Improving the Accuracy-Robustness Trade-Off for Dual-Domain Adversarial Training

Chawin Sitawarin [*1] **Arvind Sridhar** [*1] **David Wagner** [1]

Abstract

While Adversarial Training remains the standard in improving robustness to adversarial attack, it often sacrifices accuracy on natural (clean) samples to a significant extent. Dual-domain training, optimizing on both clean and adversarial objectives, can help realize a better trade-off between clean accuracy and robustness. In this paper, we develop methods to improve dual-domain training for large adversarial perturbations and complex datasets. We first demonstrate that existing methods suffer from poor performance in this setting, due to a poor training procedure and overfitting to a particular attack. Then, we develop novel methods to address these issues. First, we show that adding KLD regularization to the dual training objective mitigates this overfitting and achieves a better trade-off, on CIFAR-10 and a 10-class subset of ImageNet. Then, inspired by domain adaptation, we develop a new normalization technique, Dual Batch Normalization, to further improve accuracy. Combining these two strategies, our model sets a new state of the art in trade-off performance for dual-domain adversarial training.

1. Introduction

The ability of carefully-crafted adversarial examples to fool deep neural networks (Biggio et al., 2013; Szegedy et al., 2014; Goodfellow et al., 2015) presents a significant security vulnerability many safety-critical settings that rely on neural networks. While Adversarial Training (Madry et al., 2018) has become the standard technique for robust training, it degrads the model's performance on natural (clean) samples, an undesirable result for applications where clean performance remains the priority. Currently, there are two popular approaches. The first is to use a hybrid loss function $\gamma L_{\text{cl}} + (1-\gamma)L_{\text{adv}}$, where L_{cl} is the loss on clean samples and L_{adv} on their adversarial counterparts (Goodfellow et al., 2015). We denote this method as Mixed Adversarial Training (MAT). The other technique is TRADES (Zhang et al., 2019), which replaces L_{adv} with a more theoretically-sound KL divergence term. Both methods perform admirably against small adversarial perturbations, but surprisingly, their performance *erodes* when employed against larger perturbations and more complex datasets. In this paper, we explore techniques to gain better control over this trade-off.

We first systematically investigate this phenomenon. Xie & Yuille (2020) posit that when the perturbation and dataset complexity increase, the clean and adversarial distributions diverge, making dual-domain training more difficult. We demonstrate that this difficulty is particularly acute due to a subtle algorithmic error made by nearly all dual-domain training strategies today when the trained network contains Batch Normalization (BN). The standard practice is to forward-propagate these batches through the network sequentially, one after the other. However, this practice results in a mismatch between training and testing behavior of BN, hurting performance. Concatenating these two batches into a single large batch establishes consistency and eliminates this issue. On CIFAR-10 ($\epsilon = 16/255$), concatenated batching improves both clean and robust performance of TRADES by 3.4% and 5.7% respectively (see Table 1).

However, even with concatenated batching, a second, more perplexing problem emerges. Under certain conditions, a network trained with MAT or even TRADES overfits to the weak PGD attack used during training, providing a false indication of robustness. A model suffering from the "false robustness" issue will offer no robustness against stronger attacks like AutoAttack (Croce & Hein, 2020) at test time. This phenomenon was previously observed by Xie & Yuille (2020) on extremely large networks trained with MAT on the ImageNet dataset, and attributed to the BN layers being unable to reconcile the highly-divergent clean and adversarial distributions. We demonstrate that this issue occurs more generally than previously speculated, manifesting even on CIFAR-10 ($\epsilon = 16/255$) with smaller Pre-Act ResNet-18 models, and is not unique to networks with BN.

[*]Equal contribution [1]Department of Computer Science, UC Berkeley, USA. Correspondence to: Chawin Sitawarin <chawins@berkeley.edu>, Arvind Sridhar <arvindsridhar@berkeley.edu>.

Presented at the ICML 2021 Workshop on Uncertainty and Robustness in Deep Learning.,

We further show that the KL divergence term in the TRADES objective acts as a regularizer, driving the intermediate clean/adv. distributions closer together. Adding a similar KL regularizer to the MAT objective can drive the same effects and eliminate its false robustness issue, enabling it to realize all the benefits of BN and achieve better trade-off performance than even TRADES on CIFAR-10 ($\epsilon = 16/255$) (Fig. 2). Finally, we propose Dual Batch Normalization (DBN), an improved formulation of BN for dual-domain training. Our scheme maintains two sets of statistics, one for clean samples and one for adversarial, and utilizes a Gaussian Mixture Model to infer what combination of these statistics to use to normalize a sample (at train and test time). We show that the combination of concatenated batching, KL regularization, and DBN enables MAT to outperform all prior methods and establish a new state-of-the-art trade-off on CIFAR-10 ($\epsilon = \{8, 16\}/255$) & Imagenette ($\epsilon = 16/255$) (see Fig. 3).

2. Background and Related Work

While Adversarial Training (Madry et al., 2018) consistently achieves strong robustness, it often degrades model performance on clean samples, as only the adversarial objective is optimized. To mitigate this, early papers such as Goodfellow et al. (2015) proposed to simply optimize the model on both clean and adversarial samples. We denote this method as Mixed Adversarial Training (MAT), with objective:

$$L_{\text{MAT}}(x, y; \theta) = \gamma L_{\text{CE}}(x, y; \theta) + (1 - \gamma) \max_{\|\delta\|_p \leq \epsilon} L_{\text{CE}}(x + \delta, y; \theta) \quad (1)$$

L_{CE} denotes the cross-entropy loss, and γ controls the trade-off between clean accuracy and robustness. To develop a more theoretically-principled trade-off, Zhang et al. (2019) proposed to minimize the KL divergence between clean & adversarial logits, rather than adversarial cross-entropy. Their method, TRADES, achieves the state-of-the-art trade-off, consistently outperforming MAT (Gowal et al., 2021):

$$L_{\text{TRADES}}(x, y; \theta) = L_{\text{CE}}(x, y; \theta) + \beta \max_{\|\delta\|_p \leq \epsilon} D_{\text{KL}}(f(x; \theta) \parallel f(x + \delta; \theta)) \quad (2)$$

The robust loss weight β controls the trade-off. Tsipras et al. (2019) demonstrated that there may be an inherent tension between robustness and accuracy, making the trade-off inevitable. Optimizing this trade-off thus becomes paramount.

