INCREASING THE ROBUSTNESS OF DNNS AGAINST IMAGE CORRUPTIONS BY PLAYING THE GAME OF NOISE

Evgenia Rusak[*a,1,2], **Lukas Schott**[*a,1,2], **Roland Zimmermann**[*a,1], **Julian Bitterwolf**[b,1], **Oliver Bringmann**[†b,1], **Matthias Bethge**[†a,1], and **Wieland Brendel**[†a,1]

[*]equal contribution
[†]joint senior authors
[a]first.last@bethgelab.org
[b]first.last@uni-tuebingen.de
[1]University of Tübingen
[2]International Max Planck Research School for Intelligent Systems

ABSTRACT

The human visual system is remarkably robust against a wide range of naturally occurring variations and corruptions like rain or snow. In contrast, the performance of modern image recognition models strongly degrades when evaluated on previously unseen corruptions. Here, we demonstrate that a simple but properly tuned training with additive Gaussian and Speckle noise generalizes surprisingly well to unseen corruptions, easily reaching the previous state of the art on the corruption benchmark ImageNet-C (with ResNet50) and on MNIST-C. We build on top of these strong baseline results and show that an adversarial training of the recognition model against uncorrelated worst-case noise distributions leads to an additional increase in performance. This regularization can be combined with previously proposed defense methods for further improvement.

1 INTRODUCTION

While Deep Neural Networks (DNNs) have surpassed the functional performance of humans in a range of complex cognitive tasks (He et al., 2016; Xiong et al., 2016; Silver et al., 2017; Campbell et al., 2002; OpenAI, 2018), they still lag behind humans in numerous other aspects. One fundamental shortcoming of machines is their lack of robustness against input perturbations. Even minimal perturbations that are hardly noticeable for humans can derail the predictions of high-performance neural networks. For the purpose of this paper, we distinguish between two types of input perturbations. One type are minimal image-dependent perturbations specifically designed to fool a neural network with the smallest possible change to the input. These so-called *adversarial perturbations* have been the subject of hundreds of papers in the past five years, see e.g. (Szegedy et al., 2013; Madry et al., 2018; Schott et al., 2019; Gilmer et al., 2018). Another, much less studied type are *common corruptions*, which occur naturally in many applications. We argue that in many practical applications robustness to common corruptions is often more relevant than robustness to artificially designed adversarial perturbations. Besides its practical relevance, robustness to common corruptions is also an excellent target in its own right for researchers in the field of adversarial robustness and domain adaptation. Common corruptions can be seen as distributional shifts or as a weak form of adversarial examples that live in a smaller, constrained subspace.

We demonstrate that data augmentation with Gaussian or Speckle noise serves as a simple yet very strong baseline that is sufficient to surpass almost all previously proposed defenses against common corruptions on ImageNet-C for a ResNet50 architecture. Next, we introduce a neural network-based *adversarial noise generator* that can learn arbitrary uncorrelated noise distributions that maximally fool a given recognition network when added to their inputs. Based on this, we design and validate a constrained Adversarial Noise Training (ANT) scheme through which the recognition network learns to become robust against adversarial noise. We demonstrate that our ANT reaches state-of-the-art robustness on the corruption benchmark ImageNet-C for the commonly used ResNet50 architecture.

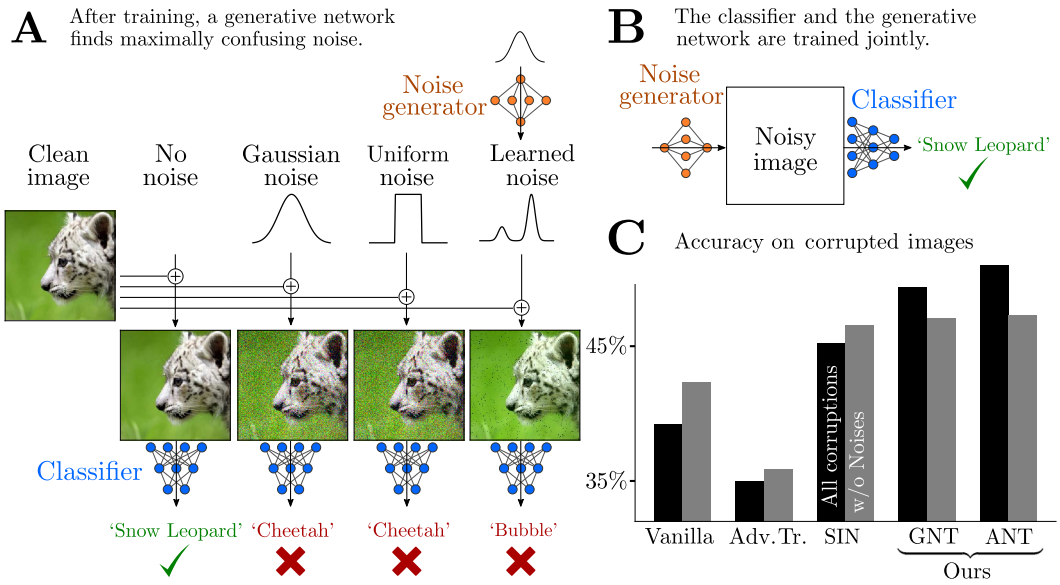

Figure 1. Outline of our approach. A: First, we train a generative network against a vanilla trained classifier to find the adversarial noise. B: To achieve robustness against adversarial noise, we train the classifier and the noise generator jointly. C: We measure the robustness against common corruptions for a vanilla, adversarially trained (Adv. Tr.), trained on Stylized ImageNet (SIN), trained via Gaussian data augmentation (GNT) and trained with the means of Adversarial Noise Training (ANT). With our methods, we achieve the highest accuracy on common corruptions, both on all and non-Noise categories.

We discuss work related to ours in Appendix A. We released our trained model weights along with evaluation code on github.com/bethgelab/game-of-noise.

2 METHODS

Training with Gaussian noise Several researchers have tried using Gaussian noise as a method to increase robustness towards common corruptions with mixed results (see Appendix A). In contrast to previous work, we treat the standard deviation σ of the distribution as a hyper-parameter of the training and measure its influence on robustness. To formally introduce the objective, let $\mathcal{D}$ be the data distribution over data samples $(\boldsymbol{x}, y)$.We train a differentiable classifier $f_\theta(\boldsymbol{x})$ by minimizing the risk on a dataset with additive Gaussian noise

$$\mathop{\mathbb{E}}_{\boldsymbol{x},y\sim\mathcal{D}} \mathop{\mathbb{E}}_{\boldsymbol{\delta}\sim\mathcal{N}(\mathbf{0},\sigma^2\mathbf{1})} \left[\mathcal{L}_{\text{CE}}\left(f_\theta(\boldsymbol{x}+\boldsymbol{\delta}), y\right)\right], \tag{1}$$

where σ is the standard deviation of the Gaussian noise and $\boldsymbol{x}+\boldsymbol{\delta}$ is clipped to the input range $[0,1]^N$. To maintain high accuracy on clean data, we only perturb 50% of the training data with Gaussian noise within each batch.

Learning Adversarial Noise Our goal is to find a noise distribution $p_\phi(\boldsymbol{\delta})$, $\boldsymbol{\delta} \in \mathbb{R}^N$ such that noise samples added to $\boldsymbol{x}$ maximally confuse the classifier f_θ. More concisely, we optimize

$$\max_\phi \mathop{\mathbb{E}}_{\boldsymbol{x},y\sim\mathcal{D}} \mathop{\mathbb{E}}_{\boldsymbol{\delta}\sim p_\phi(\boldsymbol{\delta})} \left[\mathcal{L}_{\text{CE}}\left(f_\theta(\text{clip}(\boldsymbol{x}+\boldsymbol{\delta})), y\right)\right], \tag{2}$$

where clip is an operator that clips all values to the valid interval (i.e. $\text{clip}(\boldsymbol{x}+\boldsymbol{\delta}) \in [0,1]^N$) and $||\boldsymbol{\delta}||_2 = \epsilon$.

We do not have to explicitly model the probability density function $p_\phi(\boldsymbol{\delta})$ since optimizing Eq. (2) only involves samples drawn from $p_\phi(\boldsymbol{\delta})$. This sampling process is implemented by a neural network, which we call noise generator. Details of its implementation are given in Appendix B.

