of these classifiers started improving even $\rho = 0.7$ as we slightly modulate the BN statistics using that test batches. Further, the performance started improving significantly as we choose larger $\rho$ to rely more on the current test batches. We observe that the performance started converging at $\rho = 0.7$.

**Batch Size.** Next, we investigate the minimum size of the test batches that would allow us to use $\rho = 1$ (i.e., full-adaptation). In Table 5, we fix $\rho = 1$ and vary the test batch sizes as we evaluate the models. We observe that the performance of the models started improving even when we are using the test batches of size $8$. The performance improves as we choose larger sizes of test batches. We can see that their performance started converging as we choose a batch size of $64$.

## C Visualization of Loss Gradients

In Table 1, we observe that adversarially trained models achieve significantly higher improvements by applying adaptive BN. For example, at $\sigma = 0.5$, these models respectively achieve $0.3\%$, $0.4\%$, and $0.9\%$ top-1 accuracy for ImageNet dataset (Table 1). By using adaptive BN technique, $\text{Adv}_2$ and $\text{Adv}_\infty$ improves the top-1 accuracy to $44.9\%$, $47.3\%$, i.e., minimizing the performance gaps between clean test images and noisy images at $\sigma = 0.5$

from $62.4\%$ and $58.9\%$ to only $15.9\%$ and $11.0\%$ respectively. In contrast, after correcting the BN statistics, the Baseline achieves $7.7\%$ top-1 accuracy, only slightly reducing the performance gaps from $74.9\%$ to $66.7\%$.

To investigate the effect of adaptive BN technique for adversarially trained models, we visualize the loss gradients for individual pixels of an image as we increase the Gaussian noise, i.e., $\sigma$ (see Figure 2). Loss-gradients reflect the most important input pixels that strongly affect the prediction of a classifier. We scale, translate and clip the loss-gradient values for visualization, without using any complex processing techniques. For clean test images (i.e., at $\sigma = 0$), gradients for adversarially trained models align properly with perceptually relevant features, irrespective of whether BN adaptation is used Tsipras et al. (2019); Etmann et al. (2019). The effect of adaptive BN becomes prominent as $\sigma$ is increased. We can see that the overall loss gradient tends to become noisier as we increase the noise level to $\sigma$=0.5 and $\sigma$=0.75 However, the models without BN adaptation lead to produce sharper loss gradients (i.e., providing higher importance) even for background pixels. In contrast, models with BN adaptation produce sharper loss gradients for the pixels from the object of interest while suppressing the background pixels. Hence, the classifier is made available with required semantic information for classification. It is interesting to note that $\text{Adv}_2$ produces significantly more human-aligned loss gradients compared to $\text{Adv}_\infty$ at $\sigma$=0.75. This behaviour is also reflected in their classification as we can see that $\text{Adv}_\infty$ achieves $33.7\%$ top-1 accuracy at $\sigma$=0.75 while $\text{Adv}_2$ achieves $39.8\%$ (see Table 1).