3. Problems with Dual-Domain Training

3.1. Sequential vs. Concatenated Batches

At each training iteration of MAT and TRADES, we train on both the clean and adversarial versions of the input batch. The current standard is to feed the clean/adv. batches to the model *sequentially*, one after the other, rather than *concatenating* them into a single large batch first. The official code for TRADES[1] as well as a re-implementation of MAT by Tramèr et al. (2018)[2] follow this convention. This seemingly inconsequential distinction has not been discussed in prior work, to the extent of our knowledge. Yet, it represents a key algorithmic error that can significantly erode performance.

The only difference between sequential and concatenated batching lies in the BN layer. In sequential batching, the clean and adversarial batches are normalized using their respective mean and variance during training. However, there is only one set of running statistics (used to normalize test examples), updated by both clean and adv. moments. This creates a mismatch between train- and test-time behavior: training normalization occurs with moments that are specific to the particular domain (clean or adversarial), while test normalization uses the "mixed" moments.

Concatenating the clean and adversarial batches during training eliminates this train-test mismatch. The training moments are computed over the concatenated batch, so are mixed themselves. The mixed moments are then used to update the running statistics, resulting in consistent train-test behavior. Fig. 4 confirms our hypothesis, showing that Wasserstein distance between batch and running stats for sequential version of MAT (MAT-Seq) is larger and less stable than for concatenated version (MAT-Cat). Concatenated batching also significantly boosts empirical performance (see Table 1). Notably, for TRADES on CIFAR-10 ($\epsilon = 16/255$), clean and robust accuracies increase by over 3% & 5% respectively. For MAT on Imagenette, concatenation raises clean accuracy by 5% and robustness by 10%.

3.2. False Robustness Problem

Even with concatenated batching, certain training strategies, e.g. MAT-Cat with $\gamma \geq 0.3$ (Fig. 5) or TRADES-Cat with $\beta \leq 0.5$ on CIFAR-10 ($\epsilon = 16/255$), fail to impart robustness to the model (Table 1) when measured by the strong AutoAttack benchmark (Croce & Hein, 2020). Intriguingly, when this problem occurs, the model still exhibits good robustness under the relatively weak 10-step PGD attack used during training, and achieves very high clean accuracy (similar to that of normal training). We will refer to this phenomenon as "false robustness" for the rest of this paper.

Xie & Yuille (2020) noted the same phenomenon when training BN models with MAT-Cat on the large-scale ImageNet dataset ($\epsilon = 16/255$). They attributed the cause to the BN layers being forced to model a heterogeneous distribution

[1] https://github.com/yaodongyu/TRADES.

[2] https://github.com/ftramer/ensemble-adv-training.

Table 1. Accuracy comparison of sequential and concatenated batch dual-domain training (i.e. MAT, TRADES). Pre-activation ResNet-20 is used for CIFAR-10, and ResNet-34 for Imagenette. "Clean" denotes normal accuracy. "PGD" and "AA" respectively represent robust accuracy under 10-step PGD attack (also used during training) and under AutoAttack, the current state-of-the-art robustness benchmark consisting of four different attacks (Croce & Hein, 2020). Concatenated batching achieves nearly universal improvement over sequential. The large drop-off from PGD to AA on MAT-Cat ($\gamma = 0.5$) is explained in Section 3.2. Naming convention persists throughout the paper.

	CIFAR-10 ($\epsilon = 8/255$)			CIFAR-10 ($\epsilon = 16/255$)			Imagenette ($\epsilon = 16/255$)		
Training Method	Clean	PGD	AA	Clean	PGD	AA	Clean	PGD	AA
MAT-Seq ($\gamma = 0.5$)	85.13	45.15	40.77	52.85	**21.96**	**10.08**	67.82	24.60	13.99
MAT-Cat ($\gamma = 0.5$)	**85.53**	**46.03**	**43.29**	**93.95**	33.11	3.70	**72.43**	**28.62**	**23.67**
TRADES-Seq ($\beta = 6$)	**81.26**	50.73	46.89	70.42	24.24	17.18	**68.51**	27.90	22.14
TRADES-Cat ($\beta = 6$)	80.75	**52.06**	**48.14**	**73.82**	**33.16**	**22.89**	62.14	**28.83**	**23.77**

during training, as the distributions of clean and adversarial inputs are highly divergent. To mitigate the issue, they proposed to 1) maintain a separate set of BNs for clean and adv. batches to disentangle the mixture distribution, 2) reduce the number of clean images in the concatenated training batch, and 3) use batch-unrelated normalization layers, such as Instance (IN) or Group (GN) Normalization, instead of BN. We will demonstrate that their hypothesis is *partially correct but incomplete*, particularly contrasting the second and the third points.

First, as Fig. 5 shows, the number of clean images in the concatenated batch has no bearing on false robustness; instead, the *ratio of clean to adversarial loss magnitude*, which rises with γ, is the key driver. For MAT-Cat with $\gamma \leq 0.25$, each training batch still contains the entire clean and adversarial sample set, meaning that the distribution is as heterogeneous as $\gamma \geq 0.3$; yet, the model still achieves strong robustness. Second, this issue is *not unique to BN*. Using IN/GN instead, or even just removing BN, still exhibits false robustness but often requires a larger γ before it manifests (see Table 2).

We hypothesize that, when γ is large and clean loss magnitude high, the model is incentivized to construct more prominent classification "pathways" for clean inputs through the network. Given the dissimilarity between clean and adversarial domains, these clean pathways must inevitably diverge from the model's adversarial pathways, putting pressure on the network to learn two vastly different objectives. As a result, the model overfits to the train-time PGD attack, failing against stronger attacks. We believe this is analogous to the well-known gradient obfuscation phenomenon (Athalye et al., 2018).