Adversarial Noise Training To increase robustness, we now train the classifier f_θ to minimize the risk under adversarial noise distributions jointly with the noise generator

$$\min_{\theta} \max_{\phi} \mathop{\mathbb{E}}_{\boldsymbol{x},y\sim\mathcal{D}} \mathop{\mathbb{E}}_{\boldsymbol{\delta}\sim p_\phi(\boldsymbol{\delta})} \left[\mathcal{L}_{\text{CE}}\left(f_\theta(\boldsymbol{x}+\boldsymbol{\delta}), y\right)\right], \tag{3}$$

where again $\boldsymbol{x}+\boldsymbol{\delta} \in [0,1]^N$ and $||\boldsymbol{\delta}||_2 = \epsilon$. For a joint adversarial training, we alternate between an outer loop of classifier update steps and an inner loop of generator update steps. This is also depicted schematically in Fig. 1B. Note that in regular adversarial training, e.g. (Madry et al., 2018), $\boldsymbol{\delta}$ is optimized directly whereas we optimize a constrained distribution over $\boldsymbol{\delta}$.

Combining Adversarial Noise Training with stylization As demonstrated by Geirhos et al. (2019), using random stylization as data augmentation increases the accuracy on ImageNet-C. The robustness gains are attributed to a stronger shape bias of the classifier. We combine our ANT and the stylization approach to achieve robustness gains from both.

3 EXPERIMENTS

General setup All technical details, hyper-parameters and the architecture of the noise generator can be found in Appendix B-C.

(In-)Effectiveness of regular adversarial training to increase robustness towards common corruptions We find that robustness against regular adversarial examples does not generalize to robustness against common corruptions. Details on our experiments on adversarial robustness for ImageNet and MNIST can be found in Appendix D.

Effectiveness of Gaussian data augmentation to increase robustness towards common corruptions We fine-tune a pretrained image classifier with Gaussian data augmentation from the distribution $\mathcal{N}(\mathbf{0}, \sigma^2\mathbb{1})$ and vary σ. The Top-1 accuracy of the fine-tuned models on ImageNet-C and a comparison to a vanilla trained model is shown in Fig. 2. Each black point shows the performance of one model fine-tuned with one specific σ; the vanilla trained model is marked by the point at $\sigma = 0$. The horizontal lines indicate that the model is fine-tuned with Gaussian noise where σ is sampled from a set for each image. We show both the results on the full ImageNet-C evaluation set and the results on ImageNet-C without Noises (namely Blur, Weather and Digital) since Gaussian noise is part of the test set. To give a feeling of how the different σ-levels manifest themselves in an image, we include example images for all σ-levels in Appendix E. The Figure demonstrates that Gaussian noise generalizes well to the non-noise corruptions of the ImageNet-C evaluation dataset and is a powerful baseline. This is a surprising result as it was shown in several recent works that training on Gaussian or uniform noise does not generalize to other corruption types (Geirhos et al., 2018; Lopes et al., 2019) or that the effect is very weak (Ford et al., 2019). The standard deviation σ is a crucial hyper-parameter and has an optimal value of about $\sigma = 0.5$ for ResNet50.

In the next section, we will compare Gaussian data augmentation to our Adversarial Noise Training (see Appendix F for details on evaluating adversarial noise) and baselines from the literature. For this, we will use the models with the overall best-performance: The model $\text{GN}_{0.5}$ that was trained with Gaussian data augmentation with a single $\sigma = 0.5$ and the model GN_{mult} where σ was sampled from the set $\{0.08, 0.12, 0.18, 0.26, 0.38\}$, which corresponds to the Gaussian corruption of ImageNet-C.

Comparison of different methods to increase robustness towards common corruptions We now compare the robustness of differently trained models on the ImageNet-C benchmark. We consider our two best models trained with Gaussian data augmentation (GNT) and a model trained via Adversarial Noise Training (ANT). We also train a model with a combination of ANT and stylization (ANT+SIN). Since Gaussian noise is part of ImageNet-C, we train another baseline model with data augmentation using the Speckle noise corruption from the ImageNet-C holdout set. We later denote the cases where the corruptions present during training are part of the test set by putting corresponding accuracy values in brackets. Additionally, we compare our results with several baseline models from the literature. A description and discussion of these can be found in Appendix G. The Top-1 accuracies on the full ImageNet-C dataset and ImageNet-C without the Noise corruptions are displayed in Table 1; detailed results on individual corruptions in terms of accuracy and mCE are

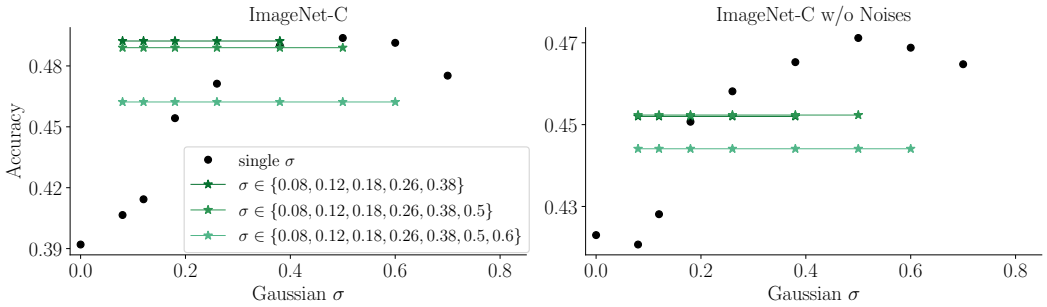

Figure 2. Top-1 accuracy on ImageNet-C (left) and ImageNet-C without the Noise corruptions (right) of a ResNet50 fine-tuned with Gaussian data augmentation of varying σ. We train on Gaussian noise sampled from a distribution with a single σ (black dots) and on distributions where σ is sampled from different sets (green lines with stars). We also compare to a vanilla trained model at $\sigma = 0$.

	IN	IN-C		IN-C w/o Noises	
model	clean acc.	Top-1	Top-5	Top-1	Top-5
Vanilla RN50	76.1	39.2	59.3	42.3	63.2
Shift Inv (Zhang, 2019)	77.0	41.4	61.8	44.2	65.1
Patch GN (Lopes et al., 2019)	76.0	(43.6)	(n.a.)	43.7	n.a.
SIN+IN (Geirhos et al., 2019)	74.6	45.2	66.6	46.6	68.2
AugMix (Hendrycks et al., 2020)	**77.5**	(48.3)	(69.2)	(50.4)	(71.8)
Speckle	75.8	**46.4**	**67.6**	44.5	65.5
GNT_{mult}	76.1	(49.2)	(70.2)	45.2	66.2
$\text{GNT}\sigma_{0.5}$	75.9	(49.4)	(70.6)	47.1	68.3
ANT	76.0	(51.1)	(72.2)	47.7	68.8
ANT+SIN	74.9	(52.2)	(73.6)	**49.2**	**70.6**

Table 1: Average accuracy on clean data, average Top-1 and Top-5 accuracies in percent on ImageNet-C and ImageNet-C without the Noise categories (higher is better). Gray numbers in brackets indicate scenarios where a corruption from the test set was used during training.

shown in Tables 8 and 9, Appendix H. We also calculate the accuracy on corruptions without the Noise category as our approach is to either add Gaussian noise or produce uncorrelated adversarial noise.

The results on full ImageNet-C are striking: a very simple baseline, namely a model trained with Speckle noise data augmentation, beats almost all previous baselines reaching an accuracy of 46.4% which is larger than the accuracy of SIN+IN (45.2%) and close to AugMix (48.3%). However, AugMix uses augmentations that are not clearly independent from the test set corruptions. The $\text{GN}\sigma_{0.5}$ surpasses SIN+IN not only on the Noise categories but also on almost all other corruptions, see Table 1 and a more detailed breakdown in Table 8, Appendix H. The ANT+SIN model produces the best results on ImageNet-C without Noises. Thus, it is slightly superior to Gaussian data augmentation and pure ANT. The results on MNIST-C can be found in Appendix I.

4 Discussion & Conclusion

So far, attempts to use simple noise augmentations for general robustness against common corruptions have produced mixed results, ranging from no generalization from one noise to other noise types (Geirhos et al., 2018) to only marginal robustness increases (Ford et al., 2019; Lopes et al., 2019). In this work, we demonstrate that carefully tuned additive noise patterns in conjunction with training on clean samples can surpass almost all current state-of-the-art defense methods against common corruptions. Additionally, we show that training against simple uncorrelated worst-case noise patterns outperforms our already strong baseline defense, with additional gains to be made in combination with previous defense methods like stylization training (Geirhos et al., 2019).

