
Distributionally Robust Deep Learning as a Generalization of Adversarial Training

Matthew Staib
MIT CSAIL
mstaib@mit.edu

Stefanie Jegelka
MIT CSAIL
stefje@mit.edu

Abstract

Machine learning models are vulnerable to adversarial attacks at test time: a correctly classified test example can be slightly perturbed to cause a misclassification. Training models that are robust to these attacks, and theoretical understanding of such defenses are active research areas. Adversarial Training (AT) via robust optimization is a promising approach, where the model is trained against an adversary acting on the training set, but it is less clear how to reason about perturbations on the unseen test set. Distributionally Robust Optimization (DRO) with Wasserstein distance is an interesting theoretical tool for understanding robustness and generalization, but it has been limited algorithmically to simple models. We link DRO and AT both theoretically and algorithmically: AT is a special case of DRO, and in general DRO yields a stronger adversary. We also give an algorithm for DRO for neural networks that is no more expensive than AT.

1 Introduction

Machine learning models are vulnerable to adversarial examples [30, 13]: given a model and a correctly predicted test example, the example can be perturbed by a small amount so that it is misclassified. There has been much recent work on constructing these *adversarial examples* [21, 5, 13, 31]. There are also many *defenses*, or ways of training models so that they are less vulnerable [23, 6, 18, 20, 19]. Adversarial Training (AT) seeks a model that is robust to specific adversarial perturbations on the training set. AT can be viewed as an attempt to solve a certain robust optimization problem [27, 18] against what we call a *pointwise* adversary, that independently perturbs each example. However, we still lack a complete theoretical understanding of this process. In particular, how well does robustness on the training set generalize to robustness on an unseen test set?

Distributionally Robust Optimization (DRO) is a promising theoretical tool that links generalization and robustness [11, 4, 8, 7, 12, 2]. DRO seeks a model that performs well under adversarial joint perturbations of the *entire training set*. Of the many approaches to DRO, we focus on the case where the distributional perturbations are measured with p -Wasserstein distance W_p as in e.g. [11, 3, 26]. W_p -DRO yields generalization guarantees but has been limited algorithmically to simple models.

Contributions. We link DRO and AT both theoretically and algorithmically: **1.** We show DRO with W_p distance is stronger than and in fact generalizes the standard pointwise AT adversary. **2.** We give a general algorithmic framework for training distributionally robust models. Our algorithms work for any reasonable choice of pointwise norm and any $p > 1$, and admit efficient one-step approximations similar to the Fast Gradient Sign Method [13]. **3.** We demonstrate the effectiveness of our method and raise several new questions about generalization, robustness, and optimization.

Finally, we note that in parallel with our work, [28] gave a DRO algorithm for differentiable models. They provide different theory that does not capture our results on mathematically relating DRO to AT. Their algorithm makes different tradeoffs between complexity, generality, guarantees, and speed.

2 Background

Notation. Throughout the paper, we have a training set of n pairs $(x_i, y_i) \in \mathcal{X} \times \mathcal{Y}$ drawn iid from \mathcal{D} , where x_i are examples and y_i are class labels. Denote by \mathcal{D}_n the empirical distribution of the pairs. We seek to learn model parameters θ minimizing a loss function $\ell(x, y; \theta)$. Let $d(x, \tilde{x})$ be a metric on \mathcal{X} ; it will depend on context and is used to define (1) balls $B_\epsilon(x) = \{\tilde{x} : d(x, \tilde{x}) \leq \epsilon\}$ around x , and (2) the p -Wasserstein distance $W_p(\mathcal{D}, \tilde{\mathcal{D}}) = \inf\{\int d(x, y)^p d\gamma(x, y) : \gamma \in \Pi(\mathcal{D}, \tilde{\mathcal{D}})\}^{1/p}$ where $\Pi(\mathcal{D}, \tilde{\mathcal{D}})$ are couplings of $\mathcal{D}, \tilde{\mathcal{D}}$ [32]. By X we mean the matrix with columns x_1, \dots, x_n . For $S \subseteq \{1, \dots, n\}$ denote by X_S the matrix whose columns are $\{x_i\}_{i \in S}$. Throughout the paper, the adversary perturbs only the examples x_i , not the labels y_i .