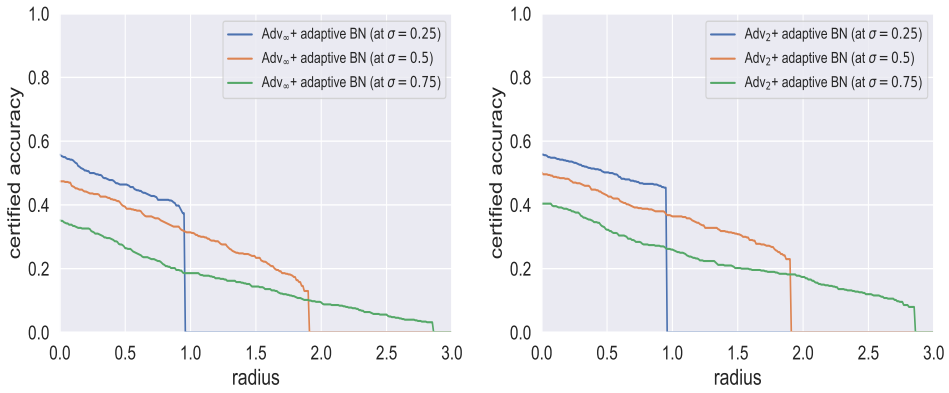



Figure 3: IMAGENET: Certified top-1 accuracy at various $\ell_2$ radii as we vary $\sigma$ for BN adaptation and certification for the same classification model. Adaptive BN allows to choose different values of optimum $\sigma$ for certifying different test examples. By selecting those optimum $\sigma$ for each test example, we can obtain an upper envelope of these curves as our certified top-1 accuracy.

| IMAGENET | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| $\ell_2$ Radius | 0.25 | 0.5 | 0.75 | 1.0 | 1.25 | 1.5 | 1.75 | 2.0 | 2.25 | 2.5 | 2.75 |
| $\text{Adv}_\infty$ + adapt BN (at $\sigma = 0.25$) | 50.0 | 46.4 | 41.6 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| $\text{Adv}_\infty$ + adapt BN (at $\sigma = 0.50$) | 43.6 | 39.4 | 35.8 | 31.4 | 27.6 | 23.4 | 18.2 | 0.0 | 0.0 | 0.0 | 0.0 |
| $\text{Adv}_\infty$ + adapt BN (at $\sigma = 0.75$) | 31.6 | 26.4 | 22.4 | 18.6 | 16.8 | 14.4 | 11.8 | 9.4 | 7.6 | 5.6 | 3.6 |
| $\text{Adv}_2$ + adapt BN (at $\sigma = 0.25$) | 53.2 | 50.2 | 46.8 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| $\text{Adv}_2$ + adapt BN (at $\sigma = 0.50$) | 47.0 | 43.0 | 39.0 | 36.4 | 32.8 | 30.8 | 27.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| $\text{Adv}_2$ + adapt BN (at $\sigma = 0.75$) | 37.8 | 32.2 | 28.4 | 26.0 | 22.4 | 20.2 | 19.0 | 17.4 | 14.2 | 12.0 | 9.6 |

Table 6: IMAGENET: Certified top-1 accuracy at various $\ell_2$ radii as we vary $\sigma$ for BN adaptation and certification for the same classification model. We use ResNet50 for IMAGENET.

## D FLEXIBILITY OF CHOOSING $\sigma$ DURING INFERENCE FOR CERTIFICATION USING BN ADAPTATION

The choice of $\sigma$ controls the tread-off between certified robust accuracy and certification boundary (Cohen et al., 2019; Salman et al., 2019; Li et al., 2019). When $\sigma$ is small, we can certify smaller radii with high accuracy. However, we cannot certify for large $\ell_2$ radii at all. In contrast, as we choose a higher value for $\sigma$, larger radii can be certified, but we get a lower certification accuracy at $\ell_2$ radii.

However, existing works require fixing an appropriate value of $\sigma$ during training to learn their base classifier (Cohen et al., 2019; Salman et al., 2019). In contrast, the adaptive BN technique provides the flexibility to choose an appropriate $\sigma$ during inference. In other words, we can choose different values of $\sigma$ for different test examples for the certification process. However, this process of choosing an appropriate value of $\sigma$ for different

| Model | Threat Boundary | Without adapt BN | Robust Test Accuracy With adaptive BN | | |
|---|---|---|---|---|---|
| | | | m=1 | m=5 | m=10 |
| $\text{Adv}_\infty$ | $\|\|\delta\|\|_\infty \leq 8/255$ | 34.2 | $41.5_{\pm 1.79}$ | $41.4_{\pm 2.55}$ | $41.1_{\pm 1.14}$ |
| $\text{Adv}_2$ | $\|\|\delta\|\|_2 \leq 1$ | 35.4 | $37.7_{\pm 0.82}$ | $37.0_{\pm 0.37}$ | $37.0_{\pm 0.3}$ |

Table 7: IMAGENET: Performance under adversarial attacks as we vary the number of models ($m$) to approximate the loss gradients for the adversarial attack. For models with BN adaptation, we randomly shuffle the test images to report ($mean + 2 \times sd$) of 3 different runs.

test examples for certification is extremely expensive. For each example, it would require applying the BN adaptation on the classifier and find the certification boundaries different values of $\sigma$ followed by choosing the largest $\ell_2$ certification boundary.