5 ACKNOWLEDGEMENTS

The authors thank the International Max Planck Research School for Intelligent Systems (IMPRS-IS) for supporting Evgenia Rusak and Lukas Schott. The authors thank Yash Sharma for helpful discussions and Alexander Ecker, Robert Geirhos and Dylan Paiton for helpful feed-back while writing the manuscript.

REFERENCES

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

Wayne Xiong, Jasha Droppo, Xuedong Huang, Frank Seide, Mike Seltzer, Andreas Stolcke, Dong Yu, and Geoffrey Zweig. Achieving human parity in conversational speech recognition. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 2016.

David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, L Robert Baker, Matthew Lai, Adrian Bolton, Yutian Chen, Timothy P. Lillicrap, Fan Fong Celine Hui, Laurent Sifre, George van den Driessche, Thore Graepel, and Demis Hassabis. Mastering the game of go without human knowledge. *Nature*, 550:354–359, 2017.

Murray Campbell, A. Joseph Hoane, Jr., and Feng-hsiung Hsu. Deep blue. *Artif. Intell.*, 134 (1-2):57–83, January 2002. ISSN 0004-3702. doi: 10.1016/S0004-3702(01)00129-1. URL http://dx.doi.org/10.1016/S0004-3702(01)00129-1.

OpenAI. Openai five. https://blog.openai.com/openai-five/, 2018.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=rJzIBfZAb.

Lukas Schott, Jonas Rauber, Matthias Bethge, and Wieland Brendel. Towards the first adversarially robust neural network model on MNIST. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=S1EHOsC9tX.

Justin Gilmer, Luke Metz, Fartash Faghri, Samuel S. Schoenholz, Maithra Raghu, Martin Wattenberg, and Ian J. Goodfellow. Adversarial spheres. *CoRR*, abs/1801.02774, 2018. URL http://arxiv.org/abs/1801.02774.

Robert Geirhos, Patricia Rubisch, Claudio Michaelis, Matthias Bethge, Felix A. Wichmann, and Wieland Brendel. Imagenet-trained CNNs are biased towards texture; increasing shape bias improves accuracy and robustness. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=Bygh9j09KX.

Robert Geirhos, Carlos R. M. Temme, Jonas Rauber, Heiko H. Schütt, Matthias Bethge, and Felix A. Wichmann. Generalisation in humans and deep neural networks. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems 31*, pages 7538–7550. Curran Associates, Inc., 2018. URL http://papers.nips.cc/paper/7982-generalisation-in-humans-and-deep-neural-networks.pdf.

Raphael Gontijo Lopes, Dong Yin, Ben Poole, Justin Gilmer, and Ekin D. Cubuk. Improving robustness without sacrificing accuracy with patch gaussian augmentation. *CoRR*, abs/1906.02611, 2019. URL http://arxiv.org/abs/1906.02611.

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

Richard Zhang. Making convolutional networks shift-invariant again. *ICML*, 2019.

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

Samuel Fuller Dodge and Lina J. Karam. Understanding how image quality affects deep neural networks. *2016 Eighth International Conference on Quality of Multimedia Experience (QoMEX)*, pages 1–6, 2016.

Samuel F. Dodge and Lina J. Karam. A study and comparison of human and deep learning recognition performance under visual distortions. *CoRR*, abs/1705.02498, 2017a. URL http://arxiv.org/abs/1705.02498.

Dong Yin, Raphael Gontijo Lopes, Jonathon Shlens, Ekin D. Cubuk, and Justin Gilmer. A Fourier perspective on model robustness in computer vision. *NeurIPS*, 2020.

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

Claudio Michaelis, Benjamin Mitzkus, Robert Geirhos, Evgenia Rusak, Oliver Bringmann, Alexander S Ecker, Matthias Bethge, and Wieland Brendel. Benchmarking robustness in object detection: Autonomous driving when winter is coming. *arXiv preprint arXiv:1907.07484*, 2019.

Norman Mu and Justin Gilmer. MNIST-C: A robustness benchmark for computer vision. *arXiv preprint arXiv:1906.02337*, 2019.

Qizhe Xie, Eduard Hovy, Minh-Thang Luong, and Quoc V. Le. Self-training with noisy student improves imagenet classification. *arXiv preprint arXiv:1911.04252*, 2019a.

Dhruv Mahajan, Ross Girshick, Vignesh Ramanathan, Kaiming He, Manohar Paluri, Yixuan Li, Ashwin Bharambe, and Laurens van der Maaten. Exploring the limits of weakly supervised pretraining. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 181–196, 2018.

Agnieszka Mikołajczyk and Michał Grochowski. Data augmentation for improving deep learning in image classification problem. *2018 International Interdisciplinary PhD Workshop (IIPhDW)*, pages 117–122, 2018.

Samuel F. Dodge and Lina J. Karam. Quality resilient deep neural networks. *CoRR*, abs/1703.08119, 2017b. URL http://arxiv.org/abs/1703.08119.

Jeremy Cohen, Elan Rosenfeld, and Zico Kolter. Certified adversarial robustness via randomized smoothing. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 1310–1320, Long Beach, California, USA, 09–15 Jun 2019. PMLR.

Alhussein Fawzi, Seyed-Mohsen Moosavi-Dezfooli, and Pascal Frossard. Robustness of classifiers: from adversarial to random noise. *NIPS*, 2016.

Logan Engstrom, Dimitris Tsipras, Ludwig Schmidt, and Aleksander Madry. A rotation and a translation suffice: Fooling cnns with simple transformations. *ICML*, 2019.

Daniel Kang, Yi Sun, Tom Brown, Dan Hendrycks, and Jacob Steinhardt. Transfer of adversarial robustness between perturbation types. *CoRR*, abs/1905.01034, 2019. URL http://arxiv.org/abs/1905.01034.

Cihang Xie, Yuxin Wu, Laurens van der Maaten, Alan L. Yuille, and Kaiming He. Feature denoising for improving adversarial robustness. *CVPR*, 2019b.

Matt Jordan, Naren Manoj, Surbhi Goel, and Alexandros G Dimakis. Quantifying perceptual distortion of adversarial examples. *arXiv preprint arXiv:1902.08265*, 2019.

Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, Omar Fawzi, and Pascal Frossard. Universal adversarial perturbations. *CVPR*, 2017.

Jamie Hayes and George Danezis. Machine learning as an adversarial service: Learning black-box adversarial examples. *CoRR*, abs/1708.05207, 2017. URL http://arxiv.org/abs/1708.05207.

Jan Hendrik Metzen. Universality, robustness, and detectability of adversarial perturbations under adversarial training. *https://openreview.net/forum?id=SyjsLqxR-*, 2018.

Ali Shafahi, Mahyar Najibi, Zheng Xu, John P. Dickerson, Larry S. Davis, and Tom Goldstein. Universal adversarial training. *CoRR*, abs/1811.11304, 2018. URL http://arxiv.org/abs/1811.11304.

Chaithanya Kumar Mummadi, Thomas Brox, and Jan Hendrik Metzen. Defending against universal perturbations with shared adversarial training. *ICCV*, 2019.

Julien Pérolat, Mateusz Malinowski, Bilal Piot, and Olivier Pietquin. Playing the game of universal adversarial perturbations. *CoRR*, abs/1809.07802, 2018. URL http://arxiv.org/abs/1809.07802.

Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in PyTorch. In *NIPS Autodiff Workshop*, 2017.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529, 2015.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Logan Engstrom, Andrew Ilyas, and Anish Athalye. Evaluating and understanding the robustness of adversarial logit pairing. *CoRR*, abs/1807.10272, 2018. URL https://arxiv.org/abs/1807.10272.

Ali Shafahi, Mahyar Najibi, Amin Ghiasi, Zheng Xu, John Dickerson, Christoph Studer, Larry S Davis, Gavin Taylor, and Tom Goldstein. Adversarial training for free! *arXiv preprint arXiv:1904.12843*, 2019.

Jérôme Rony, Luiz G Hafemann, Luiz S Oliveira, Ismail Ben Ayed, Robert Sabourin, and Eric Granger. Decoupling direction and norm for efficient gradient-based l2 adversarial attacks and defenses. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4322–4330, 2019.

Ekin D Cubuk, Barret Zoph, Dandelion Mane, Vijay Vasudevan, and Quoc V Le. Autoaugment: Learning augmentation policies from data. *arXiv preprint arXiv:1805.09501*, 2018.

Eric Wong, Frank R. Schmidt, Jan Hendrik Metzen, and J. Zico Kolter. Scaling provable adversarial defenses. *CoRR*, abs/1805.12514, 2018. URL http://arxiv.org/abs/1805.12514.