2.1 Adversarial Training and Robust Optimization

First, assume a *pointwise* attack model where the adversary can vary each input within an ϵ -ball. We seek training methods to make deep models robust to such adversaries. As observed in e.g. [27] and later [18], such a model can be written as the solution to a robust optimization problem against a specific adversary:

$$\min_{\theta} \mathbb{E}_{(x,y) \sim \mathcal{D}} \left[\max_{\tilde{x} \in B_\epsilon(x)} \ell(\tilde{x}, y; \theta) \right]. \quad (1)$$

In practice, we do not have access to \mathcal{D} , so past efforts have sought to solve the empirical version:

$$\min_{\theta} \mathbb{E}_{(x,y) \sim \mathcal{D}_n} \left[\max_{\tilde{x} \in B_\epsilon(x)} \ell(\tilde{x}, y; \theta) \right] = \min_{\theta} \frac{1}{n} \sum_{i=1}^n \max_{\tilde{x}_i \in B_\epsilon(x_i)} \ell(\tilde{x}_i, y_i; \theta). \quad (2)$$

Solving even the empirical version is a difficult nonconvex problem. In practice, the inner problem is solved approximately at each iteration, either via a one-step linear approximation called the Fast Gradient Sign Method (FGSM) [13], or an iterative variant (IFGSM) [15, 16]. The resulting adversarial \tilde{x}_i is used to estimate the gradient for θ . This ‘‘pointwise’’ procedure is called Adversarial Training (AT). Learning the parameters θ via AT yields robust models in practice, but it is not clear to what extent robustness will generalize to adversarial perturbations of a held-out test set.

2.2 Distributionally Robust Optimization

Distributionally Robust Optimization (DRO) seeks to optimize θ in the face of a stronger adversary. In particular, the adversary is not limited to moving points x individually, but can move the *entire distribution* within an ϵ -ball of \mathcal{D}_n for some notion of distance between distributions. We focus on DRO defined with respect to Wasserstein distance W_p between distributions as in [11]:

$$\min_{\theta} \max_{\tilde{\mathcal{D}} : W_p(\mathcal{D}_n, \tilde{\mathcal{D}}) \leq \epsilon} \mathbb{E}_{(x,y) \sim \tilde{\mathcal{D}}} [\ell(x, y; \theta)]. \quad (3)$$

This is especially natural in the typical finite sample setting: we perform DRO with respect to the empirical distribution \mathcal{D}_n , with the hope that the true \mathcal{D} is close enough to lie in the ϵ -ball, hence bounding the population risk. Indeed, for n sufficiently large, we will have $W_p(\mathcal{D}_n, \mathcal{D}) \leq \epsilon$ by weak convergence. Performing DRO with respect to the empirical distribution \mathcal{D}_n both encapsulates the desired adversarial element and has an eye toward good generalization: DRO estimators enjoy certain generalization guarantees [11, 3, 9, 8]. For some simple models, DRO is the same as empirical risk minimization (ERM) with a specific regularizing term [3]. However, for general models, it is less clear how to best perform DRO in practice.

3 Theoretical connections and Algorithms

For now, we study worst-case adversaries for a fixed model. We first show that the distributional adversary from Wasserstein DRO (3) is stronger than and generalizes the typical, pointwise adversary (1):

Proposition 3.1. *Fix model parameters θ . For any $p \in [1, \infty]$,*

$$\mathbb{E}_{(x,y) \sim \mathcal{D}} \left[\max_{\tilde{x} \in B_\epsilon(x)} \ell(\tilde{x}, y; \theta) \right] \leq \max_{\tilde{\mathcal{D}} : W_p(\mathcal{D}, \tilde{\mathcal{D}}) \leq \epsilon} \mathbb{E}_{(x,y) \sim \tilde{\mathcal{D}}} [\ell(x, y; \theta)], \quad (4)$$

where $B_\epsilon(\cdot)$ and W_p are defined with respect to the same metric d on \mathcal{X} . Equality holds for $p = \infty$.

Hence a model achieving loss ℓ^* against DRO should achieve loss no more than ℓ^* against the usual pointwise adversary. Proposition 3.1 also gives an intuitive reason to hope for better test-set robustness with DRO: if the population \mathcal{D} lies within ϵ_1 of \mathcal{D}_n , then by the triangle inequality we can guarantee ϵ_2 -pointwise robustness on \mathcal{D} by training a DRO model on \mathcal{D}_n with $\epsilon = \epsilon_1 + \epsilon_2$.