Here, we choose three different values $\sigma$ i.e., $\{0.25, 0.5, 0.75\}$ for our experiments. Figure 3 and Table 6 presents the results. We estimate the class label probabilities using Monte-Carlo sampling with $100,000$ noisy samples and certify the test examples with $99.9\%$ probability. We use the same $\text{Adv}_\infty$ and $\text{Adv}_2$ models as described in Section B. Note that, by choosing the appropriate values of $\sigma$ for the certification process, we can get the upper envelopes of these curves as our certified top-1 accuracy in Figure 3 for IMAGENET.

# E ADDITIONAL RESULTS UNDER ADVERSARIAL ATTACKS

In this section, we first present the EoT algorithm that we use to generate adversarial attack images. Next, we compare the strength of our adaptive attack as we increase the number of models to approximate the gradients.

## E.1 ADVERSARIAL ATTACK IMAGE GENERATION

For a test image, $x$ with label $y$, the goal of an adversarial attack is to find perturbation $\delta$ within a chosen perturbation $\epsilon$ for an $\ell_p$ norm such that:

$$\max_{\delta \in \Delta} \mathcal{L}(f_\theta(x+\delta), y) \quad where \;\; \Delta = \{\delta : ||\delta||_p \leq \epsilon\} \tag{1}$$

We use projected gradient descent (PGD) attack (Kurakin et al., 2017; Madry et al., 2018) to evaluate $\text{Adv}_\infty$ and $\text{Adv}_2$ models in the standard inference setup (i.e., without using adaptive BN). Starting with a random initial perturbation $\delta^{(0)}$, PGD attack iteratively adjusts the perturbation following the gradient steps with step-size $\eta$ and projects back onto the $\ell_p$ threat model:

$$\begin{aligned} \widetilde{\delta} &= \delta^{(t)} + \eta \cdot Q_p(\nabla_\delta \mathcal{L}(f_\theta(x+\delta^{(t)}), y)) \\ \delta^{(t+1)} &= \max(\min(\widetilde{\delta}, \epsilon), -\epsilon) \end{aligned} \tag{2}$$

For $\ell_\infty$, we use the sign of the gradient i.e., $Q_p(z) = \text{sign}(z)$. For other norms such as $\ell_2$, we normalize the gradient as: $Q_p(z) = z/||z||_2$.

In contrast to the standard inference setup, adaptive BN during inference consistently changes the BN parameters, accommodating the inputs at each forward pass. Hence, we cannot effectively compute the gradients, $\nabla_\delta \mathcal{L}(f_\theta(x+\delta), y)$, to produce the most effective adversarial examples. Here, we instead apply *expectation over transformation* (EoT) technique to approximate the gradient (Athalye et al., 2017). For cross-entropy loss, we can write the loss as: $\nabla_\delta \mathcal{L}(f_\theta(x+\delta), y) = \nabla_\delta(-\log(f_\theta(x+\delta)_y))$. To apply EoT, we select a set of $m$ models with varying BN statistics and approximate the gradient as:

$$\nabla_\delta[-\log \mathbb{E}_{\theta \in \Theta} f_\theta(x+\delta)_y] \approx \; \nabla_\delta[-\log \frac{1}{m} \sum_{i=1}^{m} f_{\theta^{(i)}}(x+\delta)_y] \tag{3}$$

where, $\{f_{\theta^{(i)}}\}_{i=1}^m$ are the classifiers with different BN parameters.