The 2D visualization in Fig. 1 (top), as well as Table 2, confirm this hypothesis. For the setting with false robustness (MAT-Cat-0.5), the clean and adversarial distributions are well-separated at various layers, and the sum of Wasserstein distances between these distributions across all layers is high. AutoAttack is able to exploit this large dichotomy between clean/adv. pathways, constructing examples that fall in between the two distributions (Fig. 1). These examples

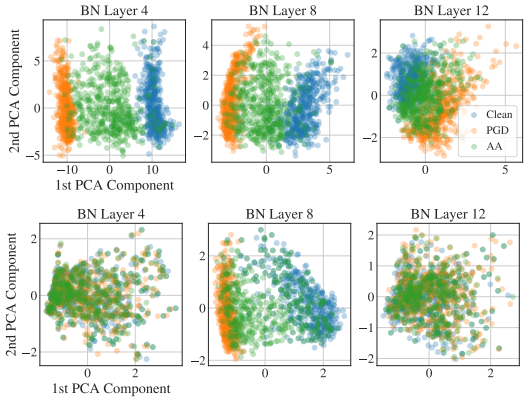


Figure 1. 2D visualization of inputs to three BN layers. **Top:** MAT-Cat-0.5. **Bottom:** MAT-Cat-0.5 + KL ($\beta = 0.4$). Both models are identically trained, apart from the KL regularization. Samples are flattened and then projected to 2D with PCA. Each dot represents one of the 500 test samples.

are considered out-of-distribution for both the network's established clean and adversarial pathways; hence, the model performs poorly on them. Why BN drives this issue at lower values of γ remains to be studied: perhaps BN improves the model's ability to construct divergent clean/adv. pathways.

4. Improving MAT with KL Regularization

While MAT-Cat suffers from false robustness with $\gamma \geq 0.3$, TRADES-Cat remains more resilient to this phenomenon, maintaining good AA robustness even with BN (Table 1). In the TRADES objective (Eqn. 2), clean cross-entropy has a much larger magnitude than the adversarial KL divergence, even with $\beta = 6$; thus, our prior analysis would indicate that TRADES-Cat should exhibit an even greater false robustness issue. However, the distance between intermediate clean and adversarial distributions remains small (Table 2), limiting AA's ability to construct evasive OOD examples.

We hypothesize that the KLD loss, forcing the model to minimize the distance between clean and adversarial output distributions, acts as a *regularizer* for the network. With this regularization, the network is dis-incentivized from constructing highly divergent clean/adv. pathways that drive the

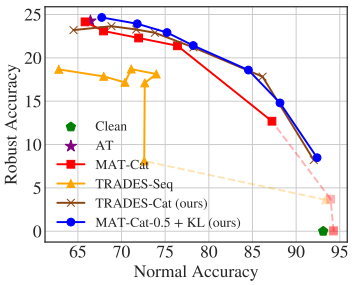


Figure 2. Clean/robust (AA) trade-off comparison for MAT-Cat-0.5 + KL (β varying from 0.25 to 4), TRADES-Cat, and MAT-Cat. All runs are with BN, on CIFAR-10 ($\epsilon = 16/255$). MAT-Cat-0.5 + KL exhibits a superior trade-off compared to all other methods.

false robustness issue, despite the demands of the training objective and the inherent dissimilarity of the two domains.

Thus, we believe that adding a similar KL regularizer term to MAT-Cat can mitigate its false robustness issue. Concretely:

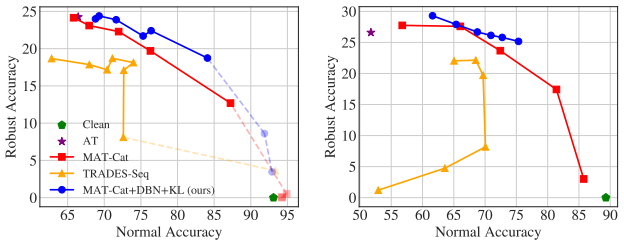
$$L_{\text{MAT+KL}}(x, y; \theta) = L_{\text{MAT}}(x, y; \theta) + \beta D_{\text{KL}}(f(x; \theta) \parallel f(x_{\text{adv}}; \theta)) \quad (3)$$

As Table 2 and Fig. 1 show, adding KL regularization to MAT-Cat-0.5 effectively bridges the gap between the clean and adversarial distributions and pairs the pathways for the two domains. Thus, false robustness is mitigated: as Fig. 2 shows, MAT-Cat-0.5 + KL (varying β) achieves a more optimal clean-robustness trade-off than MAT-Cat and even TRADES-Cat on CIFAR-10 ($\epsilon = 16/255$), with higher clean and AA accuracy in nearly all cases. Thus, MAT+KL is able to harness the full potential of BN to achieve state-of-the-art results for this setting. Next, we propose an improvement to BN itself to further enhance the performance.

5. Dual Batch Normalization

Xie & Yuille (2020) showed models which exclusively route clean and adversarial samples through separate BN layers achieve high accuracy and robustness simultaneously (provided the correct BN is used at test time). This agrees with an observation from domain adaptation, that models can be effectively applied to novel tasks by simply re-computing BN's statistics, no need to fine-tune weights (Li et al., 2017).

However, the lack of a clean/adversarial indicator at test time (telling which BN to use) makes the Separate BN idea infeasible in practice. Detecting adversarial examples is a challenging unsolved problem, and many proposed methods have failed against adaptive adversaries (Carlini & Wagner, 2017; Tramer et al., 2020; Carlini et al., 2019). Nevertheless, it is possible to distinguish clean and adversarial inputs statistically with limited success (Grosse et al., 2017; Pang et al., 2018). This inspires us to build such detection mechanism into BN to create a domain-specific normalization layer, which we call *Dual Batch Normalization* (DBN).



(a) CIFAR-10 ($\epsilon = 16/255$) (b) Imagenette ($\epsilon = 16/255$)

Figure 3. Clean/robust trade-off comparison of our best scheme, MAT-Cat + DBN + KL, with MAT-Cat and TRADES-Seq (current state-of-the-art dual-domain training strategies) in different settings. All runs use BN. The dashed lines indicate false robustness.