It remains to ask: what form does the optimal worst-case distribution $\tilde{\mathcal{D}}$ take? The next result gives a clear picture: instead of independently perturbing each point within an ϵ -ball, jointly perturb all points, subject to a budget constraint on the total deviation. In particular, for W_p we constrain the ℓ_p norm of the deviations. This matches intuition that not all examples are equally susceptible to adversarial perturbation. Joint perturbation yields (up to small error) the worst-case distribution:

Lemma 3.1 (Simplified form of Gao and Kleywegt [11, Corollary 2(iv)]). *Fix θ , and suppose that for all y , the loss $\ell(\cdot, y; \theta)$ is L -Lipschitz as a function of x . Define*

$$\text{OPT} = \max_{\tilde{\mathcal{D}}: W_p(\mathcal{D}_n, \tilde{\mathcal{D}}) \leq \epsilon} \mathbb{E}_{(x,y) \sim \tilde{\mathcal{D}}}[\ell(x, y; \theta)] \quad (5)$$

and

$$\text{MIX} = \max_{\tilde{x}_1, \dots, \tilde{x}_n} \frac{1}{n} \sum_{i=1}^n \ell(\tilde{x}_i, y_i; \theta) \quad (6)$$

s.t. $\frac{1}{n} \sum_{i=1}^n d^p(\tilde{x}_i, x_i) \leq \epsilon^p$

Then we have $\text{MIX} \geq \text{OPT} - LD/n$ where D bounds the maximum deviation of a single point.

The additive error is a small technical artifact: the true worst-case distribution may need to split one point x_i into two adversarially-chosen points, but this has negligible effect on the value MIX. The term LD/n merely bounds this effect, and shrinks with the number n of samples. Note as p increases large values of $d^p(\tilde{x}, x)$ are penalized heavily and in the limit $p \rightarrow \infty$ no example moves more than ϵ .

Solving for the worst-case distribution. In the adversarial examples literature, pointwise distance d is typically taken to be an ℓ_q norm, in particular the ℓ_∞ or ℓ_2 norm. In this case, problem (6) from Lemma 3.1 can be cast as optimization over a mixed norm ball. If our training set is given as a matrix X , and we concatenate all the adversarial examples \tilde{x}_i into a matrix \tilde{X} , then we wish to solve:

$$\max_{\tilde{X}} \frac{1}{n} \sum_{i=1}^n \ell(\tilde{x}_i, y_i; \theta) \quad (7)$$

s.t. $\|\tilde{X} - X\|_{p,q} \leq n^{1/p} \epsilon$.

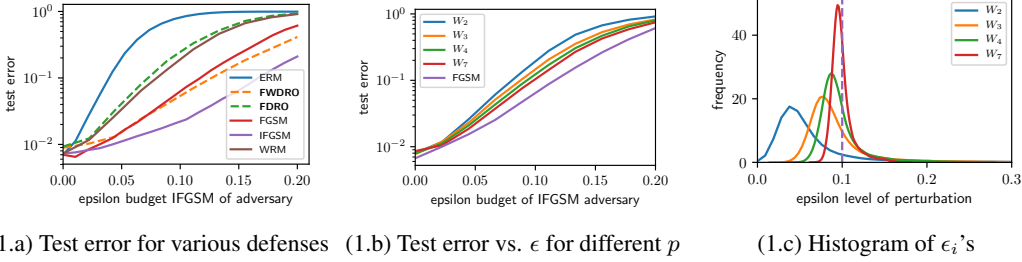
While the objective may be nonconvex, fast projection onto mixed norm balls is possible [29, 24], and we could approximately solve this via projected gradient ascent on \tilde{X} . However, these projections are hard to implement on a GPU. One alternative would be turning the constraint into a penalty:

$$\max_{\tilde{X}} \frac{1}{n} \sum_{i=1}^n \ell(\tilde{x}_i, y_i; \theta) - \lambda \|\tilde{X} - X\|_{p,q}^p \Leftrightarrow \frac{1}{n} \sum_{i=1}^n \max_{\tilde{x}_i} \{ \ell(\tilde{x}_i, y_i; \theta) - \lambda \|\tilde{x}_i - x_i\|_q^p \}. \quad (8)$$