APPENDIX

A RELATED WORK

Robustness against common corruptions Several recent publications study the vulnerability of DNNs to common corruptions. Dodge and Karam (2016) find that state-of-the-art image recognition networks are particularly vulnerable to blur and Gaussian noise. Two recent studies compare humans and DNNs on recognizing corrupted images, showing that DNN performance drops much faster than human performance for increased perturbation sizes (Dodge and Karam, 2017a; Geirhos et al., 2018). Yin et al. (2020) study the Fourier properties of common corruptions and link them to the robustness of differently trained classifiers.

Hendrycks and Dietterich (2019) introduce corrupted versions of standard datasets denoted as ImageNet-C, Tiny ImageNet-C and CIFAR10-C as standardized benchmarks for machine learning models and show that while state-of-the-art networks like ResNet50 are more accurate than outdated ones like AlexNet, their robustness is still negligible compared to humans. Similarly, common corruptions have been applied to and evaluated on COCO-C, Pascal-C, Cityscapes-C (Michaelis et al., 2019) and MNIST-C (Mu and Gilmer, 2019).

There have been attempts to increase robustness against common corruptions. Zhang (2019) integrate an anti-aliasing module from the signal processing domain in the ResNet50 architecture to restore the shift-equivariance which can get lost in deep CNNs. This results both in increased accuracy on clean data and increased generalization to corrupted image samples. Concurrent work to ours demonstrates that having more training data (Xie et al., 2019a; Mahajan et al., 2018) or using stronger backbones (Xie et al., 2019a; Michaelis et al., 2019) can significantly improve model performance on common corruptions.

A popular method to decrease overfitting and help the network generalize better to unseen data is to augment the training dataset by applying a set of (randomized) manipulations to the images (Mikołajczyk and Grochowski, 2018). Furthermore, augmentation methods have also been applied to make the models more robust against image corruptions (Geirhos et al., 2019). Geirhos et al. (2018) train ImageNet classifiers against a fixed set of corruptions but find no generalized robustness against unseen corruptions. However, they considered vastly higher noise severities than us. A similar observation is made by (Dodge and Karam, 2017b). In a follow-up study, Geirhos et al. (2019) show that recognition models are biased towards texture and suggest this bias as one source of susceptibility for corruptions. They demonstrate that an increased shape bias also leads to increased accuracy on corrupted images. Hendrycks et al. (2020) is concurrent work to ours where the authors propose a data augmentation strategy which relies on combining and mixing augmentation chains. They also report strong robustness increases on ImageNet-C.

Augmentation with Gaussian noise has been used as a regularizer for smoothing the decision boundary of the classifier and was shown to be a provable adversarial defense (Cohen et al., 2019). Conceptually, Ford et al. (2019) is the closest study to our work, since they also apply Gaussian noise to images to increase corruption robustness. They observe a low relative improvement in accuracy on corrupted images whereas we were able to outperform all previous baselines on the commonly used ResNet50 architecture.[1] They use a different architecture (InceptionV3 versus our ResNet50) and train a new model from scratch whereas we fine-tune a pretrained model. Another methodological difference is that we split every batch evenly in clean data and data augmented by Gaussian noise whereas they sample the standard deviation uniformly between 0 and one specific value and add noise to each image. Lopes et al. (2019) restrict the Gaussian noise to small image patches which improves accuracy but does not yield state-of-the-art performance on the ResNet50 architecture.

Link between adversarial robustness and common corruptions There is currently no agreement on whether adversarial training increases robustness against common corruptions in the literature. Hendrycks and Dietterich (2019) report a robustness increase on common corruptions due to adversarial logit pairing on Tiny ImageNet-C. Ford et al. (2019) suggest a link between adversarial robustness and robustness against common corruptions, claim that increasing one robustness type should simultaneously increase the other, but report mixed results on MNIST and CIFAR10-C. Addi-

[1] To compare with Ford et al. (2019), we evaluate our approach for an InceptionV3 architecture, see our results in Appendix J.

tionally, they also observe large drops in accuracy for adversarially trained networks and networks trained with Gaussian data augmentation compared to a vanilla classifier on certain corruptions. They do not evaluate adversarially robust classifiers on ImageNet. Fawzi et al. (2016) show that curvature constraints can both improve robustness against adversarial and random perturbations but they only present results on vanilla networks. On the other hand, Engstrom et al. (2019) report that increasing robustness against adversarial ℓ_∞ attacks does not increase robustness against translations and rotations, but they do not present results on noise. Kang et al. (2019) study robustness transfer between models trained against ℓ_1, ℓ_2, ℓ_∞ adversaries / elastic deformations and JPEG artifacts. They observe that adversarial training increases robustness against elastic and JPEG corruptions on a 100-class subset of ImageNet. This result contradicts our findings on full ImageNet as we see a slight decline in accuracy on those two classes for the adversarially trained model from (Xie et al., 2019b) and severe drops in accuracy on other corruptions. Jordan et al. (2019) show that adversarial robustness does not transfer easily between attack classes.

Universal adversarial perturbations Universal adversarial perturbations (UAPs) (Moosavi-Dezfooli et al., 2017) are perturbations which, if added to any image, fool a given recognition model. This contrasts with regular adversarial perturbations, which need to be designed specifically for every single image. Hayes and Danezis (2017) generate UAPs by training so-called universal adversarial networks (UANs). They also train the classifier jointly with the UAN but manage to only slightly increase robustness against UAPs. Other defenses against UAPs are similarly based on adversarial training (Metzen, 2018; Shafahi et al., 2018; Mummadi et al., 2019; Pérolat et al., 2018).

UAPs are very different from our adversarial noise setting in that UAPs can learn perturbations with global, image-wide features while our adversarial noise is identically distributed over pixels and thus inherently local.

ImageNet-C The ImageNet-C benchmark[2] (Hendrycks and Dietterich, 2019) is a conglomerate of 15 diverse corruption types that were applied to the validation set of ImageNet. The corruptions are organized into four main categories: Noise, Blur, Weather, and Digital and have five levels of severities to reflect the varying intensities of common corruptions. The MNIST-C benchmark is created similarly to ImageNet-C (Mu and Gilmer, 2019) with a slightly different set of corruptions. Our main evaluation metric for both benchmarks is the Top-1 accuracy on corrupted images for each noise category averaged over the severities; we also report the Top-5 accuracy on ImageNet-C. Since some works report the 'mean Corruption Error' (mCE) instead of accuracy, we also include results on mCE in Appendix H.

[2]For the evaluation, we use the JPEG compressed images from github.com/hendrycks/robustness as is advised by the authors to ensure reproducibility. We note that Ford et al. (2019) report a decrease in performance when the compressed JPEG files are used as opposed to applying the corruptions directly in memory without compression artifacts.

B THE NOISE GENERATOR

Formal definition We model the samples from $p_\phi(\boldsymbol{\delta})$ as the output of the Noise Generator neural network $g_\phi : \mathbb{R}^N \to \mathbb{R}^N$ which gets its input from a normal distribution $\boldsymbol{\delta} = g_\phi(\boldsymbol{z})$ where $\boldsymbol{z} \sim \mathcal{N}(\mathbf{0}, \mathbb{1})$. We enforce the independence property of $p_\phi(\boldsymbol{\delta}) = \prod_n p_\phi(\delta_n)$ by constraining the network architecture of the noise generator g_ϕ to only consist of convolutions with 1x1 kernels. Lastly, the projection onto a sphere $||\boldsymbol{\delta}||_2 = \epsilon$ is achieved by scaling the generator output with a scalar while clipping $\boldsymbol{x} + \boldsymbol{\delta}$ to the valid range $[0, 1]^N$. This fixed size projection (hyper-parameter) is motivated by the fact that Gaussian noise training with a single, fixed σ achieved the highest accuracy.

Architecture The architecture of the noise generator is displayed in Table 2. The noise generator g_ϕ has four 1x1 convolutional layers with ReLU activations and one residual connection from input to output. The convolutional weights are initialized such that the noise generator outputs a Gaussian distribution. The number of color channels is indicated by C. The noise generator only uses kernels with a size of 1 and thus produces spatially uncorrelated noise. With the stride being 1 and no padding, the spatial dimensions are preserved in each layer.

Layer	Shape
Conv2D + ReLU	$20 \times 1 \times 1$
Conv2D + ReLU	$20 \times 1 \times 1$
Conv2D + ReLU	$20 \times 1 \times 1$
Conv2D	$C \times 1 \times 1$

Table 2: Architecture of the noise generator.

