Improving robustness against common corruptions by covariate shift adaptation

Steffen Schneider [*1,2] **Evgenia Rusak** [*1,2] **Luisa Eck** [3] **Oliver Bringmann** [†2] **Wieland Brendel** [†2] **Matthias Bethge** [†2]

Abstract

Today's state-of-the-art machine vision models are vulnerable to image corruptions like blurring or compression artefacts, limiting their performance in many real-world applications. We here argue that popular benchmarks to measure model robustness against common corruptions (like ImageNet-C) underestimate model robustness in many (but not all) application scenarios. The key insight is that in many scenarios, multiple unlabeled examples of the corruptions are available and can be used for unsupervised online adaptation. Replacing the activation statistics estimated by batch normalization on the training set with the statistics of the corrupted images consistently improves the robustness across 25 different popular computer vision models. Using the corrected statistics, ResNet-50 reaches 62.2% mCE on ImageNet-C compared to 76.7% without adaptation. With the more robust AugMix model, we improve the state of the art from 56.5% mCE to 51.0% mCE. Even adapting to a single sample improves robustness for the ResNet-50 and AugMix models, and 32 samples are sufficient to improve the current state of the art for a ResNet-50 architecture. We argue that results with adapted statistics should be included whenever reporting scores in corruption benchmarks and other out-of-distribution generalization settings.

1. Introduction

Deep neural networks (DNNs) are known to perform well in the independent and identically distributed (*i.i.d.*) setting when the test and training data are sampled uniformly from the same distribution. However, for many applications this assumption does not hold. In medical imaging, X-ray images or histology slides will differ from the training data if different scanners are being used. In quality assessment, the images might differ from the training data if lighting conditions change or if dirt particles accumulate on the camera. Autonomous cars may face rare weather conditions like sandstorms or big hailstones. While human vision is quite robust to those deviations (Geirhos et al., 2018), modern high-performance machine vision models are often sensitive to such image corruptions.

We argue that current evaluations of model robustness underestimate performance in many (but not all) real-world scenarios. So far, popular image corruption benchmarks like ImageNet-C (IN-C; Hendrycks & Dietterich, 2019) focus only on ad-hoc scenarios in which the tested model has zero prior knowledge about the corruptions it encounters during test time, even if it encounters the same corruption multiple times. In the example of medical images or quality assurance, the image corruptions do not change from sample to sample but are continuously present over a potentially large number of samples. Similarly, autonomous cars will face the same weather condition over a continuous stream of inputs during the same sand- or hailstorm. These (unlabeled) observations can allow recognition models to adapt to the change in the input distribution.

Such unsupervised adaptation mechanisms are studied in the field of domain adaptation (DA), which is concerned with adapting models trained on one domain (the source, here clean images) to another for which only unlabeled samples exist (the target, here the corrupted images). Tools and methods from domain adaptation are thus directly applicable to increase model robustness against common corruptions, but so far no results on popular benchmarks have been reported. The overall goal of this work is to encourage stronger interactions between the currently disjoint fields of domain adaptation and common corruptions.

We here focus on one popular technique in DA, namely adapting batch normalization (BN; Ioffe & Szegedy, 2015) statistics (Li et al., 2017; Schneider et al., 2018; Cariucci et al., 2017; Li et al., 2016). In computer vision, BN is a popular technique for speeding up training and is present in

[*]Equal contribution [†]Equal contribution [1]IMPRS-IS, Germany [2]University of Tübingen, Germany [3]LMU Munich, Germany. Correspondence to: Steffen Schneider <steffen@bethgelab.org>.

Presented at the ICML 2020 Workshop on Uncertainty and Robustness in Deep Learning.

almost all current state-of-the-art image recognition models. BN estimates the statistics of activations for the training dataset and uses them to normalize intermediate activations in the network.

By design, activation statistics obtained during training time do not reflect the statistics of the test distribution when testing in out-of-distribution settings like corrupted images. We investigate and corroborate the hypothesis that high-level distributional shifts from clean to corrupted images largely manifest themselves in a difference of first and second order moments in the internal representations of a deep network, which can be mitigated by adapting BN statistics, i.e. by estimating the BN statistics on the corrupted images. We demonstrate that this simple adaptation alone can greatly increase recognition performance on corrupted images, leading some vanilla recognition models like DenseNet-101 to almost reaching state-of-the-art performance on IN-C without adaptation.

Our core contributions can be summarized as follows:

- We suggest to augment current benchmarks for common corruptions with two additional performance metrics that measure robustness after partial and full unsupervised adaptation to the corrupted images.
- We draw connections to domain adaptation and show that even adapting to a single corrupted sample improves the baseline performance of a ResNet-50 model trained on IN (from 76.7% mCE to 71.4%). Robustness increases with more samples for adaptation and converges to a mCE of 62.24%.
- We show that the robustness of a variety of vanilla models trained on ImageNet (IN; Russakovsky et al., 2015; Deng et al., 2009) substantially increases after adaptation, sometimes approaching the current state-of-the-art performance on IN-C without adaptation.
- Similarly, we show that the robustness of state-of-the-art ResNet-50 models on IN-C consistently increases when adapted statistics are used. We surpass the best non-adapted model (54.2% mCE) by more than 3% points.

2. Adapting batch norm statistics

IN trained models typically make use of batch normalization (BN) layers (Ioffe & Szegedy, 2015) for faster convergence and improved stability during training, which we here adapt as a simple method for covariate shift adaptation. Within these layers, first and second order statistics μ_c, σ_c^2 of the activation tensors $\mathbf{z}_c$ are estimated across the spatial dimensions and samples for each feature map c. BN subtracts the mean μ_c from the activations and divides them by σ_c^2 to normalize the activation statistics. During training, μ_c and σ_c^2 are estimated *per batch*. During evaluation, μ_c and σ_c^2 are estimated *over the whole training dataset*, typically using exponential averaging (Paszke et al., 2017).

Using the BN statistics obtained during training for testing makes the model decisions deterministic but is problematic if the input distribution changes (covariate shift, cf. §A.4). If the activation statistics μ_c, σ_c^2 change on samples from the test domain, then the activations of feature map c are no longer normalized to zero mean and unit variance, breaking a crucial assumption that all downstream layers depend on.

If the covariate shift only causes differences in the first and second order moments of the feature activations $\mathbf{z} = f(\mathbf{x})$, it can be removed by applying normalization using the correct statitics. The distribution of $(f(\mathbf{x}) - \mathbb{E}_s[f(\mathbf{x})])/\mathbb{V}_s[f(\mathbf{x})]$ and $(f(\mathbf{x}) - \mathbb{E}_t[f(\mathbf{x})])/\mathbb{V}_t[f(\mathbf{x})]$ match under this assumption.

Implementing this operation is straight-forward when using BN: it suffices to estimate the BN statistics μ_t, σ_t^2 on (unlabeled) samples from the test data available for adaptation. To improve unreliable estimates for small n, we leverage the train time statistics μ_s, σ_s^2 as a prior and finally normalize with

$$\bar{\mu} = \frac{N\mu_s + n\hat{\mu}_t}{N+n}, \quad \bar{\sigma}^2 = \frac{N\sigma_s^2 + n\hat{\sigma}_t^2}{N+n}. \tag{1}$$

The hyperparameter N controls the trade-off between prior and estimated target statistics and has the intuitive interpretation of a *pseudo sample size* (p. 117, Bishop, 2006) of the training set. The case $N \to \infty$ ignores the statistics on the test set and is equivalent to the standard ad-hoc scenario while $N = 0$ ignores the training statistics.

3. Experimental Setup

We consider all models in the torchvision library for our main experiments and give a detailed overview in §B&G. We consider ImageNet-C (IN-C; Hendrycks & Dietterich, 2019), ImageNet-A (IN-A; Hendrycks et al., 2019), ImageNet-V2 (IN-V2; Recht et al., 2020), and ObjectNet (ON; Barbu et al., 2019) (see additional information in §B.)

For large batch sizes n greater than 100 samples, our method empiricially performs well for $N = 0$ and does not require tuning of any hyperparameters. For small batch sizes, we select the optimal N based on mCE computed on the four holdout corruptions in IN-C provided for this purpose.

4. Results

Adaptation boosts robustness of a vanilla trained ResNet-50 model. We consider the pretrained ResNet-50 architecture from the torchvision library and adapt the

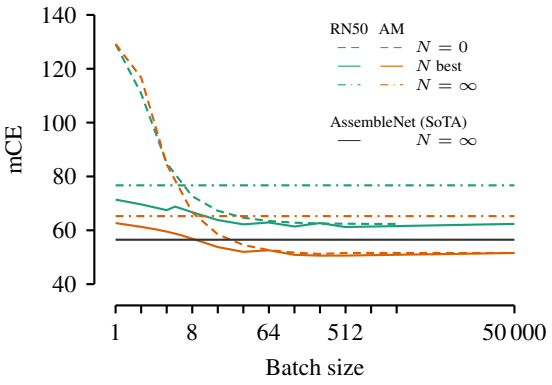

Figure 1: Sample size/performance tradeoff in terms of the mean corruption error (mCE) on IN-C for ResNet-50 and AugMix (AM). Black line corresponds to (non-adapted) state-of-the-art performance of AssembleNet on IN-C.

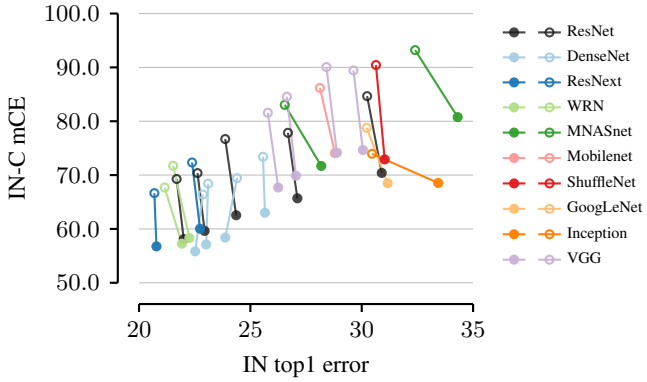

Figure 2: Across 25 torchvision models, the baseline mCE (○) improves with adaptation (●), often on the order of 10 points.

running mean and variance on all corruptions and severities of IN-C for different batch sizes. The results are displayed in Fig. 1 as the dotted green line for the vanilla ResNet-50 baseline and as the full green line with stars indicating the batch size over which the statistics are calculated. For the best choice of N, we see that even adapting to a single sample can suffice to increase robustness, suggesting that even the ad-hoc evaluation scenario can benefit from adaptation. If the training statistics are not used as a prior ($N = 0$) then it takes around 8 samples to surpass the performance of the non-adapted baseline model (76.7% mCE). After around 16 to 32 samples the performance quickly converges to 62.2% mCE, considerably improving the baseline result. These results highlight the practical applicability of batch norm adaptation in basically all application scenarios, independent of the number of available test samples.

Adaptation consistently improves corruption robustness across IN trained models. To evaluate the interaction between architecture and BN adaptation, we evaluate all 25 pre-trained models in the `torchvision` package and visualize the results in Fig. 2. All models are evaluated with N=0 and n=2000. We group models into different families based on their architecture and observe consistent improvements in mCE for all of these families, typically on the order of 10% points. We observe that in both evaluation modes, DenseNets (Huang et al., 2017) exhibit higher corruption robustness despite having a comparable or even smaller number of trainable parameters as the popular ResNets. A take-away from this study is thus that model architecture alone plays an important role for corruption robustness and the ResNet architecture might not be the optimal choice for practical applications.

Adaptation yields new state of the art on IN-C for robust models. We now investigate if BN adaptation also improves the most robust models on IN-C. The results are displayed in Table 1. All models are adapted using n = 50 000 (vanilla) or n=4096 (all other models) and N=0. The performance of all models is considerably higher whenever the BN statistics are adapted. AugMix reaches a new state of the art on IN-C for a ResNet-50 architecture of 51% mCE. Evaluating the performance of AugMix over the number of samples for adaptation (Fig. 1, we find that as little as eight samples are sufficient to improve over AssembleNet (Lee et al., 2020), the current state-of-the-art ResNet-50 model on IN-C.

Large scale pre-training alleviates the need for adaptation. Mahajan et al. (2018) train computer vision models based on the ResNeXt architecture (Xie et al., 2017) on a much larger dataset comprised of 3.5×10^9 Instagram images (IG-3.5B), achieving a 45.7% mCE on IN-C (Orhan, 2019). We re-evaluate these models with our proposed paradigm and summarize the results in Table 2. While we see improvements for the small model pre-trained on IN, these improvements vanish once the model is trained on the full IG-3.5B dataset. This observation also holds for the largest model, suggesting that training on very large datasets might alleviate the need for covariate shift adaptation.

Group Norm and Fixup Initialization outperforms non-adapted BN models, but is worse than BN with covariate shift adaptation. In our experiments so far, we assumed that popular computer vision models for image classification on IN are generally trained by using BN and are hence vulnerable to the observed effects of covariate shift. Recently, Zhang et al. (2019) proposed Fixup initialization of models, alleviating the need for BN layers. We train a model with a ResNet-50 architecture on IN for 100 epochs to obtain a top-1 error of 24.2% and top-5 error of 7.6% (compared to 27.6% reported by Zhang et al. (2019) with shorter training, and the 23.9% obtained by our ResNet-50

Table 1: Adaptation improves mCE (lower is better) and Top1 accuracy (higher is better) on IN-C for different ResNet-50 models and surpasses the previous state of the art without adaptation. We consider $n = 8$ for partial adaptation.

	IN-C mCE (↘)				Top1 accuracy (↗)			
Model	w/o adapt	partial adapt	full adapt	Δ	w/o adapt	partial adapt	full adapt	Δ
Vanilla ResNet-50	76.7	66.7	‡62.2	(−14.5)	39.2	47.2	‡50.7	(+11.5)
SIN (Geirhos et al., 2019)	69.3	63.1	59.5	(−9.8)	45.2	50.3	53.1	(+7.9)
ANT (Rusak et al., 2020)	63.4	55.0	53.6	(−9.8)	50.4	57.0	58.0	(+7.6)
ANT+SIN (Rusak et al., 2020)	60.7	54.6	53.6	(−7.0)	**52.6**	55.0	58.0	(+5.4)
Assemble Net (Lee et al., 2020)	**54.2**	–	52.1	(−2.1)	–	–	59.2	–
AugMix (Hendrycks et al., 2020)	65.3	56.9	**51.0**	(−14.3)	48.3	55.0	**59.8**	(+11.4)

Table 2: Improvements from adapting the BN parameters vanish for models trained with weakly supervised pre-training.

	IN-C mCE (↘)	
ResNeXt101	BN	BN+adapt
32x8d, IN	66.6	56.7 (−9.9)
32x8d, IG-3.5B	51.7	51.6 (−0.1)
32x48d, IG-3.5B	**45.7**	**47.3** (+1.6)

Table 3: Fixup and GN trained models perform better than non-adapted BN models but worse than adapted BN models.