For the special case $p = q = 2$, this is equivalent to the formulation in [28]. Conveniently, the problem decouples into separate problems for each training example, making stochastic methods simple to apply. However, there is no closed-form way to choose the parameter λ for a specific desired robustness level ϵ . The problem is unconstrained, so it is not obvious how to compute a good closed-form approximation like FGSM. Moreover, for the ubiquitous case of the ℓ_∞ norm, the regularizer’s subgradients will likely be zero in all but one coordinate. Hence k inner iterations will update only $O(k)$ coordinates of a high-dimensional example \tilde{x}_i . For small k the resulting \tilde{x}_i may be far from the true worst-case which perturbs all coordinates.

Our approach: constrained stochastic block coordinate descent. Instead, we aim to directly solve the constrained problem (7). To avoid updating the entire matrix \tilde{X} , we adopt a constrained variant of stochastic block coordinate descent (BCD). Begin with an initial \tilde{X} for which the constraint is tight. In each iteration, we sample a subset S of columns, and optimize \tilde{X}_S subject to a “local” mixed norm constraint chosen to ensure \tilde{X} is still at the constraint boundary. We find this approach works well in practice; for fixed θ our algorithm improves the objective each iteration, though we are aware of stronger guarantees only for linearly- or un-constrained BCD.

Solving the batch subproblem in each iteration is an optimization problem over an $\ell_{p,q}$ ball. Since linear functions can be optimized over mixed-norm balls in closed form (described in Appendix B), one could apply the Frank-Wolfe algorithm [10, 14], which has convergence rate guarantees even in the nonconvex case [17, 25]. If we have time only for one iteration, we can alternatively solve a linear approximation in closed form, just like FGSM.



(1.a) Test error for various defenses (1.b) Test error vs. ϵ for different p (1.c) Histogram of ϵ_i 's

Figure 1: (a) our methods (FDRO, FWDRO) are competitive with standard AT and WRM from [28]. (b), (c) confirm our theory that DRO with distance W_p generalizes pointwise adversarial training. As p grows: (b) performance of FDRO approaches FGSM; (c) the adversary's learned budgets ϵ_i for each training example x_i concentrate around the global budget $\epsilon = 0.1$ (in AT, all $\epsilon_i = 0.1$).

Training Algorithm. To train a robust model, it does not suffice to solve the adversarial problem for fixed θ : we *simultaneously* solve for model parameters θ and the worst-case distribution \tilde{X} . Noting that the model θ is typically learned via minibatch gradient methods, and that our mixed-norm algorithm above needs only minibatches of data at a time, we propose Algorithm 1: in each iteration, we sample a minibatch S , optimize the adversarial \tilde{X}_S , then use \tilde{X}_S to estimate the gradient for θ .

Note also that we never have to store \tilde{X}_S ; rather, for each training example we store a budget ϵ_i capturing how much it can be perturbed.

The end result is an algorithm that **1.** applies for any norm ℓ_q for measuring pointwise perturbation, **2.** can be approximately solved as fast as FGSM for each minibatch, **3.** requires little tuning since the constraints inform stepsizes, **4.** requires minimal extra storage and **5.** through learned ϵ_i gives insights about varying susceptibility of individual examples to perturbations. We also reiterate that the problem we solve strictly generalizes pointwise adversarial robustness (1).

Algorithm 1 Distributionally robust training

Input: dataset (X, y) containing n examples (x_i, y_i) , batch size k , budget ϵ

Initialize $\epsilon_i \leftarrow \epsilon$ for all $i = 1, \dots, n$

loop

Draw subset $S \subset \{1, \dots, n\}$ of size k

Compute local constraint $\epsilon_S = \|X_S\|_{p,q}$

Compute adversarial \tilde{X}_S solving:

$$\begin{aligned} \max_{\tilde{X}_S} \quad & \frac{1}{k} \sum_{i \in S} \ell(\tilde{x}_i, y_i; \theta) \\ \text{s.t.} \quad & \|\tilde{X}_S - X_S\|_{p,q} \leq \epsilon_S \end{aligned}$$