5.1. Dual Batch Normalization Algorithm

Consider the typical setting where BN is applied to the output of a convolutional layer. Essentially, DBN consists of two BN's, one for normal inputs ("clean" BN) and the other for adversarial ("adversarial" BN). Each BN's running statistics are updated with those domain-specific moments, i.e. $(\mu_{\text{cl}}, \sigma^2_{\text{cl}})$ for clean BN, and $(\mu_{\text{adv}}, \sigma^2_{\text{adv}})$ for adversarial BN. Any input goes through *both* BN's, and its normalized outputs are combined by a weighted average. The normalized input $\hat{x}$, at both training and test time, is given by:

$$\hat{x} = p_{\text{cl}} \cdot \left(\frac{x - \mu_{\text{cl}}}{\sigma_{\text{cl}}}\right) + p_{\text{adv}} \cdot \left(\frac{x - \mu_{\text{adv}}}{\sigma_{\text{adv}}}\right) \quad (4)$$

The weight, p_{cl} and p_{adv}, is the probability of an input x belonging to the clean and adversarial domain respectively, under an assumption that each distribution is C-dimensional independent Gaussian. Similarly to BN, the parameters $\mu_{\text{cl}}, \sigma_{\text{cl}}, \mu_{\text{adv}}, \sigma_{\text{adv}}$ are computed using the samples from the batch during training and are replaced by the running moments for testing. The running statistics are kept and updated separately for the two BN's and the two domains. See Appendix F and Eqn. 11 for the complete formulation.

5.2. Experimental Results

We combine the improvements we have proposed in this paper (batch concatenation, KL regularization, DBN) and compare the accuracy-robustness trade-off performance with TRADES-Seq and MAT-Cat, the current state-of-the-art dual-domain training strategies, for various settings in Fig. 3. Our MAT-Cat + DBN + KL achieves more optimal trade-off curves than the baselines across the board, particularly with CIFAR-10 and Imagenette ($\epsilon = 16/255$), establishing a new state-of-the-art. Further, we show that MAT-Cat with a small γ can outperform both Adversarial Training and TRADES-Seq, contrary to previous beliefs. The original TRADES or TRADES-Seq produces a very poor trade-off in cases other than CIFAR-10 ($\epsilon = 8/255$), due to the sequential batching error. For additional experiments, an ablation study, and hyperparameter value details, please refer to Appendix F.

Acknowledgements

The first author of this paper was supported by the Hewlett Foundation through the Center for Long-Term Cybersecurity (CLTC), by the Berkeley Deep Drive project, and by gifts from Google and Futurewei.

References

Athalye, A., Carlini, N., and Wagner, D. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. In Dy, J. and Krause, A. (eds.), *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 274–283, Stockholmsmässan, Stockholm Sweden, July 2018. PMLR.

Awais, M., Shamshad, F., and Bae, S.-H. Towards an adversarially robust normalization approach. *arXiv:2006.11007 [cs, stat]*, June 2020.

Benz, P., Zhang, C., and Kweon, I. S. Batch normalization increases adversarial vulnerability: Disentangling usefulness and robustness of model features. *arXiv:2010.03316 [cs, stat]*, October 2020.

Biggio, B., Corona, I., Maiorca, D., Nelson, B., Šrndić, N., Laskov, P., Giacinto, G., and Roli, F. Evasion attacks against machine learning at test time. In Blockeel, H., Kersting, K., Nijssen, S., and Železný, F. (eds.), *Machine Learning and Knowledge Discovery in Databases*, pp. 387–402, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg. ISBN 978-3-642-40994-3.

Carlini, N. and Wagner, D. Adversarial examples are not easily detected: Bypassing ten detection methods. In *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*, AISec '17, pp. 3–14, New York, NY, USA, 2017. Association for Computing Machinery. ISBN 978-1-4503-5202-4. doi: 10.1145/3128572.3140444.

Carlini, N., Athalye, A., Papernot, N., Brendel, W., Rauber, J., Tsipras, D., Goodfellow, I., Madry, A., and Kurakin, A. On evaluating adversarial robustness. *arXiv:1902.06705 [cs, stat]*, February 2019.

Croce, F. and Hein, M. Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks. In III, H. D. and Singh, A. (eds.), *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 2206–2216. PMLR, July 2020.

Ding, G. W., Sharma, Y., Lui, K. Y. C., and Huang, R. MMA training: Direct input space margin maximization through adversarial training. In *International Conference on Learning Representations*, 2020.

Galloway, A., Golubeva, A., Tanay, T., Moussa, M., and Taylor, G. W. Batch normalization is a cause of adversarial vulnerability. *arXiv:1905.02161 [cs, stat]*, May 2019.

Goodfellow, I., Shlens, J., and Szegedy, C. Explaining and harnessing adversarial examples. In *International Conference on Learning Representations*, 2015.

Gowal, S., Qin, C., Uesato, J., Mann, T., and Kohli, P. Uncovering the limits of adversarial training against norm-bounded adversarial examples. *arXiv:2010.03593 [cs, stat]*, March 2021.

Grosse, K., Manoharan, P., Papernot, N., Backes, M., and McDaniel, P. On the (statistical) detection of adversarial examples. *arXiv:1702.06280 [cs, stat]*, October 2017.

He, K., Zhang, X., Ren, S., and Sun, J. Identity mappings in deep residual networks. In *European Conference on Computer Vision*, pp. 630–645. Springer, 2016.

Ioffe, S. and Szegedy, C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning*, pp. 448–456. PMLR, 2015.

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

Madry, A., Makelov, A., Schmidt, L., Tsipras, D., and Vladu, A. Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations*, 2018.

Pang, T., Du, C., Dong, Y., and Zhu, J. Towards robust detection of adversarial examples. In Bengio, S., Wallach, H., Larochelle, H., Grauman, K., Cesa-Bianchi, N., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

Rebuffi, S.-A., Gowal, S., Calian, D. A., Stimberg, F., Wiles, O., and Mann, T. Fixing data augmentation to improve adversarial robustness. *arXiv:2103.01946 [cs]*, March 2021.