	IN-C mCE (↘)			
Model	Fixup	GN	BN	BN+adapt
ResNet-50	72.0	72.4	76.7	**62.2**
ResNet-101	68.2	67.6	69.0	**59.1**
ResNet-152	67.6	65.4	69.3	**58.0**

baseline trained on batch norm). On IN-C, this model obtains an mCE of 72.0% compared to the 76.7% mCE of the vanilla ResNet-50 model and the 62.2% mCE of our adapted ResNet-50 model (cf. Table 3). Additionally, we train a ResNet-101 and a ResNet-152 with Fixup initialization with similar results. GroupNorm (GN; Wu & He, 2018) has been proposed as a batch-size independent normalization technique. We train a ResNet-50, a ResNet-101 and a ResNet-152 architecture for 100 epochs and evaluate them on IN-C and find results very similar to Fixup. We put additional results in §D for space reasons.

5. Discussion & Conclusion

We showed that reducing covariate shift induced by common image corruptions improves the robustness of computer vision models trained with BN layers, typically by 10–15% points (mCE) on IN-C. Current state-of-the-art models on IN-C can benefit from adaptation, sometimes drastically like AugMix (−14% points mCE). This observation underlines that current benchmark results on IN-C underestimate the corruption robustness that can be reached in many application scenarios where additional (unlabeled) samples are available for adaptation.

Robustness against common corruptions improves even if models are adapted only to a single sample, suggesting that BN adaptation should always be used whenever we expect machine vision algorithms to encounter out-of-domain samples. Most further improvements can be reaped by adapting to 32 to 64 samples, after which additional improvements are minor.

Our empirical results suggest that the performance degradation on corrupted images can mostly be explained by the difference in feature-wise first and second order moments. While this might sound trivial, the performance could also degrade because models mostly extract features susceptible to common corruptions (Geirhos et al., 2020), which could not be fixed without substantially adapting the model weights. The fact that model robustness increases after correcting the BN statistics suggests that the features upon which the models rely on are still present in the corrupted images. The opposite is true in other out-of-domain datasets like IN-A or ObjectNet where our simple adaptation scheme does not substantially improve performance, suggesting that here the main problem is in the features that models have learned to use for prediction.

Current corruption benchmarks emphasize ad hoc scenarios and thus focus and bias future research efforts on these constraints. Unfortunately, the ad hoc scenario does not accurately reflect the information available in many machine vision applications like classifiers in medical computer vision or visual quality inspection algorithms, which typically encounter a similar corruption continuously and could benefit from adaptation. This work is meant to spark more research in this direction by suggesting two suitable evaluation metrics—which we strongly suggest to include in all future evaluations on IN-C—as well as by highlighting the potential that even a fairly simple adaptation mechanism can have for increasing model robustness. We envision future work to also adopt and evaluate more powerful domain adaptation methods on IN-C and to develop new adaptation methods specifically designed to increase robustness against common corruptions.

Acknowledgments and Disclosure of Funding

We thank Julian Bitterwolf, Roland S. Zimmermann, Lukas Schott, Mackenzie W. Mathis, Alexander Mathis, Asim Iqbal, David Klindt, Robert Geirhos and other members of the Bethge and Mathis labs for helpful suggestions for improving our manuscript and providing ideas for additional ablation studies. We thank the International Max Planck Research School for Intelligent Systems (IMPRS-IS) for supporting E.R. and St.S.; St.S. acknowledges his membership in the European Laboratory for Learning and Intelligent Systems (ELLIS) PhD program.

This work was supported by the German Federal Ministry of Education and Research (BMBF) through the Tübingen AI Center (FKZ: 01IS18039A), by the Deutsche Forschungsgemeinschaft (DFG) in the priority program 1835 under grant BR2321/5-2 and by SFB 1233, Robust Vision: Inference Principles and Neural Mechanisms (TP3), project number: 276693517.

The authors declare no conflicts of interests.

Other Versions of this Paper

This paper is short version of the pre-print available at https://arxiv.org/abs/2006.16971.

References

Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., et al. Tensorflow: A system for large-scale machine learning. In *12th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 16)*, pp. 265–283, 2016.

Barbu, A., Mayo, D., Alverio, J., Luo, W., Wang, C., Gutfreund, D., Tenenbaum, J., and Katz, B. Objectnet: A large-scale bias-controlled dataset for pushing the limits of object recognition models. In *Advances in Neural Information Processing Systems 32*, 2019.

Becker, R. A. The variance drain and jensen's inequality. 2012-004, 2012.

Bishop, C. M. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag, Berlin, Heidelberg, 2006. ISBN 0387310738.

Bug, D., Schneider, S., Grote, A., Oswald, E., Feuerhake, F., Schüler, J., and Merhof, D. Context-based normalization of histological stains using deep convolutional features. In *Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support*. Springer, 2017.

Cariucci, F. M., Porzi, L., Caputo, B., Ricci, E., and Bulo, S. R. Autodial: Automatic domain alignment layers. In *2017 IEEE International Conference on Computer Vision (ICCV)*, 2017.

Cubuk, E. D., Zoph, B., Mané, D., Vasudevan, V., and Le, Q. V. Autoaugment: Learning augmentation policies from data. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. Imagenet: A large-scale hierarchical image database. In *Conference on computer vision and pattern recognition (CVPR)*, 2009.

Ford, N., Gilmer, J., Carlini, N., and Cubuk, D. Adversarial examples are a natural consequence of test error in noise. In *International Conference on Machine Learning (ICML)*, 2019.

French, G., Mackiewicz, M., and Fisher, M. H. Self-ensembling for domain adaptation. *CoRR*, abs/1706.05208, 2017.

Geirhos, R., Temme, C. R. M., Rauber, J., Schütt, H. H., Bethge, M., and Wichmann, F. A. Generalisation in humans and deep neural networks. In Bengio, S., Wallach, H., Larochelle, H., Grauman, K., Cesa-Bianchi, N., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 31*, pp. 7538–7550. Curran Associates, Inc., 2018.

Geirhos, R., Rubisch, P., Michaelis, C., Bethge, M., Wichmann, F. A., and Brendel, W. Imagenet-trained CNNs are biased towards texture; increasing shape bias improves accuracy and robustness. In *International Conference on Learning Representations (ICLR)*, 2019.

Geirhos, R., Jacobsen, J.-H., Michaelis, C., Zemel, R., Brendel, W., Bethge, M., and Wichmann, F. A. Shortcut learning in deep neural networks. *CoRR*, abs/2004.07780, 2020.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *Conference on computer vision and pattern recognition (CVPR)*, 2016.

Hendrycks, D. and Dietterich, T. Benchmarking neural network robustness to common corruptions and perturbations. In *International Conference on Learning Representations (ICLR)*, 2019.

Hendrycks, D., Zhao, K., Basart, S., Steinhardt, J., and Song, D. Natural adversarial examples. *CoRR*, abs/1907.07174, 2019.

Hendrycks, D., Mu, N., Cubuk, E. D., Zoph, B., Gilmer, J., and Lakshminarayanan, B. Augmix: A simple data

processing method to improve robustness and uncertainty. In *International Conference on Learning Representations (ICLR)*, 2020.

Huang, G., Liu, Z., and Weinberger, K. Q. Densely connected convolutional networks. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

Ioffe, S. and Szegedy, C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning (ICLR)*, 2015.

Kamann, C. and Rother, C. Benchmarking the robustness of semantic segmentation models. *ArXiv*, abs/1908.05005, 2019.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems (NIPS)*. 2012.

Lee, J., Won, T., and Hong, K. Compounding the performance improvements of assembled techniques in a convolutional neural network. *CoRR*, abs/2001.06268, 2020.

Li, Y., Wang, N., Shi, J., Liu, J., and Hou, X. Revisiting batch normalization for practical domain adaptation. *CoRR*, abs/1603.04779, 2016.

Li, Y., Wang, N., Shi, J., Liu, J., and Hou, X. Revisiting batch normalization for practical domain adaptation. In *International Conference on Machine Learning (ICLR)*, 2017.

Ma, N., Zhang, X., Zheng, H.-T., and Sun, J. Shufflenet v2: Practical guidelines for efficient cnn architecture design. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018.

Mahajan, D., Girshick, R., Ramanathan, V., He, K., Paluri, M., Li, Y., Bharambe, A., and van der Maaten, L. Exploring the limits of weakly supervised pretraining. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018.

Marcel, S. and Rodriguez, Y. Torchvision the machine-vision package of torch. In *ACM International Conference on Multimedia*, 2010.

Merkel, D. Docker: Lightweight linux containers for consistent development and deployment. *Linux J.*, 2014(239), March 2014. ISSN 1075-3583.

Michaelis, C., Mitzkus, B., Geirhos, R., Rusak, E., Bringmann, O., Ecker, A. S., Bethge, M., and Brendel, W. Benchmarking robustness in object detection: Autonomous driving when winter is coming. *CoRR*, abs/1907.07484, 2019.

Mikołajczyk, A. and Grochowski, M. Data augmentation for improving deep learning in image classification problem. In *International Interdisciplinary PhD Workshop (IIPhDW)*, 2018.

Mu, N. and Gilmer, J. MNIST-C: A robustness benchmark for computer vision. *CoRR*, abs/1906.02337, 2019.

Orhan, A. E. Robustness properties of facebook's resnext wsl models. *CoRR*, 2019.

Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., and Lerer, A. Automatic differentiation in PyTorch. In *NIPS Autodiff Workshop*, 2017.

Rebuffi, S.-A., Bilen, H., and Vedaldi, A. Learning multiple visual domains with residual adapters. In *Advances in Neural Information Processing Systems (NIPS)*, 2017.

Recht, B., Roelofs, R., Schmidt, L., and Shankar, V. Do imagenet classifiers generalize to imagenet? In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

Rusak, E., Schott, L., Zimmermann, R., Bitterwolf, J., Bringmann, O., Bethge, M., and Brendel, W. Increasing the robustness of dnns against image corruptions by playing the game of noise. *CoRR*, abs/2001.06057, 2020.

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., et al. Imagenet large scale visual recognition challenge. *International journal of computer vision (IJCV)*, 2015.

Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., and Chen, L.-C. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Conference on computer vision and pattern recognition (CVPR)*, 2018.

Santurkar, S., Tsipras, D., Ilyas, A., and Madry, A. How does batch normalization help optimization? In *Advances in Neural Information Processing Systems (NIPS)*, 2018.

Schneider, S., Ecker, A. S., Macke, J. H., and Bethge, M. Multi-task generalization and adaptation between noisy digit datasets: An empirical study. In *Neural Information Processing Systems (NeurIPS), Workshop on Continual Learning*, 2018.

Schölkopf, B., Janzing, D., Peters, J., Sgouritsa, E., Zhang, K., and Mooij, J. On causal and anticausal learning. In *Proceedings of the 29th International Coference on International Conference on Machine Learning*, ICML'12, pp. 459–466, Madison, WI, USA, 2012. Omnipress. ISBN 9781450312851.

Simonyan, K. and Zisserman, A. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations (ICLR)*, 2015.

Sugiyama, M. and Kawanabe, M. *Machine learning in non-stationary environments: Introduction to covariate shift adaptation*. MIT press, 2012.

Sun, B., Feng, J., and Saenko, K. Correlation alignment for unsupervised domain adaptation. In *Domain Adaptation in Computer Vision Applications*, pp. 153–171. Springer, 2017.

Sun, Y., Wang, X., Liu, Z., Miller, J., Efros, A. A., and Hardt, M. Test-time training for out-of-distribution generalization. *CoRR*, abs/1909.13231, 2019.

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S. E., Anguelov, D., Erhan, D., Vanhoucke, V., and Rabinovich, A. Going deeper with convolutions. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., and Wojna, Z. Rethinking the inception architecture for computer vision. In *Conference on computer vision and pattern recognition (CVPR)*, 2016.

Tan, M., Chen, B., Pang, R., Vasudevan, V., Sandler, M., Howard, A., and Le, Q. V. Mnasnet: Platform-aware neural architecture search for mobile. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

Tange, O. Gnu parallel - the command-line power tool. *;login: The USENIX Magazine*, 36(1):42–47, Feb 2011. URL http://www.gnu.org/s/parallel.

Villani, C. *Optimal transport: old and new*, volume 338. Springer Science & Business Media, 2008.

Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., van der Walt, S. J., Brett, M., Wilson, J., Jarrod Millman, K., Mayorov, N., Nelson, A. R. J., Jones, E., Kern, R., Larson, E., Carey, C., Polat, İ., Feng, Y., Moore, E. W., Vand erPlas, J., Laxalde, D., Perktold, J., Cimrman, R., Henriksen, I., Quintero, E. A., Harris, C. R., Archibald, A. M., Ribeiro, A. H., Pedregosa, F., van Mulbregt, P., and Contributors, S. . . SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272, 2020. doi: https://doi.org/10.1038/s41592-019-0686-2.

Weisstein, E. Standard deviation distribution, 2020. URL https://mathworld.wolfram.com/StandardDeviationDistribution.html.

Wu, Y. and He, K. Group normalization. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018.

Xie, S., Girshick, R., Dollár, P., Tu, Z., and He, K. Aggregated residual transformations for deep neural networks. In *Conference on computer vision and pattern recognition (CVPR)*, 2017.

Zagoruyko, S. and Komodakis, N. Wide residual networks. *CoRR*, abs/1605.07146, 2016.

Zhang, H., Dauphin, Y. N., and Ma, T. Fixup initialization: Residual learning without normalization. *CoRR*, abs/1901.09321, 2019.

Zhang, R. Making convolutional networks shift-invariant again. *International Conference on Machine Learning (ICML)*, 2019.

A. Methods

A.1. Distances and divergences for quantifying domain shift

Besides analyzing the performance drop when evaluating a model using source statistics on a target dataset, we consider the mismatch in model statistics directly. We first take an ImageNet trained model and adapt it to each of the 95 conditions in IN-C. To obtain a more exact estimate of the true statistics, we split the model into multiple stages with only few BN layers per stage and apply the following simple algorithm[1]:

- Start with image inputs $\mathbf{z}_n^0 \leftarrow \mathbf{x}_n$ from the validation set to adapt to, for each $n \in [50000]$.
- Split the model into multiple stages, $h(\mathbf{x}) = (f_m \circ \cdots \circ f_1)(\mathbf{x})$, where each module f_i can potentially contain one or multiple BN layers. We denote the number of BN layers in the i-th module as b_i.
- For each stage $i \in [m]$, repeat b_i times: $\mathbf{z}_n^i \leftarrow f_i(\mathbf{z}_n^{i-1})$ for each n, and update the BN statistics in module $f_i(\mathbf{z}_n^{i-1})$.
- Return h with adapted statistics.

Using this scheme, we get source statistics μ_s and Σ_s for each layer and μ_t and Σ_t for each layer and corruption. In total, we get 96 different collections of statistics across network layers (for IN and the 95 conditions in IN-C). For simplicity, we will not further index the statistics. Note that all covariance matrices considered here are diagonal, which is a further simplification. We expect that our domain shift estimates could be improved by considering the full covariance matrices.