Update $\epsilon_i \leftarrow \|\tilde{x}_i\|_q$ for all $i \in S$

$g_\theta \leftarrow \frac{1}{k} \sum_{i=1}^k \nabla_\theta \ell(\tilde{x}_i, y_i; \theta)$

$\theta \leftarrow \theta - \gamma_\theta g_\theta$

end loop

4 Experiments

We implemented two variants of Algorithm 1 (FDRO and FWDRO) as well as the algorithm from [28] (WRM) in TensorFlow [1] by extending the Cleverhans library [22]. Fast DRO (FDRO) is the one-step closed-form version of our algorithm, while Frank-Wolfe DRO (FWDRO) runs multiple iterations per minibatch. We used the W_2 version of DRO unless otherwise stated. ERM is empirical risk minimization with no adversary. All training adversaries had initial $\epsilon_i = 0.1$. We test on MNIST using standard network architectures, algorithms and robustness parameters as described in Appendix C.

In Figure 1.a we test models trained against a variety of adversaries, by comparing their error on the test set when faced with an IFGSM adversary of varying strength. ERM predictably performs worst. Our fast one-step method is competitive with the *iterative* WRM, and our iterative FWDRO performs even better. Curiously, FGSM (one-step AT) is competitive with FWDRO, and IFGSM performs best. It is possible that the DRO algorithms do not reach optimality, or the DRO model tolerates more pointwise loss to counteract the stronger adversary.

In Figures 1.b and 1.c, we confirm our theoretical result that as p grows W_p -DRO behaves more like a pointwise adversary, both with respect to performance and the distribution of individual budgets ϵ_i .

Conclusion. We theoretically and algorithmically link Distributionally Robust Optimization and Adversarial Training, and propose a practical block coordinate descent algorithm for DRO on differentiable models. Future work includes analyzing convergence properties of the algorithm, and analyzing how exactly p and ϵ impact test-time robustness across datasets and model classes.

Acknowledgements

M. Staib acknowledges Government support under and awarded by DoD, Air Force Office of Scientific Research, National Defense Science and Engineering Graduate (NDSEG) Fellowship, 32 CFR 168a. This research was supported by The Defense Advanced Research Projects Agency (grant number N66001-17-1-4039), and an MIT *Systems That Learn* award. The views, opinions, and/or findings contained in this article are those of the author and should not be interpreted as representing the official views or policies, either expressed or implied, of the Defense Advanced Research Projects Agency or the Department of Defense.

References

- [1] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. URL <http://tensorflow.org/>. Software available from tensorflow.org.
- [2] Aharon Ben-Tal, Dick Den Hertog, Anja De Waegenare, Bertrand Melenberg, and Gijs Rennen. Robust solutions of optimization problems affected by uncertain probabilities. *Management Science*, 59(2): 341–357, 2013.
- [3] Jose Blanchet, Yang Kang, and Karthyek Murthy. Robust Wasserstein Profile Inference and Applications to Machine Learning. *arXiv:1610.05627 [math, stat]*, October 2016.
- [4] Jose Blanchet, Yang Kang, Fan Zhang, Fei He, and Zhangyi Hu. Doubly Robust Data-Driven Distributionally Robust Optimization. *arXiv:1705.07168 [stat]*, May 2017.
- [5] N. Carlini and D. Wagner. Towards Evaluating the Robustness of Neural Networks. In *2017 IEEE Symposium on Security and Privacy (SP)*, pages 39–57, May 2017. doi: 10.1109/SP.2017.49.
- [6] Moustapha Cisse, Piotr Bojanowski, Edouard Grave, Yann Dauphin, and Nicolas Usunier. Parseval Networks: Improving Robustness to Adversarial Examples. In *PMLR*, pages 854–863, July 2017.
- [7] Erick Delage and Yinyu Ye. Distributionally robust optimization under moment uncertainty with application to data-driven problems. *Operations research*, 58(3):595–612, 2010.
- [8] John Duchi, Peter Glynn, and Hongseok Namkoong. Statistics of Robust Optimization: A Generalized Empirical Likelihood Approach. *arXiv:1610.03425 [stat]*, October 2016.
- [9] Peyman Mohajerin Esfahani and Daniel Kuhn. Data-driven distributionally robust optimization using the Wasserstein metric: Performance guarantees and tractable reformulations. *Math. Program.*, pages 1–52, July 2017. ISSN 0025-5610, 1436-4646. doi: 10.1007/s10107-017-1172-1.
- [10] Marguerite Frank and Philip Wolfe. An algorithm for quadratic programming. *Naval Research Logistics*, 3 (1-2):95–110, March 1956. ISSN 1931-9193. doi: 10.1002/nav.3800030109.
- [11] Rui Gao and Anton J. Kleywegt. Distributionally Robust Stochastic Optimization with Wasserstein Distance. *arXiv:1604.02199 [math]*, April 2016.
- [12] Joel Goh and Melvyn Sim. Distributionally robust optimization and its tractable approximations. *Operations research*, 58(4-part-1):902–917, 2010.
- [13] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and Harnessing Adversarial Examples. In *ICLR*, 2015.
- [14] Martin Jaggi. Revisiting Frank-Wolfe: Projection-Free Sparse Convex Optimization. pages 427–435, 2013.
- [15] Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial examples in the physical world. In *ICLR Workshop*, 2017.
- [16] Alexey Kurakin, Ian J. Goodfellow, and Samy Bengio. Adversarial Machine Learning at Scale. In *ICLR*, 2017.