Santurkar, S., Tsipras, D., Ilyas, A., and Mądry, A. How does batch normalization help optimization? In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, NIPS'18, pp. 2488–2498, Red Hook, NY, USA, 2018. Curran Associates Inc.

Shafahi, A., Najibi, M., Ghiasi, M. A., Xu, Z., Dickerson, J., Studer, C., Davis, L. S., Taylor, G., and Goldstein, T. Adversarial training for free! In Wallach, H., Larochelle,

H., Beygelzimer, A., dAlché-Buc, F., Fox, E., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., and Fergus, R. Intriguing properties of neural networks. In *International Conference on Learning Representations*, 2014.

Tramèr, F., Kurakin, A., Papernot, N., Goodfellow, I., Boneh, D., and McDaniel, P. Ensemble adversarial training: Attacks and defenses. In *International Conference on Learning Representations*, 2018.

Tramer, F., Carlini, N., Brendel, W., and Madry, A. On adaptive attacks to adversarial example defenses. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M. F., and Lin, H. (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 1633–1645. Curran Associates, Inc., 2020.

Tsipras, D., Santurkar, S., Engstrom, L., Turner, A., and Madry, A. Robustness may be at odds with accuracy. In *International Conference on Learning Representations*, 2019.

Wang, Y., Zou, D., Yi, J., Bailey, J., Ma, X., and Gu, Q. Improving adversarial robustness requires revisiting misclassified examples. In *International Conference on Learning Representations*, 2020.

Wong, E., Rice, L., and Kolter, J. Z. Fast is better than free: Revisiting adversarial training. In *International Conference on Learning Representations*, 2020.

Wu, D., Xia, S.-T., and Wang, Y. Adversarial weight perturbation helps robust generalization. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M. F., and Lin, H. (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 2958–2969. Curran Associates, Inc., 2020.

Xie, C. and Yuille, A. Intriguing properties of adversarial training at scale. In *International Conference on Learning Representations*, 2020.

Zhang, H., Yu, Y., Jiao, J., Xing, E. P., Ghaoui, L. E., and Jordan, M. I. Theoretically principled trade-off between robustness and accuracy. In *International Conference on Machine Learning*, 2019.

A. Related Work

A.1. Adversarial Training

The popular Adversarial Training paradigm developed by Madry et al. (2018) involves generating loss-maximizing adversarial examples from each batch of training data, constrained within an ℓ_p ball of radius ϵ. It then computes the expected adversarial loss over the perturbed batch x_{adv} and trains the model to minimize this loss. Mathematically, AT formulates the following saddle point optimization problem, where θ are the parameters of the model and $\{(x_i, y_i)\}_{i=1}^n$ the training set:

$$\arg\min_{\theta} \frac{1}{n}\sum_{i=1}^{n} \max_{\delta:\|\delta\|_p\leq\epsilon} L_{\text{CE}}(x_i+\delta, y_i; \theta) \tag{5}$$

Madry et al. (2018) used multi-step PGD to compute adversarial examples (inner maximization) and standard optimizers, e.g., SGD or Adam, to train the network, using the cross-entropy loss. Importantly, Adversarial Training only trains on the adversarial loss, ignoring the standard objective. Several works have since sought to improve Adversarial Training, by reducing computation time (Shafahi et al., 2019; Wong et al., 2020) and trying different loss functions to improve robustness (Zhang et al., 2019; Ding et al., 2020; Wang et al., 2020; Wu et al., 2020; Gowal et al., 2021; Rebuffi et al., 2021).

A.2. Batch Normalization

Batch Normalization (BN) (Ioffe & Szegedy, 2015) has become a crucial component of modern state-of-the-art computer vision models, helping them train faster and optimize more effectively. Given a batch $\{x_i\}_{i=1}^{\mathcal{B}}$, Batch Normalization performs the following normalization on each x_i:

$$BN(x_i) = \left(\frac{x_i - \mu_{\mathcal{B}}}{\sqrt{\sigma_{\mathcal{B}}^2+\epsilon}}\right)\gamma + \beta \tag{6}$$

Here, $\mu_{\mathcal{B}}$ and $\sigma_{\mathcal{B}}^2$ denote the channel-wise mean and variance of the input batch $\{x_i\}_{i=1}^{\mathcal{B}}$. γ and β are the trainable scale and shift parameters, respectively. At test time, the concept of a "batch" disappears; thus, BN uses running statistics $\mu_{\mathcal{R}}$ and $\sigma_{\mathcal{R}}^2$, computed during training as an exponential moving average of the training batch moments $\mu_{\mathcal{B}}$ and $\sigma_{\mathcal{B}}^2$, to normalize each test point x_i.

The reasons driving BN's empirical performance improvement in the natural setting have been extensively studied. Ioffe & Szegedy (2015) postulated that BN standardizes the input distribution to each layer of the network, mitigating an "internal covariate shift" phenomenon and enabling more independent training layer-by-layer. Digging deeper, Santurkar et al. (2018) found that BN instead improves training speed and stability by increasing the β-smoothness of the optimization landscape. Several studies have claimed that BN increases adversarial vulnerability of the model, particularly for adversarial training with large perturbations on complex datasets (Xie & Yuille, 2020; Galloway et al., 2019; Awais et al., 2020; Benz et al., 2020). In this study, we dig deeper into these claims in order to better understand BN's impact on adversarial training.

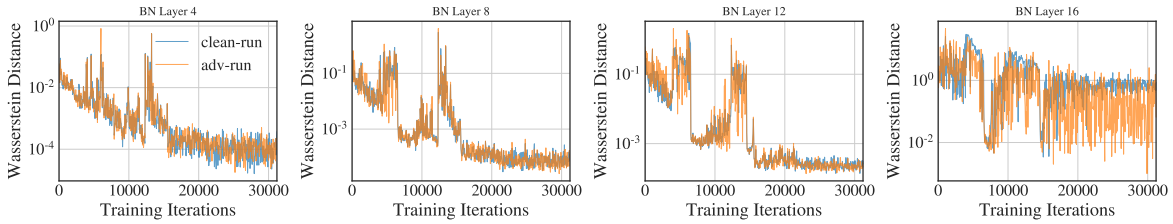
B. Experiment Setup

We provide detailed descriptions of our training frameworks below. The DNNs we experiment with all use ReLU as the activation function and are trained using SGD (momentum of 0.9). We use early stopping during training, i.e., we evaluate the model at the end of each epoch and only save the checkpoint with the highest sum of clean and adversarial validation accuracy. Dataset-specific training details are provided below.