In the following, we will introduce three possible distances and divergences which can be applied between source and target statistics to quantify the effect of common corruptions induced covariate shift. We consider the Wasserstein distance, a normalized version of the Wasserstein distance, and the Jeffrey divergence.

[1]Note that for simplicity, we do not reset the statistics of the remaining $(b_i - i)$ BN layers. This could potentially be adapted in future work.

A.1.1. THE WASSERSTEIN DISTANCE

Given a baseline ResNet-50 model with source statistics μ_s, Σ_s on IN, the Wasserstein distance (cf. Villani, 2008) between the train and test distribution with statistics μ_t, Σ_t is given as

$$W_2^2(p_s, p_t)^2 = \|\boldsymbol{\mu}_s - \boldsymbol{\mu}_t\|_2^2 \tag{2}$$

$$+ \operatorname{Tr}\left(\boldsymbol{\Sigma}_s + \boldsymbol{\Sigma}_t - 2\left(\boldsymbol{\Sigma}_t^{1/2}\boldsymbol{\Sigma}_s\boldsymbol{\Sigma}_t^{1/2}\right)^{1/2}\right). \tag{3}$$

A.1.2. THE SOURCE-NORMALIZED WASSERSTEIN DISTANCE

When estimated for multiple layers across the network, the Wasserstein distance between source and target depends on the overall magnitude of the statistics. Practically, this means the metric is dominated by features with large magnitude (e.g. in the first layer of a neural network, which receives larger inputs).

To mitigate this issue, we normalize both statistics with the source statistics and define the normalized Wasserstein distance as

$$\widetilde{W}_2^2 = W_2^2\left(\boldsymbol{\Sigma}_t\boldsymbol{\Sigma}_s^{-1}, \mathbf{I}, \boldsymbol{\Sigma}_s^{-1/2}\boldsymbol{\mu}_t, \boldsymbol{\Sigma}_s^{-1/2}\boldsymbol{\mu}_s\right) \tag{4}$$

$$= \operatorname{Tr}\left(\mathbf{I} + \boldsymbol{\Sigma}_t\boldsymbol{\Sigma}_s^{-1} - 2\boldsymbol{\Sigma}_t^{1/2}\boldsymbol{\Sigma}_s^{-1/2}\right) \tag{5}$$

$$+ (\boldsymbol{\mu}_t - \boldsymbol{\mu}_s)^T\boldsymbol{\Sigma}_s^{-1}(\boldsymbol{\mu}_t - \boldsymbol{\mu}_s). \tag{6}$$

In the uni-variate case, the normalized Wasserstein distance $\widetilde{W}_2^2$ is equal to the Wasserstein distance W_2^2 between source and target statistics divided by σ_s^2:

$$\widetilde{W}_2^2 = W_2^2\left(\frac{\mu_s}{\sigma_s}, 1, \frac{\mu_t}{\sigma_s}, \frac{\sigma_t^2}{\sigma_s^2}\right) = 1 + \frac{\sigma_t^2}{\sigma_s^2} - 2\frac{\sigma_t}{\sigma_s} + \frac{(\mu_t - \mu_s)^2}{\sigma_s^2} \tag{7}$$

$$= \frac{1}{\sigma_s^2}W_2^2(\mu_s, \sigma_s^2, \mu_t, \sigma_t^2). \tag{8}$$

A.1.3. THE JEFFREY DIVERGENCE

The Jeffrey divergence $J(p_s, p_t)$ between source distribution p_s and target distribution p_t is the symmetrized version of the Kullback-Leibler divergence D_{KL}:

$$J(p_s, p_t) = \frac{1}{2}\left(D_{KL}(p_s\|p_t) + D_{KL}(p_t\|p_s)\right) \tag{9}$$

The Kullback-Leibler divergence between the D-dimensional multivariate normal source and target distributions is defined as

$$D_{KL}(\mathcal{N}_t\|\mathcal{N}_s) = \frac{1}{2}\Bigg(\operatorname{Tr}\left(\boldsymbol{\Sigma}_s^{-1}\boldsymbol{\Sigma}_t\right) \tag{10}$$

$$+ (\boldsymbol{\mu}_s - \boldsymbol{\mu}_t)^\top\boldsymbol{\Sigma}_s^{-1}(\boldsymbol{\mu}_s - \boldsymbol{\mu}_t) \tag{11}$$

$$- D + \ln\left(\frac{\det\boldsymbol{\Sigma}_s}{\det\boldsymbol{\Sigma}_t}\right)\Bigg). \tag{12}$$

The Jeffrey divergence between the D-dimensional multivariate normal source and target distributions then follows as

$$J(\mathcal{N}_t, \mathcal{N}_s) = \frac{1}{4}\Bigg(\operatorname{Tr}\left(\boldsymbol{\Sigma}_s^{-1}\boldsymbol{\Sigma}_t\right) + \operatorname{Tr}\left(\boldsymbol{\Sigma}_t^{-1}\boldsymbol{\Sigma}_s\right) \tag{13}$$

$$+ (\boldsymbol{\mu}_s - \boldsymbol{\mu}_t)^\top\left(\boldsymbol{\Sigma}_s^{-1} + \boldsymbol{\Sigma}_t^{-1}\right)(\boldsymbol{\mu}_s - \boldsymbol{\mu}_t) - 2D\Bigg). \tag{14}$$

A.2. Practical Considerations for Implementing the method

Our method is conceptually very easy to implement. We generally recommend to first explore the easier variant of the algorithm where $N = 0$, i.e., no source statistics are used. In this case, implementing the method boils down to enabling the training mode for all BN layers across the network. We will discuss this option along with two variants important for application to practical problems: Using exponential moving averaging (EMA) to collect target statistics across multiple batches, and using the source statistics as a prior.

Example implementation in PyTorch and caveats We encourage authors of robust models to always evaluate their models, and in particular baseline algorithms on both the train and test set statistics. Implementation in both PyTorch, Tensorflow and other machine learning libraries is straightforward and adds only minimal overhead. For PyTorch, adaptation is possible by simply adding

```
def use_test_statistics(module):
  if isisinstance(module, nn._BatchNorm):
    module.train()
model.eval()
model.apply(use_test_statistics)
```

before starting a model evaluation.

For the adaptation to a full dataset, we provide a reference implementation with the source code release of this paper. Also, in contrast to the convention of not shuffling examples during test time, make sure to enable dataset shuffling also during test time in order to compute the correct statistics marginalized over class assignment.

Exponential moving averaging In practice, it might be beneficial to keep track of samples already encountered and use a running mean and variance on the test set to normalize new samples. We can confirm that this technique closely matches the full-dataset adaptation case even when evaluating with batch size 1 and is well suited for settings with less powerful hardware, or in general settings where access to the full batch of samples is not possible. Variants of this technique include the adaptation of the decay factor to discard statistics of samples encountered in the past (e.g. when the data domain slowly drifts over time).

A.3. Measuring robustness

The ImageNet-C benchmark (Hendrycks & Dietterich, 2019) consists of 15 test corruptions (and four hold-out corruptions) which are applied with five different severity levels to the 50 000 test images of the ILSVRC2012 subset of ImageNet. During evaluation,

model responses are assumed to be conditioned only on single samples, and are not allowed to adapt to e.g. a batch of samples from the same corruption. We call this the ad-hoc or non-adaptive scenario. The main performance metric of IN-C is the mean corruption error (mCE), which is a normalization of the top-1 errors of the model with the top-1 errors of AlexNet across the $C = 15$ test corruptions and $S = 5$ severities:

$$\text{mCE(model)} = \frac{1}{C}\sum_{c=1}^{C}\frac{\sum_{s=1}^{S}\text{err}_{c,s}^{\text{model}}}{\sum_{s=1}^{S}\text{err}_{c,s}^{\text{AlexNet}}}. \tag{15}$$

Note that mCE reflects only one possible averaging scheme over the IN-C corruption types. We will additionally report the overall top-1 accuracies and report results for all individual corruptions in the supplementary material and the project repository.

In many application scenarios, this ad-hoc evaluation is too restrictive. Instead, often many unlabeled samples with similar corruptions are available, which can allow models to adapt to the shifted data distribution. To reflect such scenarios, we propose to also benchmark the robustness of adapted models. To this end, we split the 50 000 validation samples with the same corruption c and severity s into batches $\mathbf{X}^{c,s}$ with n samples each and allow the prediction model $f(.)$ to condition its responses on the complete batch of images, i.e.

$$\mathbf{y}^{c,s} = f(\mathbf{X}^{c,s}) \tag{16}$$

where $\mathbf{y}_n^{c,s}$ has n elements corresponding to the predicted labels. We then compute mCE and top-1 accuracy in the usual way. For $n = 1$, this evaluation scheme reduces to the ad-hoc scenario. We distinguish two adaptation scenarios. In the *full* adaptation scenario, we set $n = 50\,000$, meaning the model may adapt to the full set of unlabeled samples with the same corruption type before evaluation. In the *partial* adaptation scenario, we set $n = 8$ to test how efficiently models can adapt to a relatively small number of unlabeled samples.

A.4. Notes on (internal) covariate shift

Covariate shift can be formalized as follows:

Definition 1 (Covariate Shift, cf. Sugiyama & Kawanabe, 2012; Schölkopf et al., 2012)**.** *There exists covariate shift between a source distribution with density $p_s : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}^+$ and a target distribution with density $p_t : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}^+$, written as $p_s(\mathbf{x}, y) = p_s(\mathbf{x})p_s(y|\mathbf{x})$ and $p_t(\mathbf{x}, y) = p_t(\mathbf{x})p_t(y|\mathbf{x})$, if $p_s(y|\mathbf{x}) = p_t(y|\mathbf{x})$ and $p_s(\mathbf{x}) \neq p_t(\mathbf{x})$ where $y \in \mathcal{Y}$ denotes the class label.*

Note that our notion of (internal) covariate shift is different from the notion used by Ioffe & Szegedy (2015) and Santurkar et al. (2018): In *i.i.d.* training settings, Ioffe & Szegedy (2015) hypothesized that covariate shift introduced by changing lower layers in the network is reduced by BN, explaining the empirical success of the method. We do not provide evidence for this line of research in this work: Instead, we focus on the covariate shift introduced (by design) in datasets such as IN-C, and provide evidence for the hypothesis that high-level domain shifts in the input partly manifests in shifts and scaling of *internal* activations.

B. Experimental Setup

B.1. Notes on models

Note that we only re-evaluate existing model checkpoints, and hence do not perform any hyperparameter tuning or adaptations to model training. Depending on the batch size and the architecture, model evaluations are done on one to eight Nvidia RTX 2080 GPUs (i.e., using 12 to 96 GB of memory) or up to four Nvidia V100 GPUs (128 GB of memory). We evaluate pre-trained variants of DenseNet (Huang et al., 2017), GoogLeNet (Szegedy et al., 2015), Inception and GoogLeNet (Szegedy et al., 2016), MNASnet (Tan et al., 2019), Mobilenet (Sandler et al., 2018), ResNet (He et al., 2016), ResNeXt (Xie et al., 2017), ShuffleNet (Ma et al., 2018), VGG (Simonyan & Zisserman, 2015) and Wide Residual Network (WRN, Zagoruyko & Komodakis, 2016) from the `torchvision` library (Marcel & Rodriguez, 2010). All models are trained on the ILSVRC2012 subset of IN comprised of 1.2 million images in the training and a total of 1000 classes (Russakovsky et al., 2015; Deng et al., 2009). We also consider a ResNeXt-101 variant pre-trained on a 3.5 billion image dataset and then fine-tuned on the IN training set (Mahajan et al., 2018). We additionally evaluate the four leading methods from the ImageNet-C leaderboard, namely Stylized ImageNet training (SIN; Geirhos et al., 2019), adversarial noise training [ANT; Rusak et al., 2020] as well as a combination of ANT and SIN (Rusak et al., 2020), optimized data augmentation using AutoAugment (AugMix; Hendrycks et al., 2020; Cubuk et al., 2019) and Assemble Net (Lee et al., 2020). For partial adaptation, we choose $N \in \{2^0, \cdots, 2^{10}\}$ and select the optimal value on the holdout corruption mCE.

B.2. Notes on datasets

In the main paper, we have used several datasets and provide more relevant information here:

ImageNet-C (IN-C) IN-C is comprised of corrupted versions of the 50 000 images in the IN validation set. The dataset offers five severities per corruption type, for a total of 15 "test" and 4 "holdout" corruptions. For the evaluation on IN-C, 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.

ImageNet-A (IN-A) IN-A consists of unmodified real-world images which yield chance level classification performance in IN trained ResNet-50 models.

ObjectNet (ON) ON is a test set containing 50 000 images like IN organized in 313 object classes with 109 unambiguously overlapping IN classes. We find that there are 9 classes with multiple possible mappings from ON to IN (see the list in Table 4); we discard these classes in our evaluation. Models trained on IN experience a large performance drop on the order of 55% points when tested on ON. ON is an interesting test case for unsupervised domain adaptation since IN and ON are likely sampled from different distributions.

ImageNet-V2 (IN-V2) IN-V2 aims to mimic the test distribution of IN, with slight differences in image selection strategies. There are three test sets in IN-V2 that differ in *selection frequencies* of the MTurk workers and hence vary slightly in their label

distribution. The selection frequency is given by the fraction of MTurk workers who selected an image for its target class. For the "MatchedFrequency" dataset, images were sampled according to the estimated selection frequency of sampling of the original IN validation dataset. For the "Threshold0.7" variant of IN-V2, images were sampled with a selection frequency of at least 0.7. The "TopImages" was sampled from images with the highest selection frequency. Although all three test sets were sampled from the same Flickr candidate pool and were labeled correctly and selected by more than 70% of MTurk workers, the model accuracies on these datasets vary by 14%. The authors observe a systematic accuracy drop when comparing model performance on the original IN validation set and IN-V2 and attribute it to the distribution gap between their datasets and the original IN-V2. They quantify the distribution gap by how much the change from the original distribution to the new distribution affects the considered model. The possibility to bridge this distribution gap makes all three datasets of IN-V2 interesting candidates to test the practical robustness of our method to violations of the assumption that no label shift is present on the test dataset. The test set intentionally shows objects from new viewpoints on new backgrounds.

B.3. Preprocessing

For IN, we resize all images to 256×256px and take the center 224×224px crop. For IN-C, images are already cropped. We also center and re-scale the color values with $\mu_{RGB} = [0.485, 0.456, 0.406]$ and $\sigma = [0.229, 0.224, 0.225]$.

B.4. Overview of models in torchvision

In Table 5, we provide a list of the models we evaluate in the main paper, along with numbers of trainable parameters and batchnorm parameters. Note that the fraction of BN parameters is at most at 1% compared to all trainable parameters in all considered models.

B.5. Baseline corruption errors

In Table 6, we report the scores used for converting top-1 error into the mean corruption error (mCE) metric proposed by Hendrycks & Dietterich (2019).