Each $f_{\theta^{(i)}}$ are obtained by applying adaptive BN technique on the original classifier $f_\theta$ using test batch, perturbed with randomly chosen $\delta \in \Delta$ from the threat model (as a proxy of the adversarial perturbations). Notably, We place log outside of the expectation as it was found to be more effective (Salman et al., 2019).

We use the same threat boundaries for attacks as used during the adversarial training. That is, we use $||\delta||_\infty \leq \frac{4}{255}$ and $||\delta||_2 \leq 3$ for $\text{Adv}_\infty$ and $\text{Adv}_2$ models respectively. The adversarial examples are generated using 100 iterations.

### E.2 EFFECT OF THE EOT HYPER-PARAMETER ($m$)

We recall from Equation 3 that for adversarially trained models with BN adaptation, we need to approximate the gradients during back-propagation at each step of the adaptive adversarial attack using Equation 3. We select a set of $m$ different classifiers, $\{f_{\theta^{(i)}}\}_{i=1}^{m}$ with different BN parameters for this approximation. We obtain each $f_{\theta^{(i)}}$ by applying adaptive BN technique on the original classifier $f_\theta$ using test batch, perturbed with randomly chosen $\delta \in \Delta$ from the threat model. For our experiments in Table 2, we choose $m = 10$.

Here, we compare the strength of our adaptive attack as we vary the number of models (i.e., $m$) to approximate the gradients. Table 7 present the results. We observe that the performance of the adversarial attacks does not change significantly as we vary $m$ to approximate the gradients. This is because we obtain a set of classifier $\{f_{\theta^{(i)}}\}_{i=1}^{m}$ by applying adaptive BN technique on the original classifier $f_\theta$ using a large test batch-size of $500$ images, perturbed with randomly sampled noise from a small threat boundary (as a proxy for the adversarial perturbations). This results in producing the classifiers, $f_{\theta^{(i)}}$ with only small changes in their batch-normalization parameters. Hence, for a given test image, these models produce almost the same loss gradients with respect to the pixels. As a result, we cannot significantly improve the strength of the adversarial attacks by choosing a larger $m$.

## F ADDITIONAL RESULTS ON COMMON CORRUPTIONS

In this section, we first present the accuracy attained by different classifiers under different common perturbations. Next, we visualize the loss gradients of adversarially trained models using adaptive BN technique under different corruptions from IMAGENET-C dataset.

### F.1 PERFORMANCE UNDER DIFFERENT CORRUPTIONS

Hendrycks & Dietterich (2019) introduced IMAGENET-C by algorithmically generated corruptions from noise, blur, weather, or digital categories with $I = 5$ different severities for each corruptions. In Table 8, we present the top-1 accuracy different classifiers under each corruption, averaged over their $5$ severity levels:

$$\text{Top-1 Acc.}(f)_k = \Big(1 - \text{error rate}\Big) = \Big(1 - \frac{\text{err}_{k,i}^{f}}{I}\Big)$$

where, $f$ denotes the classifier, $k$ is the corruption type and $i$ is the severity level.

Recall that, we use the top-1 error rate for individual corruptions, normalized using AlexNet error rates to measure the corruption robustness i.e $mCE$ and $rmCE$ scores. Hendrycks & Dietterich (2019) already provided the performance under different corruptions to compute the $mCE$ and $rmCE$ scores for IMAGENET-C.

We observe that the adversarially trained models, without BN adaptation technique, significantly dropped the accuracy under corruptions from noise, blur, and weather category for IMAGENET-C, compared to their clean test accuracy. These models often achieve significantly lower accuracy compared to the Baseline model without BN adaptation. For example, $\text{Adv}_\infty$ and $\text{Adv}_2$ achieve an average top-1 accuracy of $26.5\%$ and $24.1\%$, compared to $43.0\%$ for the Baseline model without BN adaptation. However, as we apply BN adaptation, the performance of $\text{Adv}_\infty$ and $\text{Adv}_2$ is significantly improved by approximately $20\%$ in top-1 accuracy for IMAGENET-C.