C IMPLEMENTATION DETAILS AND HYPER-PARAMETERS

We use PyTorch (Paszke et al., 2017) for all of our experiments.

Preprocessing MNIST images are preprocessed such that their pixel values lie in the range $[0, 1]$. Preprocessing for ImageNet is performed in the standard way for PyTorch ImageNet models from the model zoo by subtracting the mean $[0.485, 0.456, 0.406]$ and dividing by the standard deviation $[0.229, 0.224, 0.225]$. We add Gaussian, adversarial and Speckle noise before the preprocessing step, so the noisy images are first clipped to the range $[0, 1]$ of the raw images and then preprocessed before being fed into the model.

C.1 IMAGENET EXPERIMENTS

We evaluate all proposed methods for ImageNet-C on the ResNet50 architecture for better comparability to previous methods, e.g. (Geirhos et al., 2019; Lopes et al., 2019; Zhang, 2019). For all ImageNet experiments, we used a pretrained ResNet50 architecture from `https://pytorch.org/docs/stable/torchvision/models.html`. We fine-tuned the model with SGD-M using an initial learning rate of 0.001, which corresponds to the last learning rate of the PyTorch model training, and a momentum of 0.9. After convergence, we decayed the learning rate once by a factor of 10 and continued the training. Decaying the learning rate was highly beneficial for the model performance. We tried decaying the learning rate a second time, but this did not bring any benefits in any of our experiments. We used a batch size of 70 for all our experiments. We have also tried to use the batch sizes 50 and 100, but did not see major effects.

Gaussian noise We trained the models until convergence. The total number of training epochs varied between 30 and 90 epochs.

Speckle noise We used the Speckle noise implementation from `https://github.com/hendrycks/robustness/blob/master/ImageNet-C/create_c/make_imagenet_c.py`, line 270. The model trained with Speckle noise converged faster than with Gaussian data augmentation and therefore, we only trained the model for 10 epochs.

Adversarial Noise Training To maintain high classification accuracy on clean samples, we sample every mini-batch so that they contain 50% clean data and perturb the rest. The current state of the noise generator is used to perturb 30% of this data and the remaining 20% are augmented with samples chosen randomly from previous distributions. For this, the noise generator states are saved at regular intervals. The latter method is inspired by experience replay from reinforcement learning (Mnih et al., 2015) and is used to keep the classifier from forgetting previous adversarial noise patterns.

To prevent the noise generator from being stuck in a local minimum, we halt the Adversarial Noise Training (ANT) at regular intervals and train a new noise generator from scratch. This noise generator is trained against the current state of the classifier to find a current optimum. The new noise generator replaces the former noise generator in the ANT. This technique has proven crucial to train a robust classifier.

The adversarial noise generator was trained with the Adam optimizer with a learning rate of 0.0001. We have replaced the noise generator every 0.33 epochs. We set the ϵ-sphere to control the size of the perturbation to 135.0 which on average corresponds to the ℓ_2-size of a perturbation caused by additive Gaussian noise sampled from $\mathcal{N}(0, 0.5^2 \cdot \mathbb{1})$. We have trained the classifier until convergence for 80 epochs.

C.2 MNIST EXPERIMENTS

For the MNIST experiments, we used the same model architecture as Madry et al. (2017) for our ANT and GNT for comparability. For ANT, our learning rate for the generator was between 10^{-4} and 10^{-5}, and equal to 10^{-3} for the classifier. We used a batch size of 300. As an optimizer, we used SGD-M with a momentum of 0.9 for the classifier and Adam (Kingma and Ba, 2014) for the generator. The splitting of batches in clean, noisy and history was equivalent to the ImageNet experiments. The optimal ϵ hyper-parameter was determined with a line search similar to the optimal σ of the Gaussian

noise; we found $\epsilon = 10$ to be optimal. The parameters for the Gaussian noise experiments were equivalent. Both models were trained until convergence (around 500-600 epochs). GNT and ANT were performed on a pretrained network.

D (IN-)EFFECTIVENESS OF REGULAR ADVERSARIAL TRAINING TO INCREASE ROBUSTNESS TOWARDS COMMON CORRUPTIONS

We evaluate whether robustness against regular adversarial examples generalizes to robustness against common corruptions. We display the Top-1 accuracy of vanilla and adversarially trained models in Table 3. For all tested models, we find that regular ℓ_∞ adversarial training can strongly decrease the robustness towards common corruptions, especially for the corruption types Fog and Contrast. Universal adversarial training (Shafahi et al., 2018), on the other hand, leads to severe drops on some corruptions but the overall accuracy on ImageNet-C is slightly increased relative to the vanilla baseline model (AlexNet). Nonetheless, the absolute ImageNet-C accuracy of 22.2% is still very low. These results disagree with two previous studies which reported that (1) adversarial logit pairing[3] (ALP) increases robustness against common corruptions on Tiny ImageNet-C (Hendrycks and Dietterich, 2019), and that (2) adversarial training can increase robustness on the CIFAR10-C data set (Ford et al., 2019).

We evaluate adversarially trained models on MNIST-C and present the results and their discussion in Appendix I. The results on MNIST-C show the same tendency as on ImageNet-C: adversarially trained models have lower accuracy on MNIST-C and thus indicate that adversarial robustness does not transfer to robustness against common corruptions. This corroborates the results of (Ford et al., 2019) on MNIST who also found that an adversarially robust model had decreased robustness towards a set of common corruptions.

model	IN-C	IN-C w/o Noises
Vanilla RN50	39.2%	42.3%
Adv. Training (Shafahi et al., 2019)	29.1%	32.0%
Vanilla RN152	45.0%	47.9%
Adv. Training (Xie et al., 2019b)	35.0%	35.9%
Vanilla AlexNet	21.1%	23.9%
Universal Adv. Training (Shafahi et al., 2018)	22.2%	23.1%

Table 3: Top-1 accuracy on ImageNet-C and ImageNet-C without the Noise categories (higher is better). Regular adversarial training decreases robustness towards common corruptions; universal adversarial training seems to slightly increase it.

Detailed results on the evaluation of robustness due to regular adversarial training We find that standard adversarial training against minimal adversarial perturbations in general does not increase robustness against common corruptions. While some early results on CIFAR-10 by Ford et al. (2019) and Tiny ImageNet-C by Hendrycks and Dietterich (2019) suggest that standard adversarial training might increase robustness to common corruptions, we here observe the opposite: Adversarially trained models have lower robustness against common corruptions. An adversarially trained ResNet152 with an additional denoising layer[4] from Xie et al. (2019b) has lower accuracy across almost all corruptions except Snow and Pixelations. On some corruptions, the accuracy of the adversarially trained model decreases drastically, e.g. from 49.1% to 4.6% on Fog or 42.8% to 9.3% on Contrast. Similarly, the adversarially trained ResNet50[5] from [Shafahi et al., 2019] shows a substantial decrease in performance on common corruptions compared with a vanilla trained model.

An evaluation of a robustified version of AlexNet[5] (Shafahi et al., 2018) that was trained with the Universal Adversarial Training scheme on ImageNet-C shows that achieving robustness against universal adversarial perturbations does not noticeably increase robustness towards common corruptions (22.2%) compared with a vanilla trained model (21.1%).

[3]Note that ALP was later found to not increase adversarial robustness (Engstrom et al., 2018).

[4]Model weights from `https://github.com/facebookresearch/ImageNet-Adversarial-Training`

[5]Model weights were kindly provided by the authors.