B.6. Hyperparameter Tuning

Our method is generally parameter-free if only target statistics should be considered for normalization. This approach is generally preferred for larger batch sizes n and should also be adapted in practise when a sufficient amount of samples is available. For tuning N, we consider the pre-defined holdout corruptions in IN-C, including speckle noise, saturation, Gaussian blur and spatter using a line search across different values for N.

B.7. Software Stack

We use various open source software packages for our experiments, most notably Docker (Merkel, 2014), scipy and numpy (Virtanen et al., 2020), GNU parallel (Tange, 2011), Tensorflow (Abadi et al., 2016), PyTorch (Paszke et al., 2017) and torchvision (Marcel & Rodriguez, 2010).

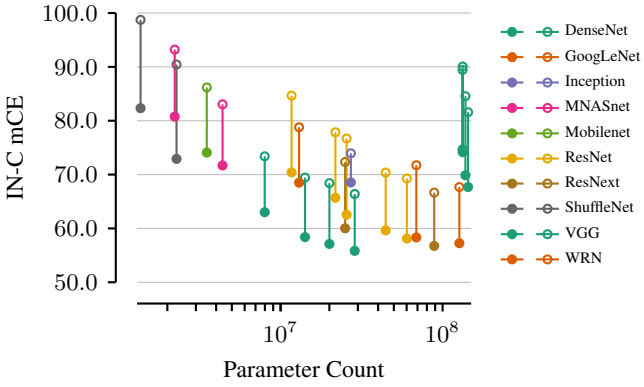

Figure 3: Adaptation (●) improves baseline (○) mCE across all 25 model architectures in the `torchvision` library, often on the order of 10 points. Best viewed in color.

C. Related Work

Robustness towards common corruptions The IN-C benchmark (Hendrycks & Dietterich, 2019) has been extended to MNIST (Mu & Gilmer, 2019), several object detection datasets (Michaelis et al., 2019) and image segmentation (Kamann & Rother, 2019) reflecting the interest of the robustness community. Most proposals for improving robustness involve special training protocols, requiring time and additional resources. This includes data augmentations like Gaussian noise (Ford et al., 2019), optimized mixtures of data augmentations in conjunction with a consistency loss (Hendrycks et al., 2020), training on stylized images (Geirhos et al., 2019; Michaelis et al., 2019; Mikołajczyk & Grochowski, 2018) or against adversarial noise distributions (Rusak et al., 2020). Other approaches tweak the architecture, e.g. through adding shift-equivariance with an anti-aliasing module, (Zhang, 2019) or assemble different training techniques (Lee et al., 2020).

Unsupervised Domain adaptation Unsupervised domain adaptation (DA) is a form of transductive inference where additional information about the test dataset is used to adapt a model to the test distribution. In this context, adapting BN parameters has been considered in many previous studies. The idea of adapting activation statistics was originally proposed by (Sun et al., 2017). Li et al. (2016); Cariucci et al. (2017) evaluate the performance of adapting BN parameters in unsupervised domain adaptation settings. Schneider et al. (2018) show that the topline performance obtained by adapting BN parameters with supervised learning on the target domain directly is actually sufficient for very good performance on the widely considered small digit datasets. This corroborates results from Rebuffi et al. (2017) who discuss batch adaptation for multi-task learning on larger scale datasets. As an application example, Bug et al. (2017) show that adaptive normalization is useful for removing domain shifts on histopathological data. Sun et al. (2019) propose a method for self-supervised domain adaptation on single examples. French et al. (2017) successfully use self-ensembling for domain adaptation.

D. Additional results

Results on other datasets: IN-A, IN-V2, ObjectNet. In all ablation studies in this subsection, we have used N=0 and varied

ON class	IN classes
wheel	wheel; paddlewheel, paddle wheel
helmet	football helmet; crash helmet
chair	barber chair; folding chair; rocking chair, rocker
still_camera	Polaroid camera, Polaroid Land camera; reflex camera
alarm_clock	analog clock; digital clock
tie	bow tie, bow-tie, bowtie; Windsor tie
pen	ballpoint, ballpoint pen, ballpen, Biro; quill, quill pen; fountain pen
bicycle	mountain bike, all-terrain bike, off-roader; bicycle-built-for-two, tandem bicycle, tandem
skirt	hoopskirt, crinoline; miniskirt, mini; overskirt

Table 4: Mapping between 9 ambiguous ON classes and the possible correspondences in IN. Different IN classes are separated with a semicolon.

Model	Parameter Count	BN Parameters	Fraction (%)
densenet121	7.98×10^6	8.36×10^4	0.010
densenet161	2.87×10^7	2.20×10^5	0.008
densenet169	1.41×10^7	1.58×10^5	0.011
densenet201	2.00×10^7	2.29×10^5	0.011
googlenet	1.30×10^7	1.51×10^4	0.001
inception-v3	2.72×10^7	3.62×10^4	0.001
mnasnet0-5	2.22×10^6	2.06×10^4	0.009
mnasnet0-75	3.17×10^6	2.98×10^4	0.009
mnasnet1-0	4.38×10^6	3.79×10^4	0.009
mnasnet1-3	6.28×10^6	4.88×10^4	0.008
mobilenet-v2	3.50×10^6	3.41×10^4	0.010
resnet101	4.45×10^7	1.05×10^5	0.002
resnet152	6.02×10^7	1.51×10^5	0.003
resnet18	1.17×10^7	9.60×10^3	0.001
resnet34	2.18×10^7	1.70×10^4	0.001
resnet50	2.56×10^7	5.31×10^4	0.002
resnext101-32x8d	8.88×10^7	2.03×10^5	0.002
shufflenet-v2-x0-5	1.37×10^6	7.95×10^3	0.006
shufflenet-v2-x1-0	2.28×10^6	1.62×10^4	0.007
shufflenet-v2-x1-5	3.50×10^6	2.34×10^4	0.007
shufflenet-v2-x2-0	7.39×10^6	3.37×10^4	0.005
vgg11-bn	1.33×10^8	5.50×10^3	4.142×10^{-5}
vgg13-bn	1.33×10^8	5.89×10^3	4.425×10^{-5}
vgg16-bn	1.38×10^8	8.45×10^3	6.106×10^{-5}
vgg19-bn	1.44×10^8	1.10×10^4	7.662×10^{-5}
wide-resnet101-2	1.27×10^8	1.38×10^5	0.001
wide-resnet50-2	6.89×10^7	6.82×10^4	0.001

Table 5: Overview of different models with parameter counts. We show the total number of BN parameters, which is a sum of affine parameters.

n. The technique does not work for the case of "natural adversarial examples" of IN-A (Hendrycks et al., 2019) and the error rate stays above 99%, suggesting that the covariate shift introduced in IN-A by design is more severe compared to the covariate shift of IN-C and can not be corrected by merely calculating the correct BN statistics.

We evaluate how the batch size for estimating the statistics at test time affects the performance on IN, IN-V2 and ON in Fig. 4. As expected, for IN the adaptation to test time statistics converges to the performance of the train time statistics in the limit of large batch sizes, see Fig. 4 left. For IN-V2, we find similar results, see Fig. 4 middle. This observation shows that (*i*) there is no systematic covariate shift between the IN train set and the IN-V2 validation set that could be corrected by using the correct statistics and (*ii*) is further evidence for the *i.i.d.* setting pursued by the authors of IN-V2. In case of ON (Fig. 4 right), we see slight improvements when using a batch size bigger than 128.

Severity of covariate shift correlates with performance degradation. The relationship between the performance degradation on IN-C and the covariate shift suggests an unsupervised way of estimating the classification performance of a model on a new corruption. Taking the normalized Wasserstein distance

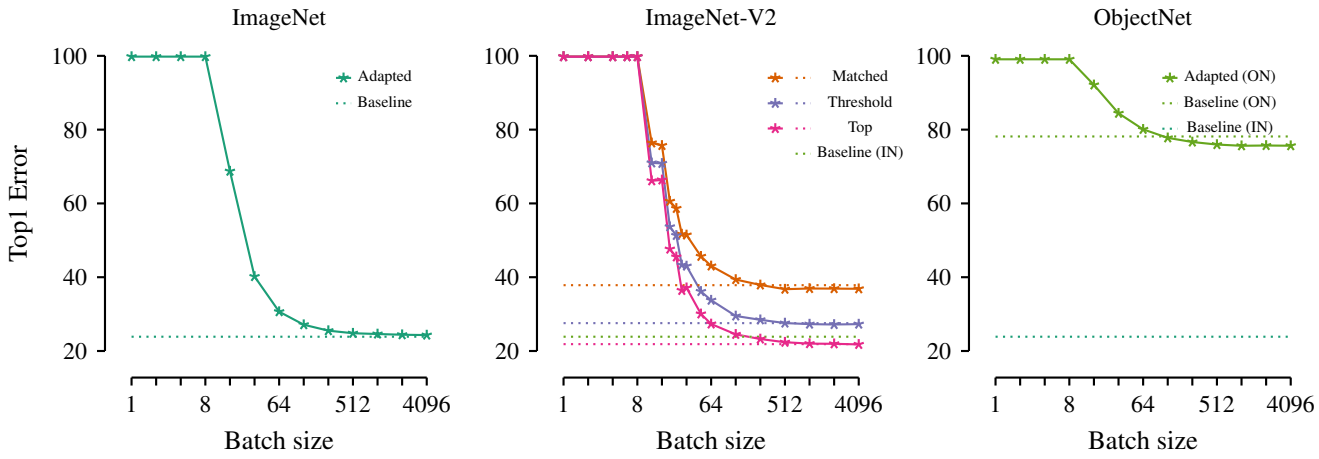

Figure 4: Batch size vs. performance trade-off for different natural image datasets with no covariate shift (IN, IN-V2) and complex covariate shift (ObjectNet).

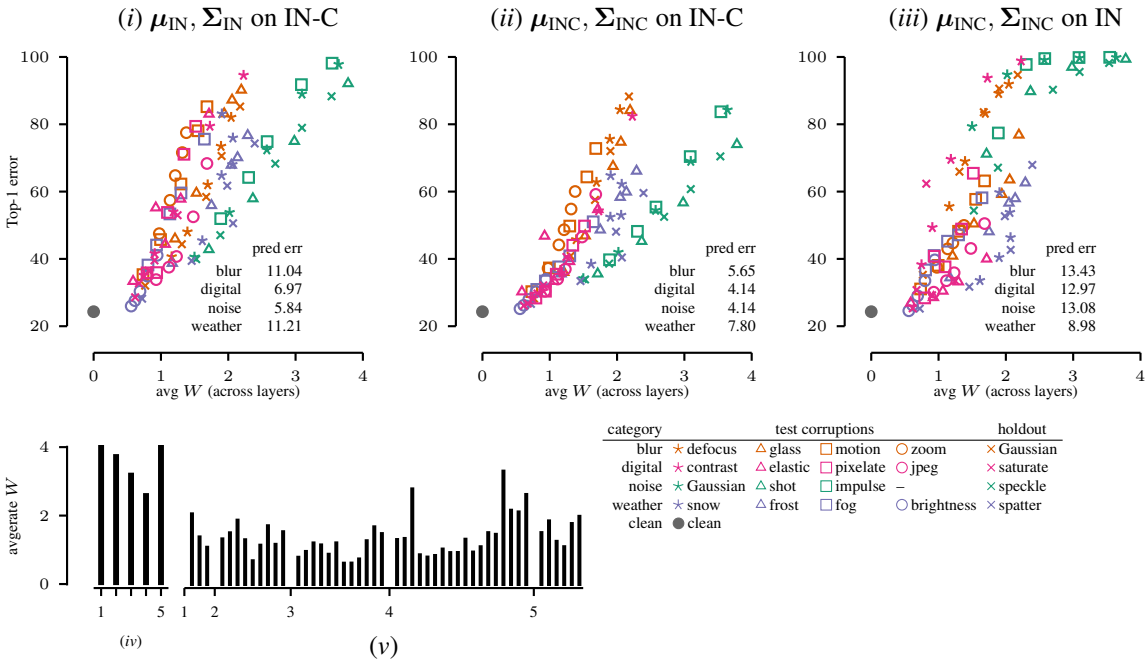

Figure 5: We employ the Wasserstein metric between optimal source (IN) and target (IN-C) statistics to quantify domain shift. The metric correlates well with top-1 errors (*i*) of non-adapted models on IN-C, (*ii*) adapted models on IN-C, indicating that even after reducing covariate shift, the metric is predictive of the remaining source–target mismatch (*iii*) IN-C adapted models on IN, the reverse case of (*i*). Holdout corruptions can be used to get a linear estimate on the prediction error of test corruptions (tables). We depict input and downsample (*iv*) as well as bottlneck layers (*v*) and notice the largest shift in early and late downsampling layers. The metric is either averaged across layers (*i–iii*) or across corruptions (*iv–v*).

Category	Corruption	top1 error
Noise	Gaussian Noise	0.886428
	Shot Noise	0.894468
	Impulse Noise	0.922640
Blur	Defocus Blur	0.819880
	Glass Blur	0.826268
	Motion Blur	0.785948
	Zoom Blur	0.798360
Weather	Snow	0.866816
	Frost	0.826572
	Fog	0.819324
	Brightness	0.564592
	Contrast	0.853204
Digital	Elastic Transform	0.646056
	Pixelate	0.717840
	JPEG Compression	0.606500
Hold-out Noise	Speckle Noise	0.845388
Hold-out Digital	Saturate	0.658248
Hold-out Blur	Gaussian Blur	0.787108
Hold-out Weather	Spatter	0.717512

Table 6: AlexNet top1 errors on IN-C

(cf. §F.1) between the statistics of the source and target domains[2] computed on all samples with the same corruption and severity and averaged across all network layers, we find a correlation with the top-1 error (Fig. 5 *i–iii*) of both adapted and fully adapted model on IN-C corruptions. Within single corruption categories (noise, blur, weather, and digital), the relationship between top-1 error and Wasserstein distance is particularly striking: using linear regression, the top-1 accuracy of hold-out corruptions can be estimated with around 1–2% absolute mean deviation (cf. §E.5) within a corruption, and with around 5–15% absolute mean deviation when the estimate is computed on the holdout corruption of each category (see Fig. 5, typically, a systematic offset remains). In Fig. 5*(iv–v)*, we display the Wasserstein distance across individual layers and observe that the covariate shift is particularly present in early and late downsampling layers of the ResNet-50.

E. Additional Results

E.1. Summary statistics and quantification of covariate shift between different IN-C conditions

Given the 95 distances/divergences between the baseline (IN) statistics and 95 IN-C conditions, we first perform a layer-wise analysis of the statistics and depict the results in Figure 6. The unnormalized Wasserstein distance is sensitive to the magnitude of the source statistics and hence differs qualitatively from the results on the normalized Wasserstein distance and Jeffrey Divergence. We appreciate that the most notable difference between source and target domains is visible in the ResNet-50 downsampling layers. All three metrics suggest that the shift is mainly present in the first and final layers of the network, supporting the hypothesis that within the common corruption dataset, we have both superficial covariate shift which can be corrected by simple means (such as brightness or contrast variations) in the first layers, and also more "high-level" domain shifts which can only be corrected in the later layers of the network.