### F.2 LOSS GRADIENTS UNDER DIFFERENT PERTURBATIONS

Previous studies have shown that the loss gradients for adversarially trained models align properly with perceptually relevant features for clean images, without the perturbations (Tsipras et al., 2019; Etmann et al., 2019). In this section, we investigate the loss gradients of these models in presence of random perturbations from different categories of corruptions.

In Section C, we have already visualized that in presence of high Gaussian noise, adversarially trained models without BN adaptation produce noisy loss gradients with higher values for the background pixels. The adaptive BN technique mitigates this effect. However, in Figure 4 we observe that this behavior is not generalized for all types of random corruptions. Here, we visualize the Fourier spectrum of the corruption images along with the loss gradients values for $\text{Adv}_\infty$ model with and without applying BN adaptation techniques [4].

We note that, when the corruption images have a higher concentration for lower Fourier frequencies (called "low" frequency corruption), the adversarially trained models without BN adaptation tend to produce blurry loss-gradients (see the first column in Figure 4). In other words, the classifier provides almost uniform importance to all input pixels to predict these corrupted images. Generally, we observe this pattern for corruption images from blur and weather categories.

[4] Please refer to (Yin et al., 2019) (see Figure 2) for more details of Fourier perspective on random corruptions.

(a) IMAGENET-C

| Category | Corruption | AlexNet | Top-1 Accuracy: Baseline (ResNet50) w/o adapt BN | Baseline (ResNet50) with adapt BN | $Adv_\infty$ w/o adapt BN | $Adv_\infty$ with adapt BN | $Adv_2$ w/o adapt BN | $Adv_2$ with adapt BN |
|---|---|---|---|---|---|---|---|---|
| Clean | - | 56.5±0.0 | 75.2±0.0 | 74.4±0.14 | 62.8±0.0 | 60.8±0.16 | 59.8±0.0 | 58.3±0.08 |
| Noise | Gaussian Noise | 11.4±0.0 | 31.5±0.0 | 42.0±0.04 | 24.0±0.0 | 55.5±0.09 | 23.4±0.0 | 54.3±0.01 |
| | Shot Noise | 10.6±0.0 | 29.6±0.0 | 41.0±0.05 | 22.7±0.0 | 54.8±0.03 | 22.9±0.0 | 54.0±0.09 |
| | Impulse Noise | 7.7±0.0 | 26.3±0.0 | 37.6±0.09 | 14.6±0.0 | 53.7±0.03 | 19.7±0.0 | 54.5±0.09 |
| Average. | | 9.9±0.0 | 29.1±0.0 | 40.2±0.02 | 20.4±0.0 | 54.7±0.04 | 22.0±0.0 | 54.3±0.06 |
| Blur | Defocus Blur | 18.0±0.0 | 39.1±0.0 | 36.3±0.09 | 23.9±0.0 | 45.2±0.03 | 25.2±0.0 | 45.8±0.01 |
| | Glass Blur | 17.4±0.0 | 27.2±0.0 | 37.4±0.02 | 33.7±0.0 | 49.8±0.09 | 34.1±0.0 | 49.3±0.04 |
| | Motion Blur | 21.4±0.0 | 38.1±0.0 | 47.9±0.07 | 33.0±0.0 | 50.6±0.06 | 32.9±0.0 | 49.7±0.03 |
| | Zoom Blur | 20.2±0.0 | 35.4±0.0 | 49.5±0.02 | 30.3±0.0 | 50.9±0.03 | 33.6±0.0 | 50.3±0.02 |