- [17] Simon Lacoste-Julien. Convergence Rate of Frank-Wolfe for Non-Convex Objectives. *arXiv:1607.00345 [cs, math, stat]*, July 2016.
- [18] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards Deep Learning Models Resistant to Adversarial Attacks. *arXiv:1706.06083 [cs, stat]*, June 2017.
- [19] Takeru Miyato, Shin-ichi Maeda, Masanori Koyama, Ken Nakae, and Shin Ishii. Distributional Smoothing with Virtual Adversarial Training. *arXiv:1507.00677 [cs, stat]*, July 2015.
- [20] Takeru Miyato, Shin-ichi Maeda, Masanori Koyama, and Shin Ishii. Virtual Adversarial Training: A Regularization Method for Supervised and Semi-supervised Learning. *arXiv:1704.03976 [cs, stat]*, April 2017.
- [21] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard. Deepfool: a simple and accurate method to fool deep neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2574–2582, 2016.
- [22] Nicolas Papernot, Ian Goodfellow, Ryan Sheatsley, Reuben Feinman, and Patrick McDaniel. Cleverhans v1.0.0: An adversarial machine learning library. *arXiv:1610.00768 [cs, stat]*, October 2016.
- [23] Nicolas Papernot, Patrick McDaniel, Xi Wu, Somesh Jha, and Ananthram Swami. Distillation as a defense to adversarial perturbations against deep neural networks. In *Security and Privacy (SP), 2016 IEEE Symposium on*, pages 582–597. IEEE, 2016.
- [24] Ariadna Quattoni, Xavier Carreras, Michael Collins, and Trevor Darrell. An Efficient Projection for L1, ∞ Regularization. In *Proceedings of the 26th Annual International Conference on Machine Learning, ICML '09*, pages 857–864, New York, NY, USA, 2009. ACM. ISBN 978-1-60558-516-1. doi: 10.1145/1553374.1553484.
- [25] S. J. Reddi, S. Sra, B. Póczos, and A. Smola. Stochastic Frank-Wolfe methods for nonconvex optimization. In *2016 54th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pages 1244–1251, September 2016. doi: 10.1109/ALLERTON.2016.7852377.
- [26] Soroosh Shafieezadeh-Abadeh, Peyman Mohajerin Esfahani, and Daniel Kuhn. Distributionally Robust Logistic Regression. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 1576–1584. Curran Associates, Inc., 2015.
- [27] Uri Shaham, Yutaro Yamada, and Sahand Negahban. Understanding Adversarial Training: Increasing Local Stability of Neural Nets through Robust Optimization. *arXiv:1511.05432 [cs, stat]*, November 2015.
- [28] A. Sinha, H. Namkoong, and J. Duchi. Certifiable Distributional Robustness with Principled Adversarial Training. *arXiv:1710.10571 [stat, cs]*, October 2017.
- [29] Suvrit Sra. Fast Projections Onto Mixed-norm Balls with Applications. *Data Min Knowl Discov*, 25(2): 358–377, September 2012. ISSN 1384-5810. doi: 10.1007/s10618-012-0277-7.
- [30] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. In *ICLR*, 2014.
- [31] Florian Tramèr, Alexey Kurakin, Nicolas Papernot, Dan Boneh, and Patrick McDaniel. Ensemble Adversarial Training: Attacks and Defenses. *arXiv:1705.07204 [cs, stat]*, May 2017.
- [32] Cédric Villani. *Optimal transport: old and new*, volume 338. Springer Science & Business Media, 2008.