[2]For computing the Wasserstein metric we make the simplifying assumption that the empirical mean and variance fully parametrize the respective distributions.

In Figure 7, we more closely analyze this relationship for different common corruptions. We can generally appreciate the increased measures as the corruption severity increases.

E.2. Relationship between parameter count and IN-C improvements

In addition to Fig. 3 in the main paper, we show the relationship between parameter count and IN-C mCE. In general, we see that the parameter counts correlates with corruption robustness since larger models have smaller mCE values.

E.3. Per-corruption results on IN-C

We provide more detailed results on the individual corruptions of IN-C for the most important models considered in our study in Fig. 8 and Fig. 9. The results are shown for models where the BN parameters are adapted on the full test sets. The adaptation consistently improves the error rates on all corruptions for both vanilla and AugMix.

E.4. Qualitative Analysis of Similarities between Common Corruptions

In this analysis, we compute a t-SNE embedding of the Wasserstein distances between the adapted models and the non-adapted model from Fig. 5. The results are displayed in Fig. 10. We observe that the different corruption categories indicated by the different colors are grouped together except for the 'digital' category (pink). This visualization shows that corruption categories mostly induce similar shifts in the BN parameters. This might be an explanation why training a model on Gaussian noise generalizes so well to other noise types as has been observed by Rusak et al. (2020): By training on Gaussian noise, the BN statistics are adapted to the Gaussian noise corruption and from Fig. 10, we observe that these statistics are similar to the BN statistics of other noises.

E.5. Error prediction based on the Wasserstein distance

In Fig. 5, we observe that the relationship between the Wasserstein distance and the top-1 error on IN-C is strikingly linear in the considered range of the Wasserstein distance. Similar corruptions and corruption types (indicated by color) exhibit similar slope, allowing to approximate the expected top-1 error rate without any information about the test domain itself. Using the split of the 19 corruptions into 15 test and 4 holdout corruptions (Hendrycks & Dietterich, 2019), we compute a linear regression model on the five data points we get for each of the holdout corruptions (corresponding to the five severity levels), and use this model to predict the expected top-1 error rates for the remaining corruptions within the corruption family. This scheme works particularly for the "well defined" corruption types such as noise and digital (4.1% points absolute mean deviation from the real error. The full results are depicted in Table 7.

E.6. Training details on the models trained with Fixup initialization and GroupNorm

In Section 5 of the main paper, we consider IN models trained with GroupNorm and Fixup initialization. For these models, we consider the original reference implementations provided by the authors. We train ResNet-50, ResNet-101 and ResNet-152 models

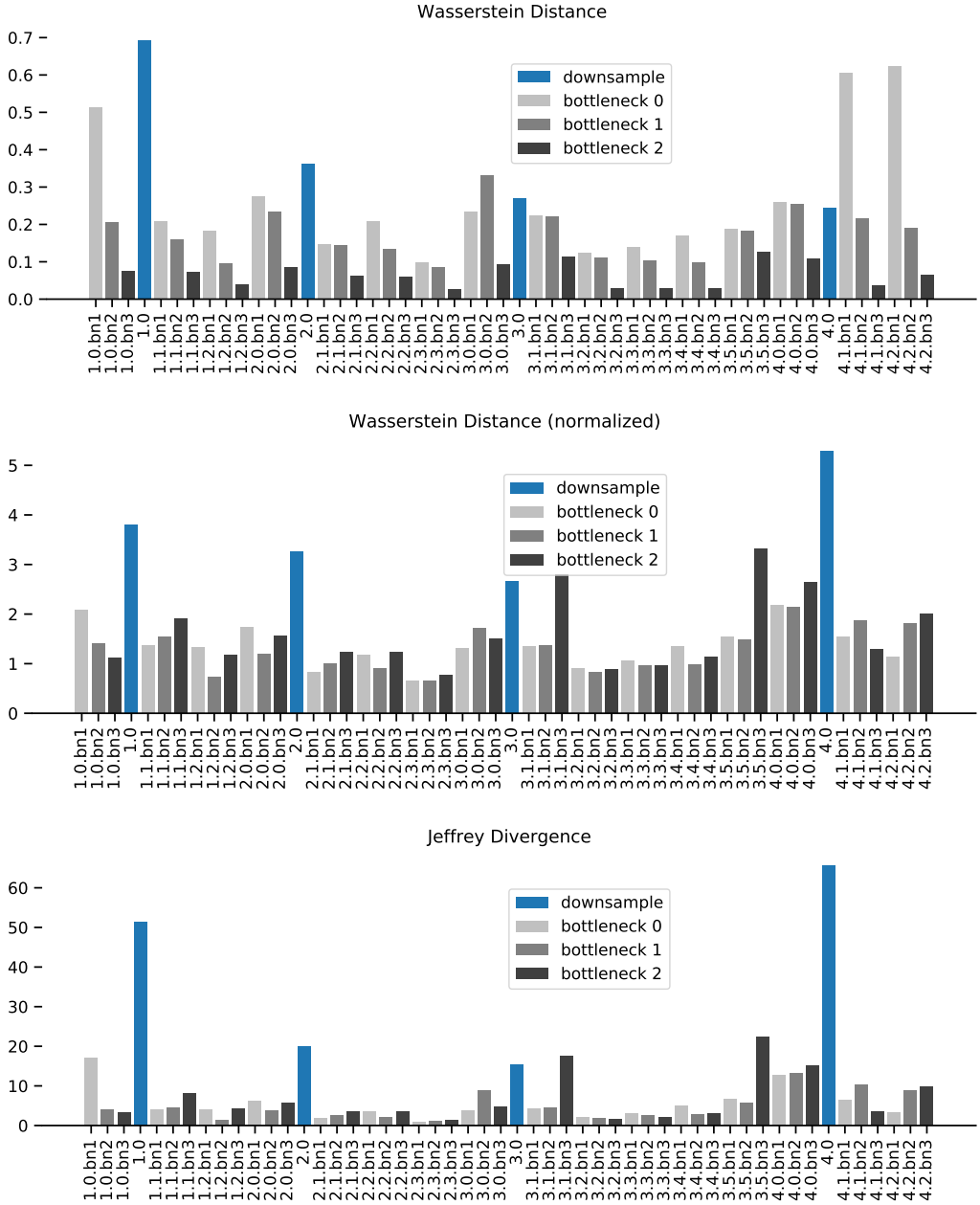

Figure 6: Wasserstein distance, normalized Wasserstein distance and Jeffrey divergence estimated among source and target statistics between different network layers. We report the respective metric w.r.t. to the difference between baseline (IN) and target (IN-C) statistics and show the value averaged across all corruptions. We note that for a ResNet-50 model, downsampling layers contribute most to the overall error.

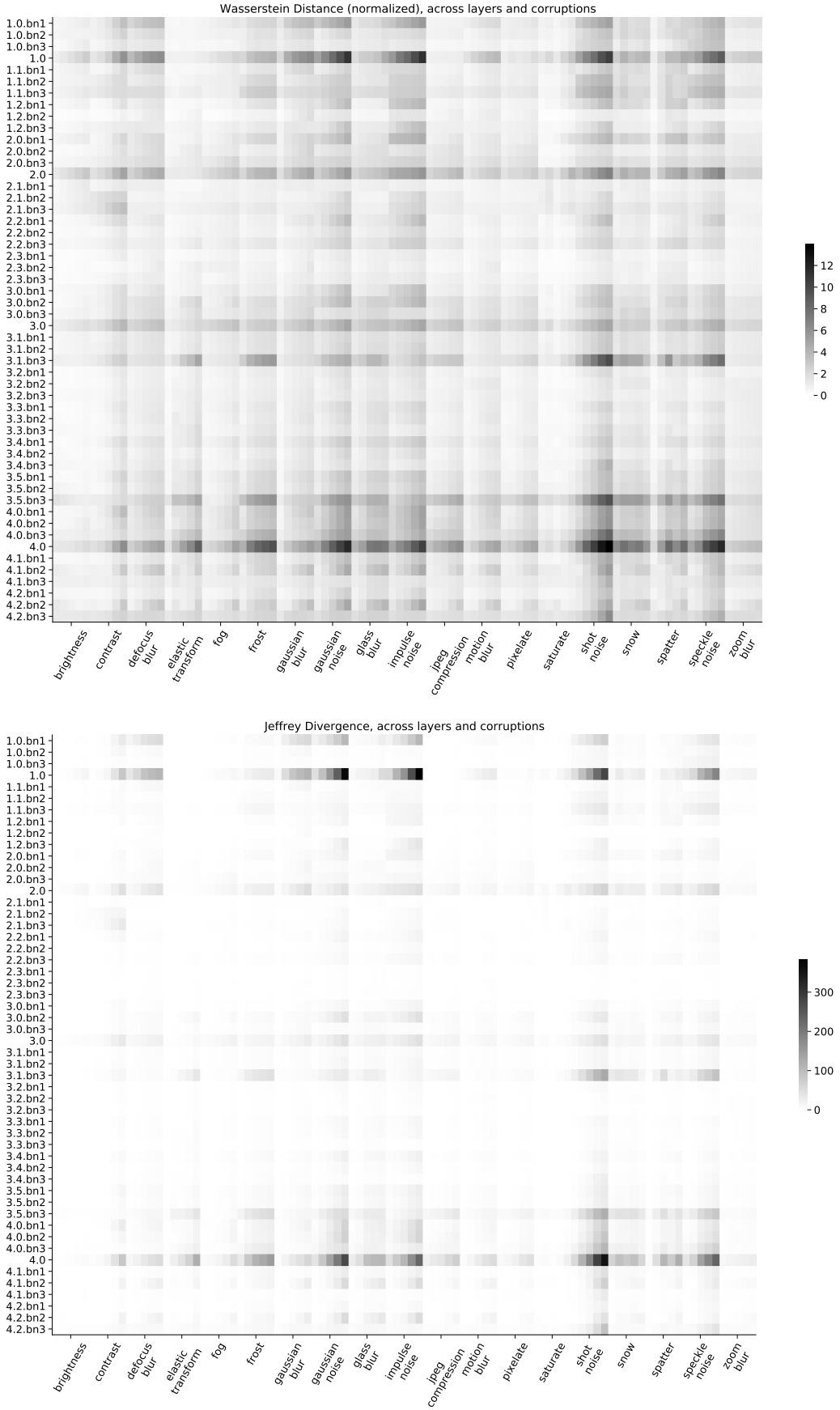

Figure 7: Normalized Wasserstein Distance and Jeffrey Divergence across corruptions and layers in a ResNet-50.

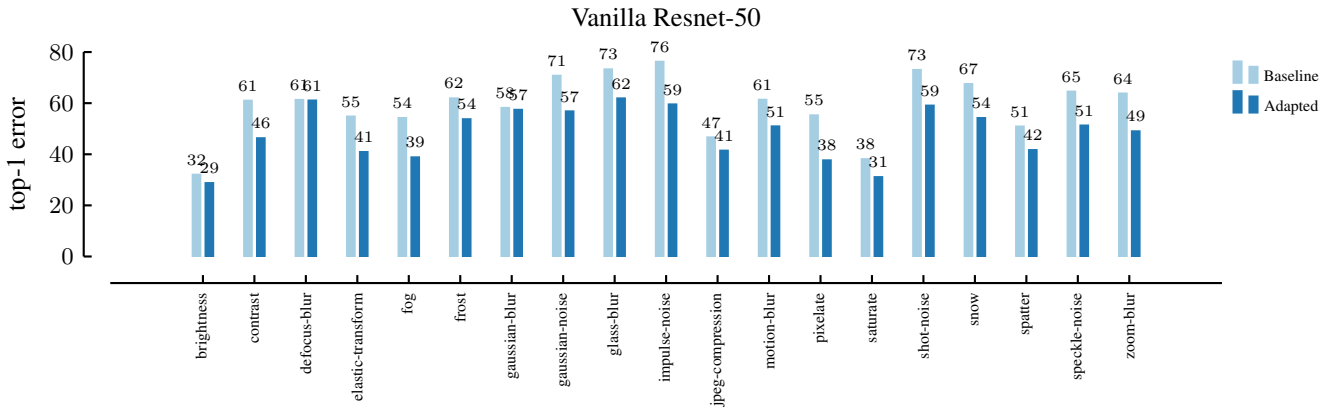

Figure 8: Results on the individual corruptions of IN-C for the vanilla trained ResNet-50 with and without adaptation. Adaptation reduces the error on all corruptions.

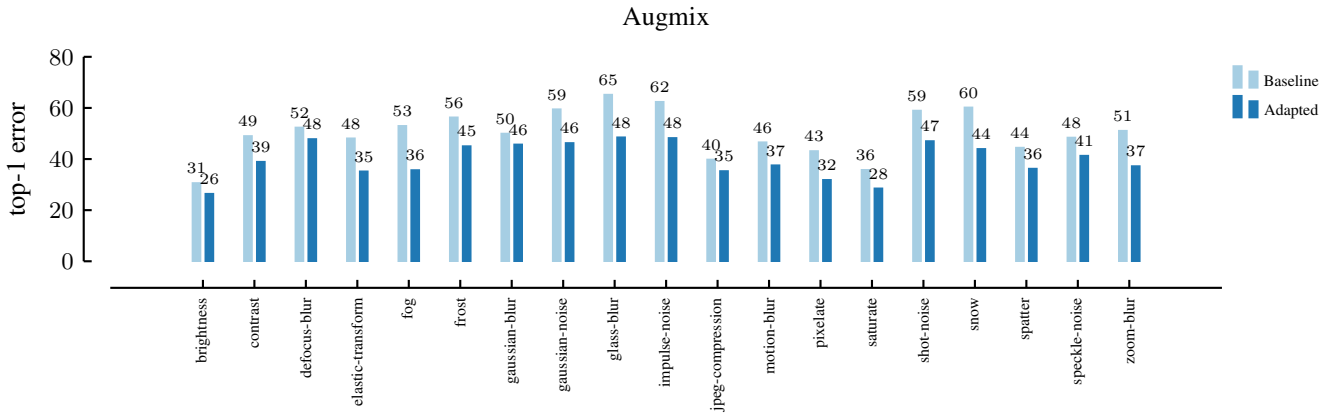

Figure 9: Results on the individual corruptions of IN-C for the AugMix model with and without adaptation. Adaptation reduces the error on all corruptions.

	test error			holdout (train) error			model	
	true	pred	$\lvert\Delta\rvert$	true	pred	$\lvert\Delta\rvert$	coef	intercept
Fig. 5 (i)								
blur	64.89	54.53	11.04	58.13	58.13	3.24	37.59	-0.70
digital	54.37	51.96	6.97	38.08	38.08	0.60	37.20	6.39
noise	73.29	69.68	5.84	64.51	64.51	0.65	24.66	1.68
weather	53.87	42.92	11.21	50.84	50.84	5.48	25.80	6.33
Fig. 5 (ii)								
blur	55.68	53.28	5.65	57.38	57.38	4.01	42.74	-9.51
digital	41.53	39.80	4.14	31.05	31.05	0.34	23.44	11.09
noise	58.43	55.04	4.14	51.24	51.24	1.01	18.13	5.06
weather	43.84	36.16	7.80	41.63	41.63	4.32	17.80	10.91
Fig. 5 (iii)								
blur	57.10	69.84	13.43	74.01	74.01	3.96	43.50	5.93
digital	46.16	38.06	12.97	36.22	36.22	10.52	4.94	32.01
noise	93.60	85.84	13.08	81.10	81.10	3.52	22.56	23.65
weather	43.74	36.90	8.98	44.05	44.05	6.20	23.29	3.87

Table 7: Estimating top-1 error of unseen corruptions within the different corruption classes. We note that especially for well defined corruptions (like noise or digital corruptions), the estimation scheme works well. We follow the categorization originally proposed by Hendrycks & Dietterich (2019).