| Average. | | 19.3±0.0 | 35.0±0.0 | 42.8±0.04 | 31.2±0.0 | 49.1±0.02 | 31.5±0.0 | 48.8±0.01 |
| Weather | Snow | 13.3±0.0 | 29.3±0.0 | 43.5±0.07 | 30.3±0.0 | 48.4±0.09 | 25.1±0.0 | 45.0±0.03 |
| | Fog | 18.1±0.0 | 44.1±0.0 | 58.3±0.03 | 6.9±0.0 | 45.2±0.06 | 6.1±0.0 | 42.4±0.06 |
| | Frost | 17.3±0.0 | 36.4±0.0 | 44.2±0.05 | 31.6±0.0 | 49.5±0.04 | 28.3±0.0 | 45.7±0.06 |
| | Brightness | 43.5±0.0 | 66.5±0.0 | 69.8±0.04 | 54.4±0.0 | 58.4±0.06 | 51.4±0.0 | 55.4±0.05 |
| | Contrast | 14.7±0.0 | 38.8±0.0 | 50.9±0.05 | 9.1±0.0 | 43.0±0.05 | 9.6±0.0 | 44.0±0.04 |
| Average. | | 21.4±0.0 | 43.0±0.0 | 53.4±0.01 | 26.5±0.0 | 48.9±0.04 | 24.1±0.0 | 46.5±0.02 |
| Digital | Elastic Transform | 35.4±0.0 | 45.4±0.0 | 58.9±0.01 | 48.2±0.0 | 52.7±0.03 | 46.5±0.0 | 50.9±0.05 |
| | Pixelate | 28.2±0.0 | 43.8±0.0 | 61.4±0.06 | 56.4±0.0 | 58.6±0.06 | 54.5±0.0 | 56.4±0.02 |
| | JPEG Compression | 39.3±0.0 | 53.4±0.0 | 56.7±0.05 | 59.4±0.0 | 59.2±0.09 | 56.1±0.0 | 56.7±0.05 |
| Average. | | 34.2±0.0 | 47.5±0.0 | 59.0±0.01 | 54.7±0.0 | 56.8±0.04 | 52.4±0.0 | 54.7±0.03 |
| Hold-out Noise | Speckle Noise | 15.5±0.0 | 37.5±0.0 | 48.2±0.03 | 30.0±0.0 | 55.7±0.04 | 30.1±0.0 | 54.9±0.03 |
| Hold-out Blur | Gaussian Blur | 21.3±0.0 | 42.7±0.0 | 40.2±0.03 | 27.7±0.0 | 47.1±0.08 | 28.4±0.0 | 47.1±0.04 |
| Hold-out Weather | Spatter | 28.2±0.0 | 48.3±0.0 | 55.0±0.11 | 44.1±0.0 | 51.9±0.05 | 43.1±0.0 | 51.0±0.05 |
| Hold-out Digital | Saturate | 34.2±0.0 | 61.0±0.0 | 67.8±0.03 | 48.1±0.0 | 56.4±0.04 | 43.9±0.0 | 54.6±0.03 |

Table 8: IMAGENET-C: Top-1 Accuracy achieved by different classification models under each individual corruption. For models with BN adaptation, we randomly shuffle the test images to report $(mean + 2 \times sd)$ of 3 different runs.

In contrast, for corruption images with larger concentration for higher Fourier frequencies, adversarially trained models without BN adaptation produces sharper loss-gradients. As the concentration for higher Fourier frequencies increases, it produces high loss-gradient values even for back-ground pixels (see the third column in Figure 4). As a consequence, the loss-gradients become noisy, obfuscating the pixels from object-of-interest. We observe this pattern for corruption images from the noisy category.

Finally, we can see that the adaptive BN technique mitigates the blurry patterns for corruption images from blur and weather categories as well as the noisy patterns for the corruption images from the noise category. In other words, it mitigates the effect of any random corruption to restore the alignment with perceptually relevant features. This allows the adversarially trained models to produce significantly higher Top-1 classification accuracy for random corruptions (Table 8).