Model	All	Noise (Compressed)			Blur (Compressed)			
		Gaussian	Shot	Impulse	Defocus	Glass	Motion	Zoom
Vanilla RN50	39.2	29.3	27.0	23.8	38.7	26.8	38.7	36.2
AT (Shafahi et al., 2019)	29.1	20.5	19.1	12.4	21.4	30.8	30.4	31.4
Vanilla RN152	45.0	35.7	34.3	29.6	45.1	32.8	48.4	40.5
AT (Xie et al., 2019b)	35.0	35.2	34.4	24.8	22.1	31.7	30.9	32.0
Vanilla AlexNet	21.1	11.4	10.6	7.7	18.0	17.4	21.4	20.2
UAT (Shafahi et al., 2018)	22.2	20.1	19.1	16.2	13.1	21.6	19.7	19.2

Model	Weather (Compressed)				Digital (Compressed)			
	Snow	Frost	Fog	Brightness	Contrast	Elastic	Pixelate	JPEG
Vanilla RN50	32.5	38.1	45.8	68.0	39.1	45.2	44.8	53.4
AT (Shafahi et al., 2019)	24.4	25.6	5.8	51.1	7.8	45.4	53.4	56.3
Vanilla RN152	38.7	43.9	49.1	71.2	42.8	51.1	50.5	60.5
AT (Xie et al., 2019b)	42.0	40.4	4.6	58.8	9.3	47.2	54.1	58.0
Vanilla AlexNet	13.3	17.3	18.1	43.5	14.7	35.4	28.2	39.4
UAT (Shafahi et al., 2018)	13.8	18.3	4.3	36.5	4.8	36.8	42.3	47.1

Table 4: Average Top-1 accuracy over 5 severities of common corruptions on ImageNet-C in percent. A high accuracy on a certain corruption type indicates high robustness of a classifier on this corruption type, so higher accuracy is better. Adversarial training (AT) decreases the accuracy on common corruptions, especially on the corruptions Fog and Contrast. Universal Adversarial Training (UAT) slightly increases the overall performance.

Generalization of robustness towards common corruptions to adversarial robustness As regular adversarial training can decrease the accuracy on common corruptions, it is also interesting to check what happens vice-versa: How does a model which is robust on common corruptions behave under adversarial attacks? Both our ANT and GNT models have slightly increased ℓ_2 and ℓ_∞ robustness scores compared to a vanilla trained model, see Table 5. We tested this using the white-box attacks PGD (Madry et al., 2017) and DDN (Rony et al., 2019). Note that, of course, adversarially trained models still have significantly higher ℓ_2 and ℓ_∞ robustness.

model	clean acc. [%]	ℓ_2 acc. [%]	ℓ_∞ acc. [%]
Vanilla RN50	75.2	41.1	18.1
GNT$\sigma_{0.5}$	75.3	49.0	28.1
ANT	75.7	50.1	28.6

Table 5: Adversarial robustness on ℓ_2 ($\epsilon = 0.12$) and ℓ_∞ ($\epsilon = 0.001$) compared to a Vanilla ResNet50.

Details for the evaluation of adversarial robustness

ImageNet To evaluate adversarial robustness on ImageNet, we used PGD (Madry et al., 2017) and DDN (Rony et al., 2019). For the ℓ_∞ PGD attack, we allowed for 200 iterations with a step size of 0.0001 and a maximum sphere size of 0.001. For the DDN ℓ_2 attack, we also allowed for 200 iterations, set the sphere adjustment parameter γ to 0.02 and the maximum epsilon to 0.125. We note that for both attacks increasing the number of iterations from 100 to 200 did not make a significant difference in robustness of our tested models. The results on adversarial robustness on ImageNet can be found in the main paper in Table 5.

MNIST To evaluate adversarial robustness on MNIST, we also used PGD (Madry et al., 2017) and DDN (Rony et al., 2019). For the ℓ_∞ PGD attack, we allowed for 100 iterations with a step size of 0.01 and a maximum sphere size of 0.1. For the DDN ℓ_2 attack, we also allowed for 100 iterations, set the sphere adjustment parameter γ to 0.05 and the maximum epsilon to 1.5. All models have the same architecture as Madry et al. (2017). The results on adversarial robustness on MNIST can be found in Table 6.

model	clean acc. [%]	ℓ_2 acc. [%]	ℓ_∞ acc. [%]
Vanilla	99.1	73.2	55.8
GNT$\sigma_{0.5}$	99.3	89.2	73.6
ANT	99.4	90.4	76.3

Table 6: Adversarial robustness on MNIST on ℓ_2 ($\epsilon = 1.5$) and ℓ_∞ ($\epsilon = 0.1$) compared to a Vanilla CNN.

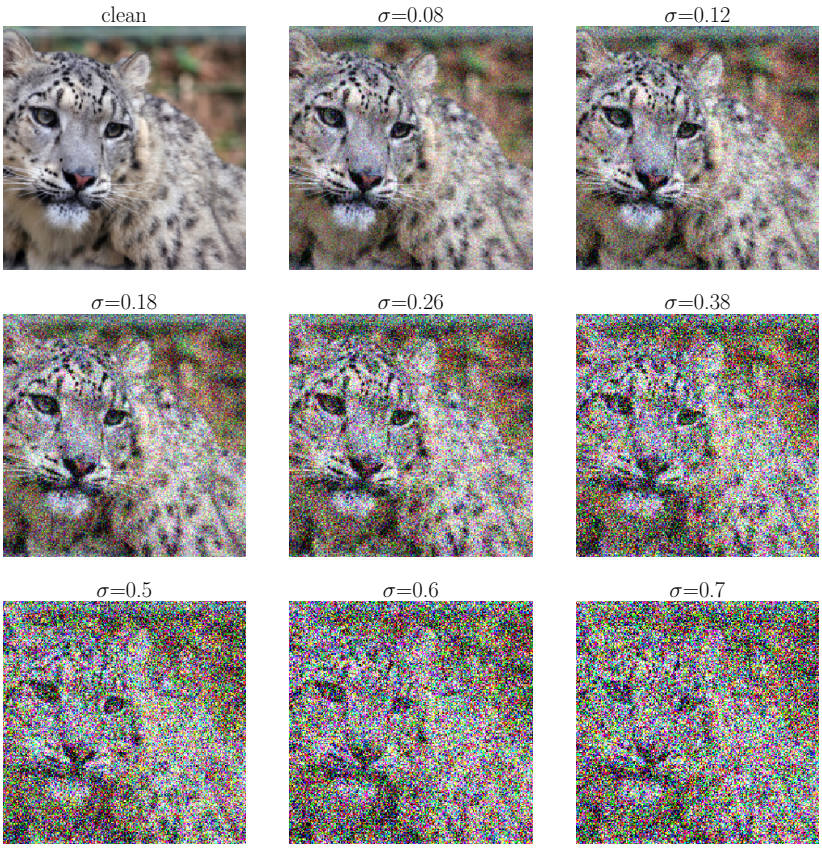

Figure 3. Example images with different σ-levels of additive Gaussian noise on ImageNet.

E EXAMPLE IMAGES FOR ADDITIVE GAUSSIAN NOISE

Example images with additive Gaussian noise of varying standard deviation σ are displayed in Fig. 3. The considered σ-levels correspond to those studied in section 3 in the main paper.

F EVALUATION OF THE SEVERITY OF ADVERSARIAL NOISE AS AN ATTACK

In this section, we try to answer the question: Can we learn the most severe uncorrelated additive noise distribution for a classifier? Following the success of simple uncorrelated Gaussian noise data augmentation (section 3) and the ineffectiveness of regular adversarial training (Appendix D) which allows for highly correlated patterns, we restrict our learned noise distribution to be sampled independently for each pixel. We denote this learned adversarial noise distribution $p_\phi(\boldsymbol{\delta})$ as adversarial noise (AN, section 2).

Evaluation of noise robustness We evaluate the robustness of a model by sampling a Gaussian noise vector $\boldsymbol{\delta}$. We then do a line search along the direction $\boldsymbol{\delta}$ starting from the original image $\boldsymbol{x}$ until it is misclassified. We denote the resulting minimal perturbation as $\boldsymbol{\delta}_{\text{min}}$. The robustness of a model is then denoted by the median[6] over the test set

$$\epsilon^* = \underset{\boldsymbol{x},y\sim\mathcal{D}}{\text{median}}\ ||\boldsymbol{\delta}_{\text{min}}||_2, \tag{4}$$

with $f_\theta(\boldsymbol{x}+\boldsymbol{\delta}_{\text{min}}) \neq y$ and $\boldsymbol{x}+\boldsymbol{\delta}_{\text{min}} \in [0,1]^N$. Note that a higher ϵ^* denotes a more robust classifier. To test the robustness against adversarial noise, we train a new noise generator at the end of the Adversarial Noise Training until convergence and evaluate it according to Eq. (4).

To measure the effectiveness of our adversarial noise, we report the median perturbation size ϵ^* that is necessary for a misclassification for each image in the test set. In Table 7, we see that our AN is much more effective at fooling the classifier compared to Gaussian and uniform noise. This is also reflected qualitatively in the noisy images in Fig. 1 where we show images at the decision boundary: The amount of noise to fool the classifier is smaller in the right-most image produced by the generative network than in the central images (Gaussian and uniform noise).

model	ϵ^*_{GN}	ϵ^*_{UN}	ϵ^*_{AN}
Vanilla RN50	39.0	39.1	16.2

Table 7: Median ℓ_2 perturbation size ϵ^* that is required to misclassify an image for Gaussian (GN), uniform (UN) and adversarial noise (AN). A lower ϵ^* indicates a more severe noise, since on average, a smaller perturbation size is sufficient to fool a classifier.