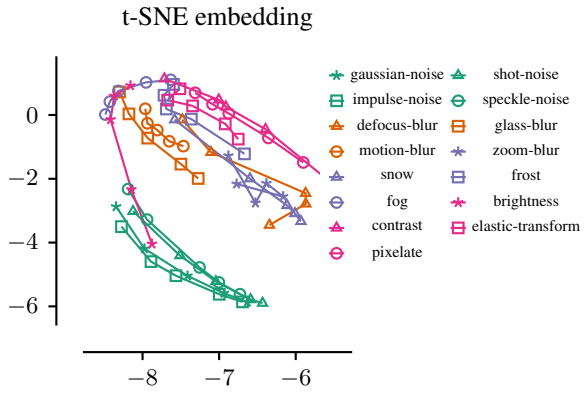

Figure 10: t-SNE embeddings of the Wasserstein distances between BN statistics adapted on the different corruptions. This plot shows evidence on the similarities between different corruption types.

with stochastic gradient descent with momentum (learning rate 0.1, momentum 0.9), with batch size 256 and weight decay 1×10^{-4} for 100 epochs.

E.7. Effect of Pseudo Batchsize

We show the full results for considering different choices of N for ResNet-50, Augmix, ANT, ANT+SIN and SIN models and display the result in Fig. 12. We observe a characteristic shape which we believe can be attributed to the way statistics are estimated. We provide evidence for this view by proposing an analytical model further below.

F. A model for selecting the number of pseudo samples

Choosing the number of pseudo-samples N offers an intuitive trade-off between estimating accurate target statistics (low N) and relying on the source statistics (large N). We propose a simple model to investigate optimal choices for N, disregarding all special structure of DNNs, and focusing on the statistical error introduced by estimating $\hat{\mu}_t$ and $\hat{\sigma}_t^2$ from a limited number of samples n. To this end, we estimate upper (U) and lower (L) bounds of the Wasserstein distance W as a function of N and the covariate shift ($\mu_t - \mu_s$ and σ_t^2/σ_s^2) which provides good empirical fits between the estimated W and empirical performance for ResNet-50 for different N (Fig. 13; bottom row). Choosing N such that L or U are minimised (Fig. 13; example in top row) qualitatively matches the values we find, see Appendix D for all details.

Univariate model. We start with defining a univariate model. We denote the source statistics as μ_s, σ_s^2, the true target statistics as μ_t, σ_t^2 and the estimated target statistics as $\hat{\mu}_t, \hat{\sigma}_t^2$. For normalization, we take a convex combination of the source statistics and estimated target statistics:

$$\bar{\mu} = \frac{N\mu_s + n\hat{\mu}_t}{N+n}, \bar{\sigma}^2 = \frac{N\sigma_s^2 + n\hat{\sigma}_t^2}{N+n}. \tag{17}$$

We now analyze the trade-off between using an estimate closer to the source or closer to the estimated target statistics. In the former case, the model will suffer under the covariate shift present between target and source distribution. In the latter case, small batch sizes n will yield unreliable estimates for the true target statistics, which might hurt the performance even more than the source-target mismatch. Hence, we aim to gain understanding in the trade-off between both options, and potential optimal choices of N for a given sample size n.

As a metric of domain shift with good properties for our following derivation, we leverage the Wasserstein distance. In E.4, we already established an empirical link between domain shift measured in terms of the top-1 performance vs. the Wasserstein distance between model statistics and observed a linear relationship for case of common corruptions.

Proposition 1 (Bounds on the expected value of the Wasserstein distance between target and combined estimated target and source statistics)**.** *We denote the source statistics as μ_s, σ_s^2, the true target statistics as μ_t, σ_t^2 and the biased estimates of the target statistics as $\hat{\mu}_t, \hat{\sigma}_t^2$. For normalization, we take a convex combination of the source statistics and estimated target statistics as discussed in Eq. 17. At a confidence level $1-\alpha$, the expectation value of the squared Wasserstein distance $W_2^2(\mu_t, \sigma_t, \bar{\mu}, \bar{\sigma})$ between ideal and estimated target statistics w.r.t. to the distribution of sample mean $\hat{\mu}_t$ and sample variance $\hat{\sigma}_t^2$ is bounded from above and below with $L \leq \mathbb{E}[W_2^2] \leq U$, where*

$$\begin{aligned} L &= \left(\sigma_t - \sqrt{\frac{N}{N+n}\sigma_s^2 + \frac{n-1}{N+n}\sigma_t^2}\right)^2 \\ &\quad + \frac{N^2}{(N+n)^2}(\mu_t - \mu_s)^2 + \frac{n}{(N+n)^2}\sigma_t^2, \\ U &= L + \sigma_t^5 \frac{(n-1)}{2(N+n)^2} a^{-3/2}, \\ \text{with } a &= \frac{N}{N+n}\sigma_s^2 + \frac{1}{N+n}\chi^2_{1-\alpha/2,n-1}\sigma_t^2. \end{aligned} \tag{18}$$

The quantity $\chi^2_{1-\alpha,n-1}$ denotes the left tail value of a chi square distribution with $n-1$ degrees of freedom, defined as $P\left(X \leq \chi^2_{1-\alpha/2,n-1}\right) = \alpha/2$ for $X \sim \chi^2_{n-1}$.

Proof. see Appendix F.2. □

Proof Sketch We are interested in the expected value of the Wasserstein distance defined in (F.1) between the target statistics μ_t, σ_t^2 and the mixed statistics $\bar{\mu}, \bar{\sigma}^2$ introduced above in equation (17), taken with respect to the distribution of the sample moments $\hat{\mu}_t, \hat{\sigma}_t^2$. The expectation value itself cannot be evaluated in closed form because the Wasserstein distance contains a term proportional to $\bar{\sigma}$ being the square root of the convex combination of target and source variance. In Lemma 3, the square root term is bounded from above and below using Jensen's inequality and Holder's defect formula which is reviewed in Lemma 2. After having bounded the problematic square root term, the proof of Proposition 1 reduces to inserting the expectation values of sample mean and sample variance reviewed in Lemma 1.

F.1. Prerequisites

The Wasserstein distance Given a baseline ResNet-50 model with source statistics μ_s, Σ_s on IN, the Wasserstein distance [cf. Villani, 2008] between the train and test distribution

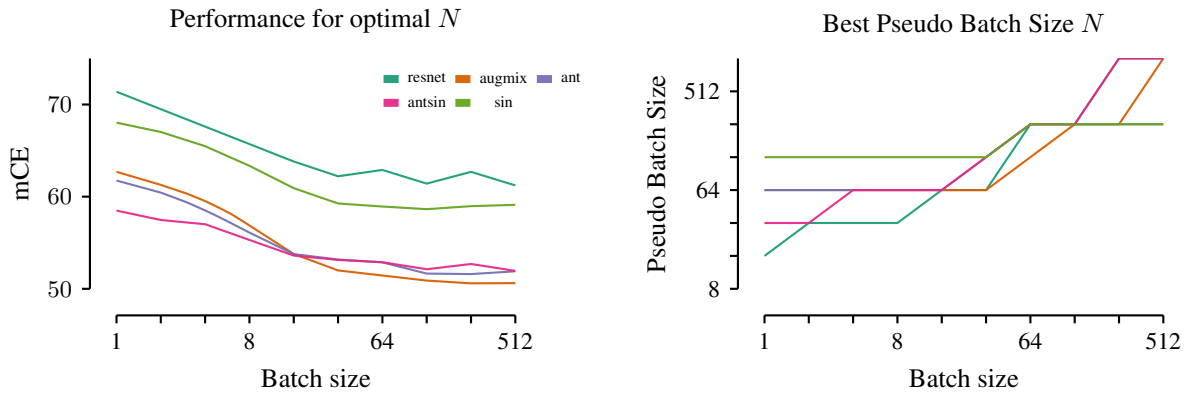

Figure 11: Left: Performance for all the considered ResNet-50 variants based on the sample batch size. The optimal N is chosen according to the mCE on the holdout corruptions. Right: Best choice for N depending on the input batchsize n. Note that in general for high values n, the model is generally more robust to the choice of N.

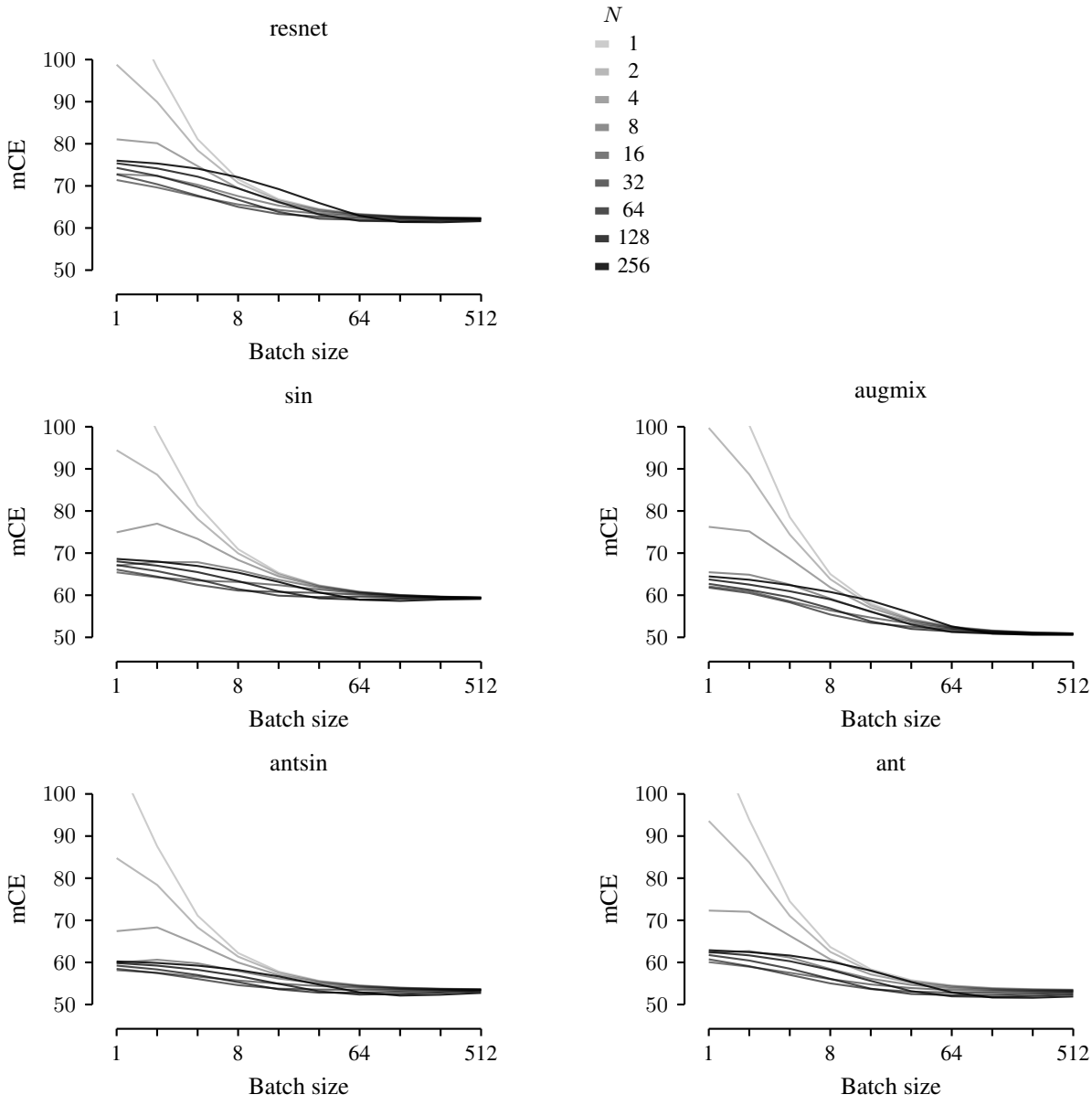

Figure 12: Effects of batch size n and pseudo batch size N for the various considered models. We report mCE averaged across 15 test corruptions.