A Distributionally Robust Adversaries are Stronger

Proof of Proposition 3.1. First, consider $p \in (1, \infty)$. We will show that

$$\mathbb{E}_{(x,y) \sim \mathcal{D}} \left[\max_{\tilde{x} \in B_\epsilon(x)} \ell(\tilde{x}, y; \theta) \right] \leq \max_{\hat{\mathcal{D}} : W_p(\mathcal{D}, \hat{\mathcal{D}}) \leq \epsilon} \mathbb{E}_{(x,y) \sim \hat{\mathcal{D}}} [\ell(x, y; \theta)]. \quad (9)$$

To begin, observe that the adversary which perturbs individual points can be expressed as a mapping $T : \mathcal{X} \times \mathcal{Y} \rightarrow \mathcal{X}$ given by

$$T(x, y) = \operatorname{argmax}_{\tilde{x} \in B_\epsilon(x)} \ell(\tilde{x}, y; \theta). \quad (10)$$

Moreover, applying $(x, y) \mapsto (T(x, y), y)$ to \mathcal{D} implicitly defines a particular distribution $\hat{\mathcal{D}}$ with the property that

$$\mathbb{E}_{(x,y) \sim \mathcal{D}} \left[\max_{\tilde{x} \in B_\epsilon(x)} \ell(\tilde{x}, y; \theta) \right] = \mathbb{E}_{(x,y) \sim \mathcal{D}} [\ell(T(x, y), y; \theta)] \quad (11)$$

$$= \mathbb{E}_{(x,y) \sim \hat{\mathcal{D}}} [\ell(x, y; \theta)]. \quad (12)$$

It remains to show that $\hat{\mathcal{D}}$ is a feasible solution for the DRO adversary, i.e. that $W_p(\mathcal{D}, \hat{\mathcal{D}}) \leq \epsilon$. This can be done by observing that T is a feasible transport map between \mathcal{D} and $\hat{\mathcal{D}}$. Wasserstein distance is defined as an infimum over probabilistic transport *plans*, and a transport *map* is just a special case:

$$W_p^p(\mathcal{D}, \hat{\mathcal{D}}) = \inf_{\pi \in \Pi(\mathcal{D}, \hat{\mathcal{D}})} \int d^p(x, \hat{x}) d\pi((x, y), (\hat{x}, \hat{y})) \quad (13)$$

$$\leq \mathbb{E}_{(x,y) \sim \mathcal{D}} [d^p(x, T(x, y))] \quad (14)$$

$$\leq \mathbb{E}_{(x,y) \sim \mathcal{D}} [\epsilon^p] = \epsilon^p, \quad (15)$$

from which it follows that $W_p(\mathcal{D}, \hat{\mathcal{D}}) \leq \epsilon$.

For the $p = \infty$ case, simply note that between \mathcal{D} and $\hat{\mathcal{D}}$, no point must move more than ϵ from x to \tilde{x} . Hence the essential supremum of the cost incurred by T is bounded by ϵ . Tightness follows because, under the constraint $W_\infty(\mathcal{D}, \hat{\mathcal{D}})$, no point is allowed to move more than ϵ anyway, and there is no reason to split mass. \square

B Linear Optimization in the $\ell_{p,q}$ ball

We seek to solve problems of the form

$$\begin{aligned} \max_X \quad & \langle C, X \rangle \\ \text{s.t.} \quad & \|X\|_{p,q} \leq 1. \end{aligned} \quad (16)$$

The result for general constraint values is the same but scaled linearly.