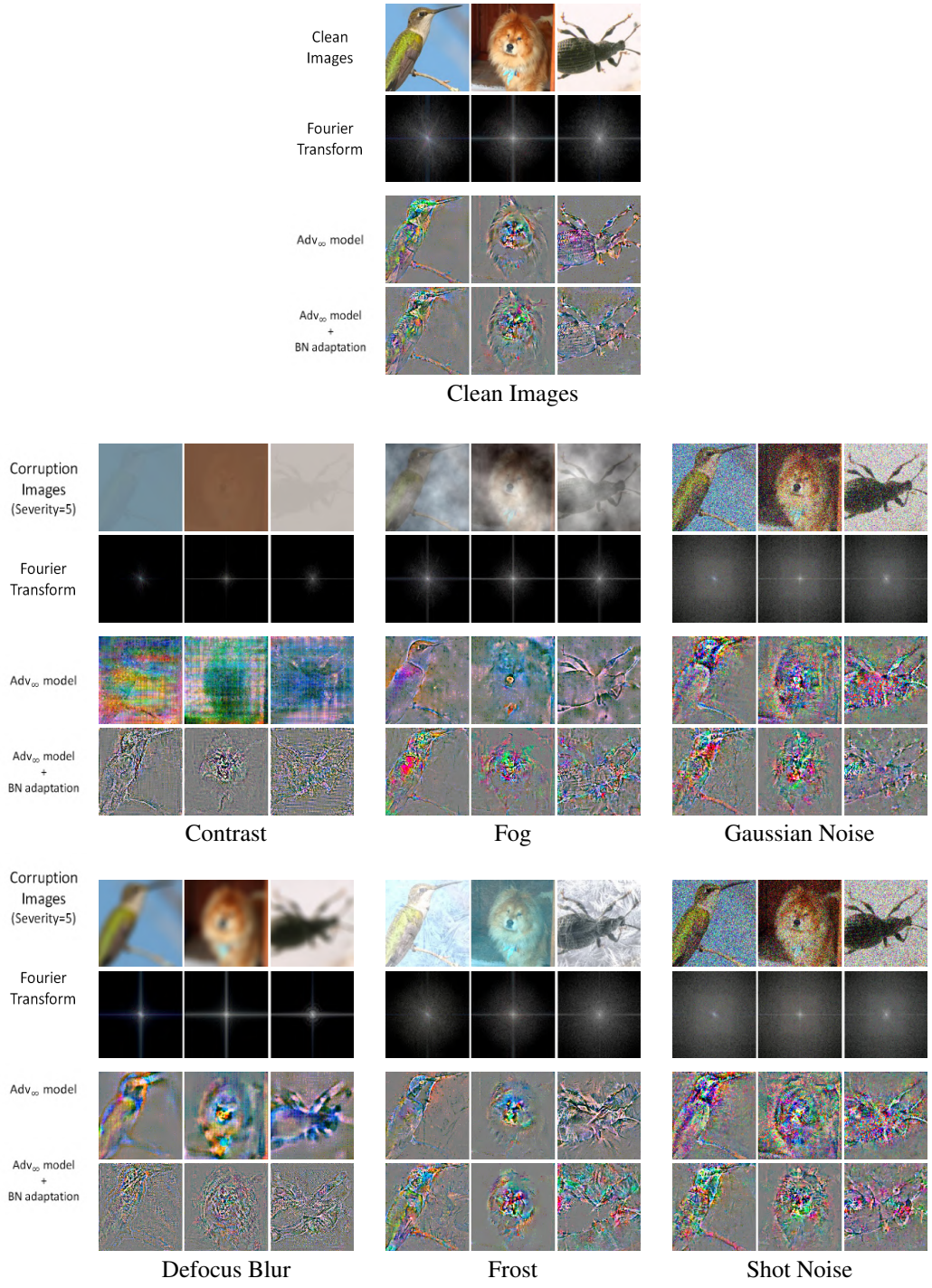



Figure 4: Corruption images with a higher concentration of Fourier frequencies produce blurry loss-gradients, while, corruption images with a lower concentration of Fourier frequencies lead to noisy loss-gradient patterns. Adaptive BN significantly adaptive BN technique mitigates these effects to restore the perceptual alignment.