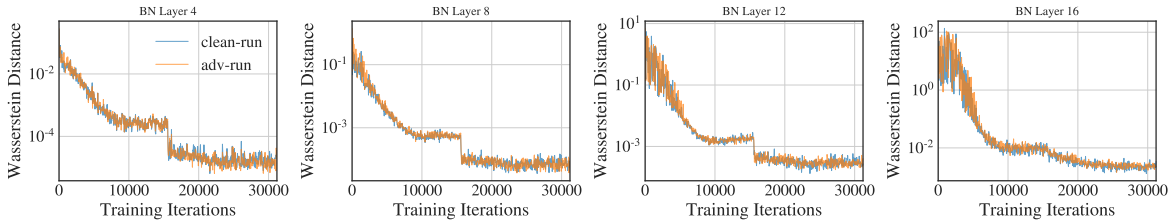
CIFAR-10. We use a Pre-Activation ResNet-18 model (He et al., 2016) for all experiments and perturbation budgets ϵ. Unless otherwise specified, we use the standard formulation of this model, with Batch Normalization (BN) layers within each block. We train all models for 80 epochs with a batch size of 128 (thus, concatenated clean-adversarial batches have a batch size of 256). We use untargeted 10-step PGD with step size of $2/255$ for $\epsilon = 8/255$ and $4/255$ for $\epsilon = 16/255$, and uniform random initialization (no restarts), for training. The initial learning rate is set to 0.05, and is decreased by a factor of 10 at epochs 40, 55, and 70. Weight decay is set to 5×10^{-4}. Standard data augmentation (random crop, flip, scaling, brightness jitter) and input normalization is also used.

Imagenette. The hyperparameters are almost identical to those of CIFAR-10. We use a ResNet-34 model with BN, training for 80 epochs with a batch size of 64. We use SGD, with initial learning rate set to 0.1 and decreased by a factor of 10 at epochs 40, 55, and 70. Weight decay is set to 5×10^{-4}. Attack parameters for Imagenette ($\epsilon = 16/255$) are the same as for CIFAR-10 ($\epsilon = 16/255$).

All of our code is written in PyTorch, and we train all our models on servers with multiple NVIDIA 1080ti and 2080ti GPUs. Each server has 12 cores of Intel(R) Xeon(R) E5-2690 (2.60GHz) CPU and 252 GB of memory. With this setup, a single Adversarial Training run on CIFAR-10 takes roughly 6 hours, with MAT and TRADES taking roughly 7 and 8 hours respectively. After saving the checkpoint with the best clean and robust performance on a hold-out validation set (randomly chosen from 10% of the training samples for CIFAR-10 and Imagenette), we then evaluate this model on the entire test set. For AutoAttack (Croce & Hein, 2020), we use the standard attack mechanism (APGD-CE, APGD-T, FAB-T, and Square) with default parameters.



(a) MAT-Seq ($\gamma = 0.25$)



(b) MAT-Cat ($\gamma = 0.25$)

Figure 4. Wasserstein distance between clean (blue) / adversarial (orange) batch moments and BN's running statistics at different training iterations. Both models trained on CIFAR-10 with MAT ($\gamma = 0.25$), $\epsilon = 16/255$.

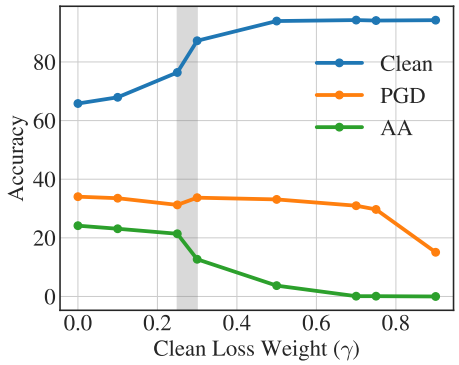


Figure 5. MAT-Cat on CIFAR-10 ($\epsilon = 16/255$) exhibits the false robustness issue when $\gamma \geq 0.3$ (all with BN). With γ increasing, clean accuracy increases and AA decreases as expected. However, PGD falsely rises after $\gamma = 0.3$, staying near 33% until $\gamma = 0.75$.

C. Additional Comparisons between Sequential and Concatenated Batching

Fig. 4 demonstrates the distinction between MAT-Seq and MAT-Cat. Assuming that inputs to each BN layer follow an independent normal distribution, we compute Wasserstein distance between the clean/adversarial batch and the running statistics. In MAT-Seq, both distances are unstable and shoot up multiple times over the training, whereas the distances in MAT-Cat steadily converges to zero. This empirically confirms our analysis and indicates that there is a larger difference between the training and test normalization for MAT-Seq compared to MAT-Cat.

D. Additional Experiments on the False Robustness Problem

Fig. 5 shows the clean accuracy and the accuracy under PGD and AA attacks. The false robustness problem occurs when there is a large gap between PGD and AA accuracies for $\gamma \geq 0.3$. We can also see that when this issue starts to take effects, PGD and AA accuracies diverge, i.e. PGD accuracy unexpectedly increases when γ goes from 0.25 to 0.3 (gray area in Fig. 5). At the same time, AA accuracy decreases and clean accuracy increases as expected, even though the trend immediately becomes steeper.

Table 2 shows that the false robustness phenomenon can happen on Instance and Group normalization as well as no normalization at all. All these cases, denoted in red, share a common characteristic which is a larger Wasserstein distance between any pair of the distributions (Clean, PGD, AA) by about an order of magnitude. To compute this Wasserstein distance, we use the batch mean and variance and assume that all data follow independent Gaussian distributions. For two univariate Gaussians with means μ_1, μ_2 and standard deviations σ_1, σ_2, the (squared) 2-Wasserstein distance is given by

$$W_2^2(\mathcal{N}(\mu_1, \sigma_1^2), \mathcal{N}(\mu_2, \sigma_2^2)) = (\mu_1 - \mu_2)^2 + (\sigma_1 - \sigma_2)^2. \tag{7}$$

This can be easily generalized to multivariate independent Gaussians by summing across all the dimensions.