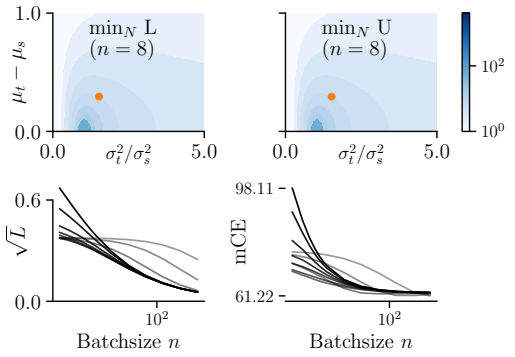

Figure 13: The bound suggests small optimal N for most parameters (i) and qualitatively explains our empirical observation (ii).

with statistics μ_t, Σ_t is given as

$$W_2(p_s, p_t)^2 = \text{tr}\left(\mathbf{\Sigma}_s + \mathbf{\Sigma}_t - 2\left(\mathbf{\Sigma}_t^{1/2}\mathbf{\Sigma}_s\mathbf{\Sigma}_t^{1/2}\right)^{1/2}\right) + \|\boldsymbol{\mu}_s - \boldsymbol{\mu}_t\|_2^2. \tag{19}$$

We define the normalized Wasserstein distance between source and target statistics as

$$\begin{aligned}\widetilde{W}_2^2 =& W_2^2\left(\mathbf{\Sigma}_t\mathbf{\Sigma}_s^{-1}, \mathbf{I}, \mathbf{\Sigma}_s^{-1/2}\boldsymbol{\mu}_t, \mathbf{\Sigma}_s^{-1/2}\boldsymbol{\mu}_s\right)\\ =& \text{tr}\left(\mathbf{I} + \mathbf{\Sigma}_t\mathbf{\Sigma}_s^{-1} - 2\mathbf{\Sigma}_t^{1/2}\mathbf{\Sigma}_s^{-1/2}\right)\\ &+ (\boldsymbol{\mu}_t - \boldsymbol{\mu}_s)^T\mathbf{\Sigma}_s^{-1}(\boldsymbol{\mu}_t - \boldsymbol{\mu}_s).\end{aligned} \tag{20}$$

Lemma 1 (Mean and variance of sample moments, following (Weisstein, 2020))**.** *Given samples x_j from a normal distribution with mean μ_t and variance σ_t^2, the sample moments $\hat{\mu}_t, \hat{\sigma}_t^2$ are random variables depending on the sample size n.*

$$\hat{\mu}_t = \frac{1}{n}\sum_{j=1}^{n} x_j, \quad \hat{\sigma}_t^2 = \frac{1}{n}\sum_{j=1}^{n}(x_j - \hat{\mu}_t)^2 \tag{21}$$

For brevity, we use the shorthand $\mathbb{E}[\cdot]$ for all expectation values with respect to the distribution of $p(\hat{\mu}_t, \hat{\sigma}_t^2|n)$. In particular, our computation uses mean and variance of $\hat{\mu}_t$ and $\hat{\sigma}_t^2$ which are well known for a normal target distribution:

$$\begin{aligned}&\hat{\mu}_t \sim \mathcal{N}\left(\mu_t, \frac{1}{n}\sigma_t^2\right),\ \mathbb{E}[\hat{\mu}_t] = \mu_t,\ \mathbb{V}[\hat{\mu}_t] = \frac{1}{n}\sigma_t^2\\ &\frac{\hat{\sigma}_t^2}{\sigma_t^2/n} \sim \chi_{n-1}^2,\ \mathbb{E}[\hat{\sigma}_t^2] = \frac{n-1}{n}\sigma_t^2,\\ &\mathbb{V}[\hat{\sigma}_t^2] = \frac{\sigma_t^4}{n^2}\mathbb{V}\left[\frac{\hat{\sigma}_t^2}{\sigma_t^2/n}\right] = \frac{\sigma_t^4}{n^2}2(n-1).\end{aligned} \tag{22}$$

The derivation of the variance $\mathbb{V}[\hat{\sigma}_t^2]$ in the last line uses the fact that the variance of a chi square distributed variable with $(n-1)$ degrees of freedom is equal to $2(n-1)$.

Lemma 2 (Holder's defect formula for concave functions in probabilistic notation, following Becker (2012))**.** *If the concave function $f : [a, b] \to \mathbb{R}$ is twice continuously differentiable and there are finite bounds m and M such that*

$$-M \leq f''(x) \leq -m \leq 0\ \forall x \in [a, b], \tag{23}$$

then the defect between Jensen's inequality estimate $f(\mathbb{E}[X])$ for a random variable X taking values $x \in [a, b]$ and the true expectation value $\mathbb{E}[f(X)]$ is bounded from above by a term proportional to the variance of X:

$$f(\mathbb{E}[X]) - \mathbb{E}[f(X)] \leq \frac{1}{2}M\mathbb{V}[X]. \tag{24}$$

Lemma 3 (Upper and lower bounds on the expectation value of $\bar{\sigma}$)**.** *The expectation value of the square root of the random variable $\bar{\sigma}^2$ defined as*

$$\bar{\sigma}^2 = \frac{N}{N+n}\sigma_s^2 + \frac{n}{N+n}\hat{\sigma}_t^2, \tag{25}$$

is bounded from above and below at a confidence level $1-\alpha$ by

$$\begin{aligned}&\sqrt{\mathbb{E}\left[\bar{\sigma}^2\right]} - \frac{1}{2}M\mathbb{V}[\bar{\sigma}^2] \leq \mathbb{E}\left[\sqrt{\bar{\sigma}^2}\right] \leq \sqrt{\mathbb{E}\left[\bar{\sigma}^2\right]}\\ &\sqrt{\mathbb{E}\left[\bar{\sigma}^2\right]} = \sqrt{\frac{N}{N+n}\sigma_s^2 + \frac{n-1}{N+n}\sigma_t^2},\\ &\frac{1}{2}M\mathbb{V}[\bar{\sigma}^2] = \frac{(n-1)}{4(N+n)^2}\sigma_t^4 a^{-3/2}\\ &\text{with } a = \frac{N}{N+n}\sigma_s^2 + \frac{1}{N+n}\chi_{1-\alpha/2,n-1}^2\sigma_t^2.\end{aligned} \tag{26}$$

The quantity $\chi_{1-\alpha/2,n-1}^2$ denotes the left tail value of a chi square distribution with $n-1$ degrees of freedom, defined as $P\left(X \leq \chi_{1-\alpha/2,n-1}^2\right) = \frac{\alpha}{2}$ for $X \sim \chi_{n-1}^2$.

Proof. The square root function is concave, therefore Jensen's inequality implies the upper bound

$$\mathbb{E}\left[\sqrt{\bar{\sigma}^2}\right] \leq \sqrt{\mathbb{E}[\bar{\sigma}^2]}. \tag{27}$$

The square root of the expectation value of $\bar{\sigma}^2$ is computed using the expectation value of the sample variance as given in Lemma 1.

$$\begin{aligned}\sqrt{\mathbb{E}[\bar{\sigma}^2]} &= \sqrt{\frac{N}{N+n}\sigma_s^2 + \frac{n}{N+n}\frac{n-1}{n}\sigma_t^2}\\ &= \sqrt{\frac{N}{N+n}\sigma_s^2 + \frac{n-1}{N+n}\sigma_t^2}.\end{aligned} \tag{28}$$

To state a lower bound, we use Holder's defect formula in probabilistic notation stated in Lemma 2. Holder's formula for concave functions requires that the random variable $\bar{\sigma}^2$ can take values in the compact interval $[a, b]$ and that the second derivative of the square root function $f(\bar{\sigma}^2) = \sqrt{\bar{\sigma}^2}$, exists and is strictly smaller than zero in $[a, b]$. Regarding the interval of $\bar{\sigma}^2$, we provide probabilistic upper and lower bounds. The ratio of sample variance and true variance divided by n follows a chi square distribution with $n-1$ degrees of freedom. At confidence level $1-\alpha$, this ratio is then between $\chi_{1-\alpha/2,n-1}^2$ and $\chi_{\alpha/2,n-1}^2$ defined as follows:

$$\begin{aligned}&\chi_{1-\alpha/2,n-1}^2 \leq \frac{\hat{\sigma}_t^2}{\sigma_t^2/n} \leq \chi_{\alpha/2,n-1}^2,\\ &Pr(X \leq \chi_{1-\alpha/2,n-1}^2) = Pr(X \geq \chi_{\alpha/2,n-1}^2) = \frac{\alpha}{2}.\end{aligned} \tag{29}$$

Then at the same confidence level, the sample variance itself lies between the two quantiles multiplied by σ_t^2/n,

$$\chi_{1-\alpha/2,n-1}^2\frac{\sigma_t^2}{n} \leq \hat{\sigma}_t^2 \leq \chi_{\alpha/2,n-1}^2\frac{\sigma_t^2}{n}, \tag{30}$$

and the random variable $\bar{\sigma}^2$ lies in the interval $[a, b]$ with

$$a = \frac{N}{N+n}\sigma_s^2 + \frac{1}{N+n}\chi^2_{1-\alpha/2,n-1}\sigma_t^2, \tag{31}$$

$$b = \frac{N}{N+n}\sigma_s^2 + \frac{1}{N+n}\chi^2_{\alpha/2,n-1}\sigma_t^2. \tag{32}$$

The variances and chi square values are all positive and therefore both a and b are positive as well, implying that the second derivative of the square root is strictly negative in the interval $[a, b]$.

$$f(\bar{\sigma}^2) = \sqrt{\bar{\sigma}^2},\ f'(\bar{\sigma}^2) = \frac{1}{2}(\bar{\sigma}^2)^{-1/2}, \tag{33}$$

$$f''(\bar{\sigma}^2) = -\frac{1}{4}(\bar{\sigma}^2)^{-3/2} < 0 \in [a, b]. \tag{34}$$

Consequently the second derivative is in the interval $[M, m]$ at the given confidence level:

$$-M \leq f''(\bar{\sigma}^2) \leq -m \leq 0 \text{ for } \bar{\sigma}^2 \in [a, b] \tag{35}$$

$$\text{with } M = \frac{1}{4}a^{-3/2} \text{ and } m = \frac{1}{4}b^{-3/2}. \tag{36}$$

The defect formula 2 states that the defect is bounded by

$$\sqrt{\mathbb{E}[\bar{\sigma}^2]} - \mathbb{E}[\sqrt{\bar{\sigma}^2}] \leq \frac{1}{2}M\mathbb{V}[\bar{\sigma}^2]. \tag{37}$$

The constant M was computed above in (35), and the variance of $\bar{\sigma}^2$ is calculated in the next lines, using the first and second moment of the sample variance as stated in 1.

$$\begin{aligned}
\mathbb{V}[\bar{\sigma}^2] &= \mathbb{E}[(\bar{\sigma}^2 - \mathbb{E}[\bar{\sigma}^2])^2] \\
&= \mathbb{E}\left[\left(\frac{n}{N+n}\hat{\sigma}_t^2 - \frac{n}{N+n}\frac{n-1}{n}\sigma_t^2\right)^2\right] \\
&= \frac{n^2}{(N+n)^2}\mathbb{E}\left[\left(\hat{\sigma}_t^2 - \mathbb{E}[\hat{\sigma}_t^2]\right)^2\right] = \frac{n^2}{(N+n)^2}\mathbb{V}\left[\hat{\sigma}_t^2\right] \\
&= \frac{n^2}{(N+n)^2}\frac{2(n-1)}{n^2}\sigma_t^4 = \frac{2(n-1)}{(N+n)^2}\sigma_t^4.
\end{aligned} \tag{38}$$

Inserting $\mathbb{V}[\bar{\sigma}^2]$ computed in (38) and M defined in (35) with a as defined in (31) into the defect formula (37) yields the lower bound:

$$\begin{aligned}
&\sqrt{\mathbb{E}[\bar{\sigma}^2]} - \frac{1}{2}M\mathbb{V}[\bar{\sigma}^2] \\
&= \sqrt{\mathbb{E}[\bar{\sigma}^2]} - \frac{1}{2}\cdot\frac{1}{4}a^{-3/2}\frac{2(n-1)}{(N+n)^2}\sigma_t^4 \\
&= \sqrt{\mathbb{E}[\bar{\sigma}^2]} - \frac{(n-1)}{4(N+n)^2}\sigma_t^4 a^{-3/2} \leq \mathbb{E}[\sqrt{\bar{\sigma}^2}] \\
&\text{with } a = \frac{N}{N+n}\sigma_s^2 + \frac{1}{N+n}\chi^2_{1-\alpha/2,n-1}\sigma_t^2.
\end{aligned} \tag{39}$$

Assuming that source and target variance are of the same order of magnitude σ, the defect will be of order of magnitude σ: The factor $\mathbb{V}[X]$ scales with σ^4 and M with σ^{-3}. □

F.2. Proof of Proposition 1

For two univariate normal distributions with moments μ_t, σ_t^2 and $\bar{\mu}, \bar{\sigma}^2$, the Wasserstein distance as defined in (F.1) reduces to

$$W_2^2 = \sigma_t^2 + \bar{\sigma}^2 - 2\bar{\sigma}\sigma_t + (\bar{\mu} - \mu)^2. \tag{40}$$

The expected value of the Wasserstein distance across many batches is given as

$$\begin{aligned}
\mathbb{E}[W_2^2] &= \sigma_t^2 + \mathbb{E}[\bar{\sigma}^2] - 2\mathbb{E}[\bar{\sigma}]\sigma_t + \mathbb{E}[(\mu_t - \bar{\mu})^2] \\
&= \sigma_t^2 + \frac{N}{N+n}\sigma_s^2 + \frac{n}{N+n}\frac{n-1}{n}\sigma_t^2 \\
&\quad - 2\sigma_t\mathbb{E}\left[\sqrt{\frac{N}{N+n}\sigma_s^2 + \frac{n}{N+n}\hat{\sigma}_t^2}\right] \\
&\quad + \mathbb{E}\left[\left(\mu_t - \frac{N}{N+n}\mu_s - \frac{n}{N+n}\hat{\mu}_t\right)^2\right]
\end{aligned} \tag{41}$$

which can already serve as the basis for our numerical simulations. To arrive at a closed form analytical solution, we invoke Lemma 3 to bound the expectation value $\mathbb{E}[\bar{\sigma}]$ in equation (41).

$$\begin{aligned}
-2\sigma_t\sqrt{\mathbb{E}[\bar{\sigma}^2]} &\leq -2\sigma_t\mathbb{E}\left[\sqrt{\bar{\sigma}^2}\right] \\
&\leq -2\sigma_t\sqrt{\mathbb{E}[\bar{\sigma}^2]} - 2\sigma_t\left(-\frac{1}{2}M\mathbb{V}[\bar{\sigma}^2]\right)
\end{aligned} \tag{42}$$

Apart from the square root term bounded in equation (42) above, the expectation value of the Wasserstein distance can be computed exactly. Hence the bounds on $\mathbb{E}[\bar{\sigma}]$ multiplied by a factor of $(-2\sigma_t^2)$ coming from equation (41) determine lower and upper bounds L and U on the expected value of W_2^2:

$$L \leq \mathbb{E}\left[W_2^2\right] \leq U = L + \sigma_t M\mathbb{V}[\bar{\sigma}^2] \tag{43}$$

In the next lines, the lower bound is calculated:

$$\begin{aligned}
L &= \sigma_t^2 + \frac{N}{N+n}\sigma_s^2 + \frac{n-1}{N+n}\sigma_t^2 \\
&\quad - 2\sigma_t\sqrt{\mathbb{E}\left[\frac{N}{N+n}\sigma_s^2 + \frac{n-1}{N+n}\sigma_t^2\right]} \\
&\quad + \left(\mu_t - \frac{N}{N+n}\mu_s\right)^2 + \frac{n^2}{(N+n)^2}\mathbb{E}[\hat{\mu}_t^2] \\
&\quad - 2\left(\mu_t - \frac{N}{N+n}\mu_s\right)\frac{n}{N+n}\mathbb{E}[\hat{\mu}_t] \\
&= \sigma_t^2 + \frac{N}{N+n}\sigma_s^2 + \frac{n-1}{N+n}\sigma_t^2 \\
&\quad - 2\sigma_t\sqrt{\frac{N}{N+n}\sigma_s^2 + \frac{n-1}{N+n}\sigma_t^2} \\
&\quad + \left(\mu_t - \frac{N}{N+n}\mu_s\right)^2 + \frac{n^2}{(N+n)^2}\left(\frac{1}{n}\sigma_t^2 + \mu_t^2\right) \\
&\quad - 2\left(\mu_t - \frac{N}{N+n}\mu_s\right)\frac{n}{N+n}\mu_t \\
&= \left(\sigma_t - \sqrt{\frac{N}{N+n}\sigma_s^2 + \frac{n-1}{N+n}\sigma_t^2}\right)^2 \\
&\quad + \left(\mu_t - \frac{N}{N+n}\mu_s - \frac{n}{N+n}\mu_t\right)^2 + \frac{n}{(N+n)^2}\sigma_t^2 \\
&= \left(\sigma_t - \sqrt{\frac{N}{N+n}\sigma_s^2 + \frac{n-1}{N+n}\sigma_t^2}\right)^2 \\
&\quad + \frac{N^2}{(N+n)^2}\left(\mu_t - \mu_s\right)^2 + \frac{n}{(N+n)^2}\sigma_t^2
\end{aligned} \tag{44}$$

After having derived the lower bound, the upper bound is the sum of the lower bound and the defect term as derived in Lemma 3.

$$\begin{aligned}\mathbb{E}[W^2] \geq U &= L + \sigma_t M \mathbb{V}[\bar{\sigma}^2] \\ &= L + \sigma_t \frac{1}{4} a^{-3/2} \frac{2(n-1)}{(N+n)^2} \sigma_t^4 \\ &= L + a^{-3/2} \frac{(n-1)}{2(N+n)^2} \sigma_t^5. \\ \text{with } a &= \frac{N}{N+n}\sigma_s^2 + \frac{n}{N+n}\chi^2_{1-\alpha/2,n-1}\frac{\sigma_t^2}{n}\end{aligned} \tag{45}$$

F.3. Extension to multivariate distributions.