G BASELINES FOR IMAGENET-C

The ImageNet-C benchmark has been published recently and we use all baselines we could find for a ResNet50 architecture:

1. Shift Inv: The model is modified to enhance shift-equivariance using anti-aliasing (Zhang, 2019).[7]
2. Patch GN: The model was trained on Gaussian patches (Lopes et al., 2019). Since no model weights are released, we can only include their Top-1 ImageNet-C accuracy values from their paper (and not the Top-5).
3. SIN+IN: The model was trained on a stylized version of ImageNet (Geirhos et al., 2019).[8]
4. AugMax: Hendrycks et al. (2020) trained their model using diverse augmentations.[9] They use image augmentations from AutoAugment (Cubuk et al., 2018) and exclude the contrast, color, brightness, sharpness, and Cutout operations to make sure that the test set of ImageNet-C is disjoint from the training set. However, they use the Posterize operation which, as we argue, is visually similar to the JPEG corruption in ImageNet-C (see Appendix K). Additionally, it should be noted that JPEG compression is also used in conjunction with every image in ImageNet-C. As shown by Ford et al. (2019), evaluating on a non-compressed

[6] Samples for which no ℓ_2-distance allows us to manipulate the classifier's decision contribute a value of ∞ to the median.

[7] Weights were taken from github.com/adobe/antialiased-cnns.

[8] Weights were taken from github.com/rgeirhos/texture-vs-shape.

[9] Weights were taken from github.com/google-research/augmix.

version of ImageNet-C affects model performance. Therefore, we argue that the training dataset as used in AugMix is not fully disjoint from the test set of ImageNet-C. Following the line of argumentation above, we put their accuracy values in brackets.

H DETAILED IMAGENET-C RESULTS

We show detailed results on individual corruptions in Table 8 in accuracy and in Table 9 in mCE for differently trained models. In Fig. 4, we show the degradation of accuracy for different severity levels. To avoid clutter, we only show results for a vanilla trained model, for the previous state of the art SIN+IN (Geirhos et al., 2019), for several Gaussian trained models and for the overall best model ANT+SIN.

The Corruption Error (Hendrycks and Dietterich, 2019) is defined as

$$\mathrm{CE}_c^f = \left(\sum_{s=1}^{5} E_{s,c}^f\right) \Big/ \left(\sum_{s=1}^{5} E_{s,c}^{\mathrm{AlexNet}}\right), \tag{5}$$

where $E_{s,c}^f$ is the Top-1 error of a classifier f for a corruption c with severity s. The mean Corruption error (mCE) is taken by averaging over all corruptions.

		Noise			Blur				Weather				Digital			
model	mean	Gauss	Shot	Impulse	Defocus	Glass	Motion	Zoom	Snow	Frost	Fog	Bright	Contrast	Elastic	Pixel	Jpeg
Vanilla RN50	39	29	27	24	39	27	39	36	33	38	46	68	39	45	45	53
Shift Inv	42	36	34	30	40	29	38	39	33	40	48	68	42	45	49	57
Patch GN	44	45	43	42	38	26	39	38	30	39	**54**	67	39	**52**	47	56
SIN+IN	45	41	40	37	43	32	45	36	41	42	47	67	43	50	56	58
AugMix	48	41	41	38	**48**	35	**54**	**49**	40	44	47	69	**51**	**52**	**57**	60
Speckle	46	55	58	49	43	32	40	36	34	41	46	68	41	47	49	58
$\mathrm{GNT}_{\mathrm{mult}}$	49	**67**	65	**64**	43	33	41	37	34	42	45	68	41	48	50	60
$\mathrm{GNT}\sigma_{0.5}$	49	58	59	57	47	**38**	43	42	35	44	44	68	39	50	55	**62**
ANT	51	65	**66**	**64**	47	37	43	40	36	46	44	**70**	43	49	55	**62**
ANT+SIN	**52**	64	65	63	46	**38**	46	39	**42**	**47**	49	69	47	50	**57**	60

Table 8: Average Top-1 accuracy over 5 severities of common corruptions on ImageNet-C in percent obtained by different models; higher is better.

		Noise			Blur				Weather				Digital			
model	mCE	Gauss	Shot	Impulse	Defocus	Glass	Motion	Zoom	Snow	Frost	Fog	Bright	Contrast	Elastic	Pixel	Jpeg
Vanilla	77	80	82	83	75	89	78	80	78	75	66	57	71	85	77	77
SIN	69	66	67	68	70	82	69	80	68	71	65	58	66	78	62	70
Patch GN	71	62	63	62	75	90	78	78	81	74	**57**	59	71	**74**	74	72
Shift Inv.	73	73	74	76	74	86	78	77	77	72	63	56	68	86	71	71
AugMix	65	67	66	68	**64**	79	**59**	**64**	69	68	65	**54**	**57**	**74**	60	65
Speckle	68	51	47	55	70	83	77	80	76	71	66	57	70	82	71	69
$\mathrm{GNT}_{\mathrm{mult}}$	65	**37**	39	**39**	69	81	76	79	76	70	67	56	69	81	69	66
$\mathrm{GNT}\sigma_{0.5}$	64	46	46	47	65	**75**	72	74	75	68	69	57	71	78	63	63
ANT	62	39	**38**	**39**	65	77	72	75	74	66	68	53	67	78	62	**62**
ANT+SIN	**61**	40	39	40	65	76	69	76	**67**	**64**	62	55	63	77	**59**	66

Table 9: Average mean Corruption Error (mCE) obtained by different models on common corruptions from ImageNet-C; lower is better.

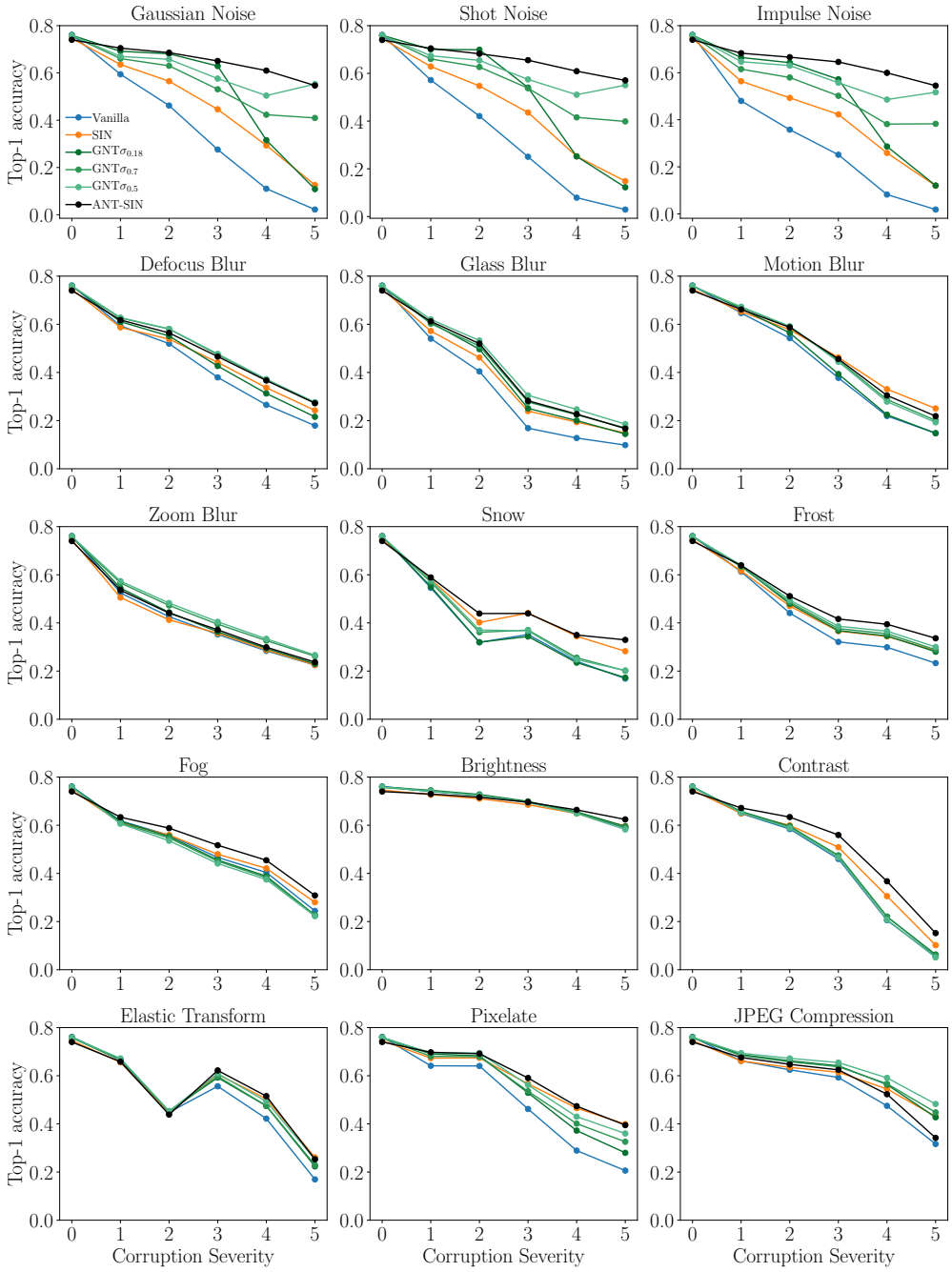

Figure 4. Top-1 accuracy for each corruption type and severity on ImageNet-C.