E. Additional Experiments with KLD Regularization for MAT

Table 3 compares MAT-Cat-0.5 + KL to various comparable strategies representing the current state-of-the-art. Our

Table 2. Wasserstein distances (summed over all layers) between the clean, PGD-10, and AA test set distributions for various train and network normalizer settings (all CIFAR-10, $\epsilon = 16/255$). "Acc. Diff." denotes the difference in accuracy under PGD-10 and AA attacks, marked in red for models with the false robustness issue.

Training Method	Normalizer	Acc. Diff.	Wasserstein Distance		
			Clean-PGD	Clean-AA	PGD-AA
Adversarial Training	BN	9.77	1.6057	0.5072	1.2728
MAT-Cat-0.25	BN	9.85	1.5499	0.6168	1.2830
MAT-Cat-0.5	BN	29.41	**25.0450**	13.5271	12.5853
MAT-Cat-0.5	IN	27.48	**99.8295**	51.9016	48.6722
MAT-Cat-0.75	GN	29.10	**58.2817**	25.1242	36.8039
MAT-Cat-0.5	None	7.74	6.1931	2.7366	4.6481
MAT-Cat-0.75	None	30.08	**278.7880**	109.8840	177.3556
TRADES-Cat-6	BN	10.27	1.5399	0.566	1.0391
MAT-Cat-0.5 + KL-0.4	BN	13.64	2.5128	1.3143	1.4567

Table 3. Comparing performance of MAT-Cat-0.5 + KL with comparable training strategies for various β. Our method demonstrates better results in nearly all cases, achieving the state of the art in large-ϵ dual-domain training. All runs are with BN unless otherwise stated, on CIFAR-10 ($\epsilon = 16/255$). KL-x denotes added KL regularization with $\beta = x$.

Training Method	Clean	PGD	AA
MAT-Cat-0.5, no BN	79.03	23.33	15.59
MAT-Cat-0.5 + KL-0.4 (ours)	**84.53**	**32.23**	**18.59**
MAT-Cat-0.25	76.41	**31.24**	21.39
MAT-Cat-0.5 + KL-0.5 (ours)	**78.21**	30.41	**21.42**
TRADES-Cat-6	73.82	**33.16**	22.89
MAT-Cat-0.5 + KL-1 (ours)	**75.21**	32.19	**22.92**
Adversarial Training	66.44	34.03	24.26
MAT-Cat-0.5 + KL-4 (ours)	**67.73**	**34.68**	**24.66**

method outperforms these other techniques in nearly all cases, even achieving greater robustness than Adversarial Training for $\beta = 4$ (along with higher clean accuracy).

F. Additional Details and Experiments on Dual Batch Normalization

Extending from Section 5, we include the full formulation of DBN here. Once again, the batch moments as well as the normalization are computed as follows:

$$\mu_{\text{cl}} = \frac{1}{N}\sum_{i=1}^{N} x_{\text{cl}}^i,\ \sigma_{\text{cl}}^2 = \frac{1}{N}\sum_{i=1}^{N}(x_{\text{cl}}^i - \mu_{\text{cl}})^2 \tag{8}$$

$$\mu_{\text{adv}} = \frac{1}{N}\sum_{i=1}^{N} x_{\text{adv}}^i,\ \sigma_{\text{adv}}^2 = \frac{1}{N}\sum_{i=1}^{N}(x_{\text{adv}}^i - \mu_{\text{adv}})^2 \tag{9}$$

$$\hat{x} = p_{\text{cl}} \cdot \left(\frac{x - \mu_{\text{cl}}}{\sigma_{\text{cl}}}\right) + p_{\text{adv}} \cdot \left(\frac{x - \mu_{\text{adv}}}{\sigma_{\text{adv}}}\right) \tag{10}$$

where p_{cl} and p_{adv} are defined below:

$$p_{\text{cl}} \coloneqq p(z = \text{cl} \mid \boldsymbol{X} = \boldsymbol{x}) \tag{11}$$

$$= \frac{\mathcal{N}(\boldsymbol{x} \mid \boldsymbol{\mu}_{\text{cl}}, \boldsymbol{\sigma}_{\text{cl}}^2)}{\mathcal{N}(\boldsymbol{x} \mid \boldsymbol{\mu}_{\text{cl}}, \boldsymbol{\sigma}_{\text{cl}}^2) + \mathcal{N}(\boldsymbol{x} \mid \boldsymbol{\mu}_{\text{adv}}, \boldsymbol{\sigma}_{\text{adv}}^2)} \tag{12}$$

$$p_{\text{adv}} \coloneqq p(z = \text{adv} \mid \boldsymbol{X} = \boldsymbol{x}) \tag{13}$$

$$= \frac{\mathcal{N}(\boldsymbol{x} \mid \boldsymbol{\mu}_{\text{adv}}, \boldsymbol{\sigma}_{\text{adv}}^2)}{\mathcal{N}(\boldsymbol{x} \mid \boldsymbol{\mu}_{\text{cl}}, \boldsymbol{\sigma}_{\text{cl}}^2) + \mathcal{N}(\boldsymbol{x} \mid \boldsymbol{\mu}_{\text{adv}}, \boldsymbol{\sigma}_{\text{adv}}^2)} \tag{14}$$

$$= 1 - p_{\text{cl}} \tag{15}$$

$$\text{where}\quad \mathcal{N}(\boldsymbol{x} \mid \boldsymbol{\mu}, \boldsymbol{\sigma}^2) \coloneqq \prod_{j=1}^{C} \frac{1}{\sqrt{2\pi}\sigma_j} e^{-\frac{(x_j - \mu_j)^2}{2\sigma_j^2}} \tag{16}$$