We now derive a multivariate variant that can be fit to data from a DNN. Due to the estimation of running statistics in the network, we have access to a diagonal approximation of the true covariance matrix.

We denote the diagonal covariance matrices with matrix elements σ_i^2 as

$$(\Lambda_t)_{ii} = (\sigma_t^2)_i,\ (\hat{\Lambda}_t)_{ii} = (\hat{\sigma}_t^2)_i,\ (\Lambda_s)_{ii} = (\sigma_s^2)_i \tag{46}$$

and extend our definition of the statistics used for normalization to $\bar{\boldsymbol{\mu}}$ and $\bar{\boldsymbol{\Sigma}}$:

$$\bar{\boldsymbol{\mu}} = \frac{N\boldsymbol{\mu}_s + n\hat{\boldsymbol{\mu}}_t}{N+n},\ \bar{\boldsymbol{\Sigma}} = \frac{N\boldsymbol{\Sigma}_s + n\hat{\boldsymbol{\Sigma}}_t}{N+n}. \tag{47}$$

The Wasserstein distance between $\bar{\boldsymbol{\mu}}, \bar{\boldsymbol{\Sigma}}$ and $\boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t$ is then defined as

$$\begin{aligned}W_2^2 &= \mathrm{tr}\Big(\boldsymbol{\Sigma}_t + \bar{\boldsymbol{\Sigma}} - 2\boldsymbol{\Sigma}_t^{1/2}\bar{\boldsymbol{\Sigma}}^{1/2}\Big) + (\boldsymbol{\mu}_t - \bar{\boldsymbol{\mu}})^T(\boldsymbol{\mu}_t - \bar{\boldsymbol{\mu}}) \\ &= \sum_{i=1}^{D} (\sigma_t^2)_i + (\bar{\sigma}^2)_i - 2(\bar{\sigma})_i(\sigma_t)_i + ((\mu_t)_i - (\bar{\mu}_t)_i)^2 \\ &= \sum_{i=1}^{D} (W_2^2)_i\end{aligned} \tag{48}$$

Every component $(W_2^2)_i$ in the sum above is bounded by the univariate bound discussed above. The multivariate Wasserstein distance which sums over the diagonal covariance matrix entries is then bounded by the sums over the individual bounds L_i and U_i given in (18).

$$L_i \leq (W_2^2)_i \leq U_i \Rightarrow \sum_{i=1}^{D} L_i \leq W_2^2 \leq \sum_{i=1}^{D} U_i. \tag{49}$$

F.4. Limits of Proposition 1

Limit $n \to \infty$ In the limit of infinite batch size $n \to \infty$, upper and lower bounds on the expected Wasserstein distance between $\bar{\mu}, \bar{\sigma}^2$ and μ_t, σ_t^2 both go to zero.

$$\begin{aligned}\lim_{n\to\infty} L &= \lim_{n\to\infty} \left(\sigma_t - \sqrt{\frac{N}{N+n}\sigma_s^2 + \frac{n-1}{N+n}\sigma_t^2}\right)^2 \\ &\quad + \lim_{n\to\infty} \frac{N^2}{(N+n)^2}(\mu_t - \mu_s)^2 + \frac{n}{(N+n)^2}\sigma_t^2 \\ &= (\sigma_t - \sigma_t)^2 = 0 \\ \lim_{n\to\infty} U &= \lim_{n\to\infty} L + \lim_{n\to\infty} \sigma_t^5 \frac{(n-1)}{2(N+n)^2} a^{-3/2} = 0.\end{aligned} \tag{50}$$

The intuition behind this limit is that if a large number of samples from the target domain is given, $\hat{\mu}$ and $\hat{\sigma}^2$ approximate the true target statistics very well. As $\hat{\mu}$ and $\hat{\sigma}^2$ dominate $\bar{\mu}$ and $\bar{\sigma}^2$ for large n, the expected Wasserstein distance has to vanish.

Limit $N \to \infty$ In the opposite limit $N \to \infty$, the expected value of the Wasserstein distance reduces to the Wasserstein distance between source and target statistics.

$$\begin{aligned}&\lim_{N\to\infty} \bar{\mu} = \mu_s,\ \lim_{N\to\infty} \bar{\sigma}^2 = \sigma_s^2, \\ \Rightarrow &\lim_{N\to\infty} \mathbb{E}[W_2^2] = \sigma_t^2 + \sigma_s^2 - 2\sigma_t\sigma_s + (\mu_t - \mu_s)^2 \\ &= W_2^2\left(\sigma_s^2, \sigma_t^2, \mu_s, \mu_t\right).\end{aligned} \tag{51}$$

Special case $\mu_t = \mu_s$ and $\sigma_t^2 = \sigma_s^2$ When source and target domain coincide, and the statistics $\sigma_s^2 = \sigma_t^2$ and $\mu_s = \mu_t$ are known, then the source target mismatch is not an error source.

Using the definition of $\bar{\mu}$ and $\bar{\sigma}^2$, the bounds on the expected Wasserstein distance follow from setting σ_t^2 to σ_s^2 and μ_t to μ_s in Proposition 1.

$$\begin{aligned}&\bar{\mu} = \frac{N\mu_t + n\hat{\mu}_t}{N+n},\ \bar{\sigma}^2 = \frac{N\sigma_t^2 + n\hat{\sigma}_t^2}{N+n},\ L \leq \mathbb{E}[W_2^2] \leq U \\ &L = \sigma_t^2\left(\frac{2N^2 + 4Nn - N + 2n^2}{(N+n)^2} - 2\sqrt{1 - \frac{1}{N+n}}\right), \\ &U = L + \sigma_t^2 \frac{n-1}{2(N+n)^2}\left(\frac{N + \chi^2_{1-\alpha/2,n-1}}{N+n}\right)^{-3/2}.\end{aligned} \tag{52}$$

In a third scenario, it is known that source and target statistics are the same but the values of the statistics are unknown. In our model, this is the case where the number of pseudo samples N is set to zero and source and target statistics are equal. Setting $N = 0$ in equation (52) yields

$$\begin{aligned}L &= 2\sigma_t^2\left(1 - \sqrt{1 - \frac{1}{n}}\right), \\ U &= L + \sigma_t^2 \frac{n-1}{2n^2}\left(\frac{\chi^2_{1-\alpha/2,n-1}}{n}\right)^{-3/2}.\end{aligned} \tag{53}$$

G. Full List of Models Evaluated on IN

The following lists contains all models we evaluated on various datasets with references and links to the corresponding source code.

G.1. Torchvision models trained on IN

Weights were taken from `https://github.com/pytorch/vision/tree/master/torchvision/models`

1. `alexnet` (Krizhevsky et al., 2012)
2. `densenet121` (Huang et al., 2017)
3. `densenet161` (Huang et al., 2017)
4. `densenet169` (Huang et al., 2017)

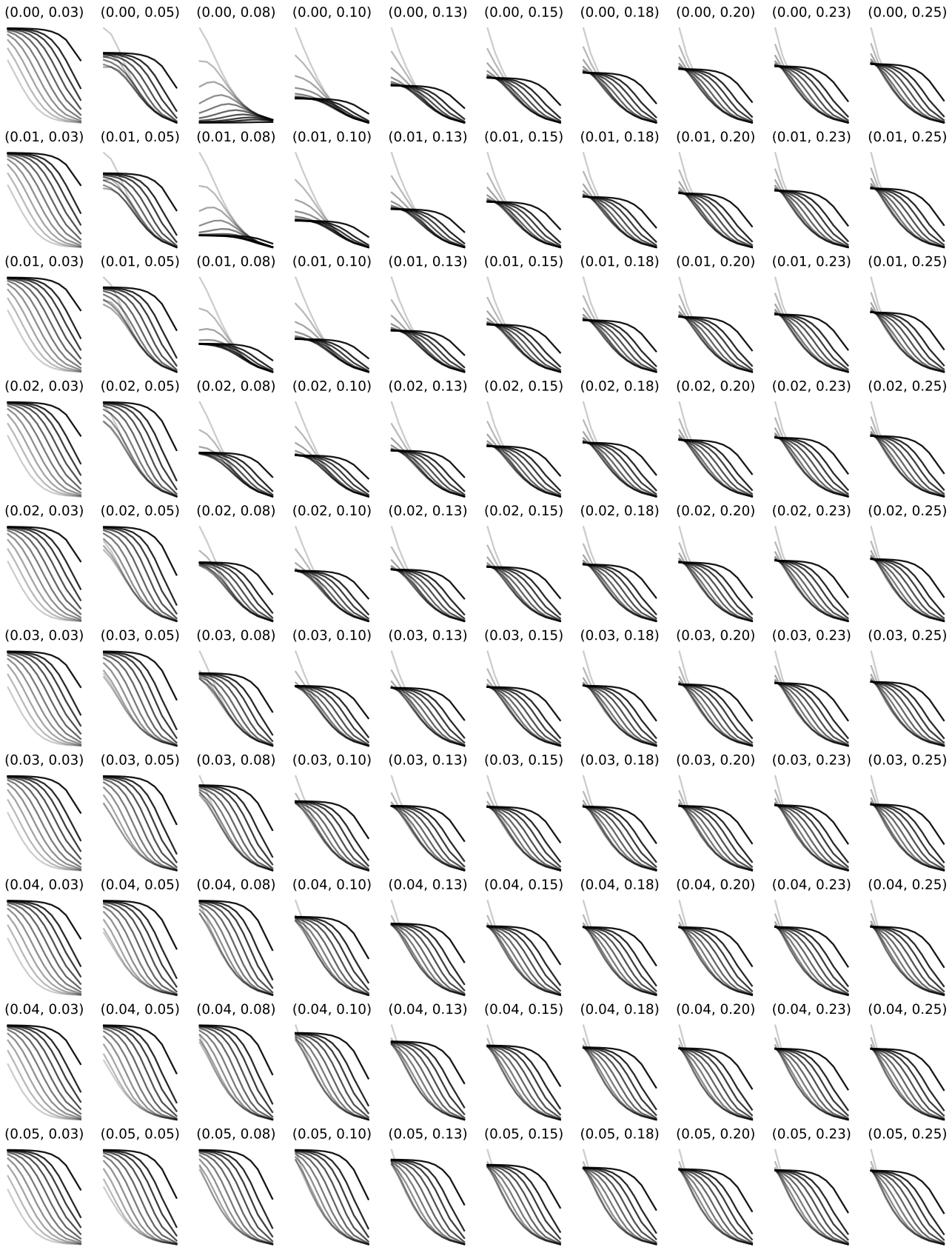

Figure 14: Overview of different parametrizations of the model. We denote each plot with $(\mu_t - \mu_s, \sigma_t/\sigma_s)$ and report the lower bound $\sqrt{L}$ on the Wasserstein distance. Parametrizations in columns four to seven produce qualitatively similar results we observed in our experiments, assuming a linear relationship between the Wasserstein distance and the error rate.

5. densenet201 (Huang et al., 2017)
6. densenet201 (Huang et al., 2017)
7. googlenet (Szegedy et al., 2015)
8. inception_v3 (Szegedy et al., 2016)
9. mnasnet0_5 (Tan et al., 2019)
10. mnasnet1_0 (Tan et al., 2019)
11. mobilenet_v2 (Sandler et al., 2018)
12. resnet18 (He et al., 2016)
13. resnet34 (He et al., 2016)
14. resnet50 (He et al., 2016)
15. resnet101 (He et al., 2016)
16. resnet152 (He et al., 2016)
17. resnext50_32x4d (Xie et al., 2017)
18. resnext101_32x8d (Xie et al., 2017)
19. shufflenet_v2_x0_5 (Ma et al., 2018)
20. shufflenet_v2_x1_0 (Ma et al., 2018)
21. vgg11_bn (Simonyan & Zisserman, 2015)
22. vgg13_bn (Simonyan & Zisserman, 2015)
23. vgg16_bn (Simonyan & Zisserman, 2015)
24. vgg19_bn (Simonyan & Zisserman, 2015)
25. wide_resnet101_2 (Zagoruyko & Komodakis, 2016)
26. wide_resnet50_2 (Zagoruyko & Komodakis, 2016)

G.2. Robust ResNet50 models

1. resnet50 AugMix (Hendrycks et al., 2020) https://github.com/google-research/augmix
2. resnet50 SIN+IN (Geirhos et al., 2019) https://github.com/rgeirhos/texture-vs-shape
3. resnet50 ANT (Rusak et al., 2020) https://github.com/bethgelab/game-of-noise
4. resnet50 ANT+SIN (Rusak et al., 2020) https://github.com/bethgelab/game-of-noise

G.3. Robust ResNext models (Xie et al., 2017)

Weights were taken from: https://github.com/facebookresearch/WSL-Images/blob/master/hubconf.py Note that the baseline resnext50_32x4d model trained on ImageNet is available as part of the torchvision library.

1. resnext50_32x4d WSL
2. resnext101_32x4d WSL

G.4. ResNet50 with Group Normalization (Wu & He, 2018)

Model weights and training code was taken from https://github.com/ppwwyyxx/GroupNorm-reproduce

1. resnet50 GroupNorm
2. resnet101 GroupNorm
3. resnet152 GroupNorm

G.5. ResNet50 with Fixup initialization (Zhang et al., 2019)

Model weights and training code was taken from https://github.com/hongyi-zhang/Fixup/tree/master/imagenet. For training, we keep all hyperparameters at their default values and note that in particular the batchsize of 256 is a sensitive parameter.

1. resnet50 FixUp
2. resnet101 FixUp
3. resnet152 FixUp