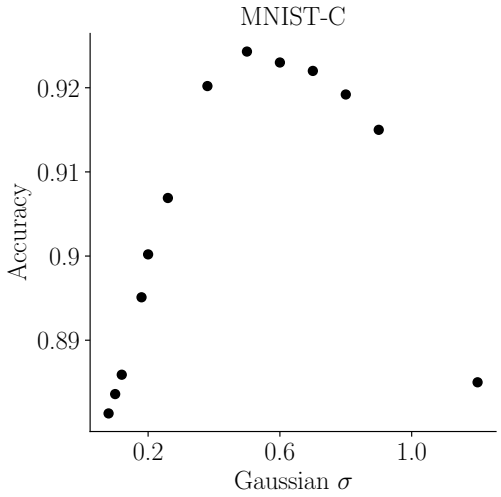

Figure 5. Average accuracy on MNIST-C over all severities and corruptions for different values of sigma σ of the Gaussian noise training (GNT) during training. Each point corresponds to one converged training.

I MNIST-C RESULTS

Similar to the ImageNet-C experiments, we are interested how vanilla, adversarially and noise trained models perform on MNIST-C.

The adversarially robust MNIST model by Wong et al. (2018) was trained with a robust loss function and is among the state of the art in certified adversarial robustness. The other baseline models were trained with Adversarial Training in ℓ_2 (DDN) by Rony et al. (2019) and ℓ_∞ (PGD) by Madry et al. (2017). Our GNT and ANT trained versions are trained as described in the main paper and Appendix C. The results are shown in Table 10. Similar to ImageNet-C, the models trained with GNT and ANT are significantly better than our vanilla trained baseline. Also, regular adversarial training has severe drops and does not lead to significant robustness improvements. We achieve similar results with both approaches and report a new state-of-the-art accuracy on MNIST-C: 92.4%.

As for ImageNet and GNT, we have treated σ as a hyper-parameter. The accuracy on MNIST-C for different values of σ is displayed in Fig. 5 and has a maximum around $\sigma = 0.5$ like for ImageNet.

model	clean acc	mean	Shot	Impulse	Glass Blur	Motion Blur	Shear	Scale	Rotate	Brightness	Translate	Stripe	Fog	Splatter	Dotted Line	Zig Zag	Canny Edges
Vanilla	99.1	86.9	98	96	96	94	98	95	92	88	57	88	50	97	96	86	72
(Madry et al., 2017)	98.5	75.6	98	55	94	94	97	88	92	27	53	40	63	96	78	74	84
Vanilla	98.8	74.3	98	91	96	88	95	80	89	34	45	41	23	96	96	80	63
(Wong et al., 2018)	98.2	68.6	97	65	93	93	94	87	89	11	40	20	25	96	89	61	68
Vanilla	99.5	89.8	98	96	95	97	98	96	94	95	61	89	79	98	98	90	63
DDN Tr (Rony et al., 2019)	99.0	87.0	99	97	96	94	98	91	93	72	55	92	64	99	98	91	66
Vanilla	99.1	86.9	98	96	96	94	98	95	92	88	57	88	50	97	96	86	72
GNT$\sigma_{0.5}$	99.3	92.4	99	99	98	97	98	95	93	98	56	91	91	99	99	96	78
ANT	99.4	92.4	99	99	98	97	98	95	93	98	55	89	91	99	99	96	80

Table 10: Accuracy in percent for the MNIST-C dataset for adversarially robust ((Wong et al., 2018), (Madry et al., 2017), DDN (Rony et al., 2019)) and our noise trained models (GNT and ANT). Vanilla always denotes the same network architecture as its adversarially or noise trained counterpart but with standard training. Note that we used the same network architecture as Madry et al. (2017).

J COMPARISON TO FORD ET AL.

Ford et al. trained an InceptionV3 model from scratch both on clean data from the ImageNet dataset and on data augmented with Gaussian noise (Ford et al., 2019). Since we use a very similar approach, we compare our approach to theirs directly. The results for comparison on ImageNet both for the vanilla and the Gaussian noise trained model can be found in Table 11. Since we use a pretrained model provided by PyTorch and fine-tune it instead of training a new one, the performance of our vanilla trained model differs from the performance of their vanilla trained model, both on clean data and on ImageNet-C. The accuracy on clean data is displayed in Table 12. Another difference between our training and theirs is that we split every batch evenly in clean and data augmented by Gaussian noise with one standard deviation whereas they sample σ uniformly between 0 and one specific value. With our training scheme, we were able to outperform their model significantly on all corruptions except for Elastic, Fog and Brightness.

		Noise (Compressed)			Blur (Compressed)			
model	All	Gaussian	Shot	Impulse	Defocus	Glass	Motion	Zoom
Vanilla InceptionV3 (Ford et al., 2019)	38.8	36.6	34.3	34.7	31.1	19.3	35.3	30.1
Gaussian ($\sigma = 0.4$) (Ford et al., 2019)	42.7	40.3	38.8	37.7	32.9	29.8	35.3	33.1
Vanilla InceptionV3 [ours]	41.6	42.0	40.3	38.5	33.5	27.1	36.1	28.8
GNT$\sigma_{0.4}$ [ours]	49.5	60.8	59.6	59.4	43.8	37.0	42.8	38.4
GNT$\sigma_{0.5}$ [ours]	**50.2**	**61.6**	**60.9**	**60.8**	**44.6**	**37.3**	**44.0**	**39.3**

	Weather (Compressed)				Digital (Compressed)			
model	Snow	Frost	Fog	Brightness	Contrast	Elastic	Pixelate	JPEG
Vanilla InceptionV3 (Ford et al., 2019)	33.1	34.0	52.4	66.0	35.9	47.8	38.2	50.0
Gaussian ($\sigma = 0.4$) (Ford et al., 2019)	36.6	43.5	**52.3**	**67.1**	35.8	**52.2**	47.0	55.5
Vanilla InceptionV3 [ours]	33.5	39.6	42.2	64.2	41.0	43.5	57.4	56.9
GNT$\sigma_{0.4}$ [ours]	35.6	43.7	43.3	64.8	43.0	49.0	59.3	61.7
GNT$\sigma_{0.5}$ [ours]	**37.1**	**44.2**	43.6	64.6	**43.3**	49.4	**59.6**	**61.9**

Table 11: ImageNet-C accuracy for InceptionV3.

model	clean accuracy [%]
Vanilla InceptionV3 (Ford et al., 2019)	75.9
Gaussian ($\sigma = 0.4$) (Ford et al., 2019)	74.2
Vanilla InceptionV3 [ours]	77.2
GNT$\sigma_{0.4}$ [ours]	78.1
GNT$\sigma_{0.5}$ [ours]	77.9

Table 12: Accuracy on clean data for differently trained models.

Figure 6. Example images for the JPEG compression from ImageNet-C and the `PIL.ImageOps.Posterize` operation.

K VISUALIZATION OF POSTERIZE VS JPEG

AugMix (Hendrycks et al., 2020) uses Posterize as one of their operations for data augmentation during training. We argue that Posterize is too similar to the JPEG corruption in ImageNet-C and therefore, the training set is not disjoint from the test set. To visualize our point, we show example images for the JPEG compression in ImageNet-C and `PIL.ImageOps.Posterize` operation in Fig. 6.