In other words, $p(z = \text{cl} \mid \boldsymbol{X} = \boldsymbol{x})$ is the posterior probability of the domain indicator z being "clean" given an input $\boldsymbol{x}$. Note that we use the bolded variables to denote vectors in $\mathbb{R}^C$. This is a direct consequence of Bayes' rule which is shown here for completeness:

$$p(\text{cl} \mid \boldsymbol{x}) = \frac{p(\boldsymbol{x} \mid \text{cl})p(\text{cl})}{p(\boldsymbol{x})} \tag{17}$$

$$= \frac{\mathcal{N}(\boldsymbol{x} \mid \boldsymbol{\mu}_{\text{cl}}, \boldsymbol{\sigma}_{\text{cl}}^2) \cdot 0.5}{p(\boldsymbol{x} \mid \text{cl})p(\text{cl}) + p(\boldsymbol{x} \mid \text{adv})p(\text{adv})} \tag{18}$$

$$= \frac{\mathcal{N}(\boldsymbol{x} \mid \boldsymbol{\mu}_{\text{cl}}, \boldsymbol{\sigma}_{\text{cl}}^2) \cdot 0.5}{\mathcal{N}(\boldsymbol{x} \mid \boldsymbol{\mu}_{\text{cl}}, \boldsymbol{\sigma}_{\text{cl}}^2) \cdot 0.5 + \mathcal{N}(\boldsymbol{x} \mid \boldsymbol{\mu}_{\text{adv}}, \boldsymbol{\sigma}_{\text{adv}}^2) \cdot 0.5} \tag{19}$$

Note that we remove the random variables, $\boldsymbol{X}$ and z, to declutter. As we are using the equal number of clean and adversarial samples, the priors are uniform, i.e. $p(z = \text{cl}) = p(z = \text{adv}) = 0.5$. In practice, we compute the log probability instead and use the sigmoid function to obtain the posterior distribution or the score to avoid numerical instability. Similarly to BN, the parameters $\mu_{\text{cl}}, \sigma_{\text{cl}}, \mu_{\text{adv}}, \sigma_{\text{adv}}$ are computed using the samples from the batch during training and are replaced by the running moments for testing. The statistics are computed and updated separately for the

Table 4. Normal and robust accuracy of TRADES-Cat and MAT-Cat when augmented with our DBN and KLD regularization techniques. DBN improves the accuracy slightly in all cases. The KLD regularization eliminates the false robustness problem (marked in red) and slightly improves the trade-off. We use $\beta = 6$ for TRADES-Cat, $\gamma = 0.5$ for MAT-Cat, and $\beta = 1$ for MAT-Cat-0.5 + KL.

Method	CIFAR-10 ($\epsilon = 8/255$)			CIFAR-10 ($\epsilon = 16/255$)			Imagenette ($\epsilon = 16/255$)		
	Clean	PGD	AA	Clean	PGD	AA	Clean	PGD	AA
TRADES-Cat	82.20	52.06	48.10	73.82	33.16	22.89	62.14	28.83	23.77
+ DBN	81.68	**53.65**	**49.01**	74.79	**36.91**	22.63	64.89	31.59	24.79
MAT-Cat	85.53	46.03	43.29	93.95	33.11	3.70	72.43	28.62	23.67
+ DBN	**87.64**	48.05	43.85	93.60	32.19	0.98	**75.34**	31.69	25.17
+ KL	84.28	51.42	47.38	75.21	32.19	**22.92**	66.29	33.50	**27.39**
+ DBN + KL	82.97	51.96	48.05	**76.39**	31.39	22.41	72.74	**33.89**	25.81

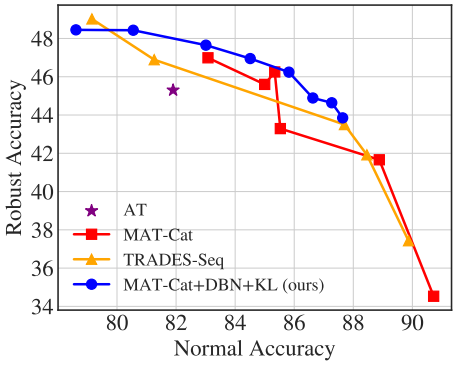


Figure 6. Accuracy-robustness trade-off of our method and the baselines on CIFAR-10 ($\epsilon = 8/255$). This is omitted from Fig. 3, but the labels and layout remain the same as they are in Fig. 3.

two domains. When p_{cl} (or p_{adv}) is always 1, we recover the normal BN, and when p_{cl} and p_{adv} are binary and always assigned correctly, DBN results in the Separate BN.

The affine parameters, $\boldsymbol{w}, \boldsymbol{b} \in \mathbb{R}^C$, of DBN are "shared" between the two domains and applied to the final normalized output.

$$\text{DBN}(\boldsymbol{x}) = \boldsymbol{w} \odot \hat{\boldsymbol{x}} + \boldsymbol{b} \tag{20}$$

Additionally, to encourage the correct matching between clean (and adversarial) samples and clean (and adversarial) BN, we add a negative log-likelihood loss which is summed across all pixels and all DBN layers.

$$\begin{aligned} L_{\text{DBN}}(\boldsymbol{x}) \coloneqq & - \mathbb{1}\{\boldsymbol{x} \in \text{cl}\} \cdot \log(p_{\text{cl}}) \\ & - \mathbb{1}\{\boldsymbol{x} \in \text{adv}\} \cdot \log(1 - p_{\text{cl}}) \end{aligned} \tag{21}$$

F.1. Additional Experiments

Table 4 shows an ablation study with each of the components we introduced being added to MAT-Cat and TRADES-Cat. While DBN does not solve the false robustness problem on MAT-Cat ($\gamma = 0.5$) on CIFAR-10 ($\epsilon = 16/255$), it offers an improvement for $\epsilon = 8/255$ and for Imagenette. It is expected that DBN also overfits to the PGD attack and hence, suffers from the same false robustness problem because it is designed to utilize the two domains more effectively. However, the KLD regularization easily mitigates this issue on both BN and DBN, with a minor improvement to boost.