Let x_i be the columns of X . The problem above can be rewritten as two nested problems:

$$\max_{\epsilon: \|\epsilon\|_p \leq 1} \max_{X: \|x_i\|_q \leq \epsilon_i \forall i} \langle C, X \rangle, \quad (17)$$

where $\epsilon = (\epsilon_1, \dots, \epsilon_n)$. The inner optimization can be separated out for each i :

$$\max_{\epsilon: \|\epsilon\|_p \leq 1} \sum_{i=1}^n \max_{x_i: \|x_i\|_q \leq \epsilon_i} \langle c_i, x_i \rangle. \quad (18)$$

We now need a quick lemma, which is proved just by checking optimality conditions:

Lemma B.1. *Let $p \in (1, \infty)$. The optimal solution x^* to the problem*

$$\max_{x: \|x\|_p \leq \gamma} \langle c, x \rangle \quad (19)$$

satisfies $x_j^ \propto c_j^{1/(p-1)}$, with normalization to ensure the constraint is satisfied. For $p = \infty$, we have $x_j^* \propto \operatorname{sign}(c_j)$. Note this means that given only p and c , we can compute v with $\|v\|_p = 1$ so that $x^* = \gamma v$.*

Hence, in problem (18) we can compute a corresponding v_i for each x_i so that the problem is equivalent to

$$\max_{\epsilon: \|\epsilon\|_p \leq 1} \sum_{i=1}^n \langle c_i, \epsilon_i v_i \rangle = \max_{\epsilon: \|\epsilon\|_p \leq 1} \sum_{i=1}^n \epsilon_i \langle c_i, v_i \rangle. \quad (20)$$

Write $a_i = \langle c_i, v_i \rangle$. This is equivalent to $\max_{\epsilon: \|\epsilon\|_p \leq 1} \langle a, \epsilon \rangle$, which can also be solved in closed form by the lemma, and once ϵ is determined, we just set $x_i = \epsilon_i v_i$. More compactly, to solve problem (18) we:

Table 1: Parameters for experiments in the main text.

Figure	defense	q	p	ϵ	γ	batch size	epochs	optimizer	nonlinearity
1.a	ERM	N/A	N/A	N/A	N/A	128	25	SGD	ReLU
1.a	FWDRO	∞	2	0.1	N/A	128	25	SGD	ReLU
1.a	FDRO	∞	2	0.1	N/A	512	25	SGD	ReLU
1.a	FGSM	∞	2	0.1	N/A	128	25	SGD	ReLU
1.a	IFGSM	∞	2	0.1	N/A	128	25	SGD	ReLU
1.a	WRM	2	2	N/A	0.001	128	25	Adam	ELU
1.b	FDRO	∞	2	0.1	N/A	512	10	SGD	ReLU
1.b	FDRO	∞	3	0.1	N/A	512	10	SGD	ReLU
1.b	FDRO	∞	4	0.1	N/A	512	10	SGD	ReLU
1.b	FDRO	∞	7	0.1	N/A	512	10	SGD	ReLU
1.b	FGSM	∞	2	0.1	N/A	256	10	SGD	ReLU
1.c	FDRO	∞	2	0.1	N/A	512	10	SGD	ReLU
1.c	FDRO	∞	3	0.1	N/A	512	10	SGD	ReLU
1.c	FDRO	∞	4	0.1	N/A	512	10	SGD	ReLU
1.c	FDRO	∞	7	0.1	N/A	512	10	SGD	ReLU

- From c_i and q , compute v_i per the lemma.
- Compute $a_i = \langle c_i, v_i \rangle$.
- Solve $\max_{\epsilon: \|\epsilon\|_p \leq 1} \langle a, \epsilon \rangle$ via the lemma.
- Finally set each column $x_i = \epsilon_i v_i$.

C Experiment Details

Parameters for all experiments in the main text are given in Table 1. A stepsize of 0.5 was used for SGD, and stepsize 0.001 for Adam. In all cases, q refers to the ℓ_q norm for comparing points, p refers to the W_p distance between distributions, ϵ is as in Equation (3), γ is the regularization parameter from [28].

SGD was used as the default optimizer for our algorithms as its simplicity helped stability: from the perspective of the model learner, the loss function is actually changing as the distributional adversary evolves.

For WRM, $\gamma = 0.001$ was chosen from $\{0.0001, 0.0003, 0.001, 0.003, 0.01, 0.03, 0.1, 0.3, 1\}$ as the best performer. ELU was used instead of ReLU as recommended in [28], and we switched from SGD to Adam here to help with stability. We used $p = 2$ for WRM as in [28], despite the mismatch with the IFGSM adversary. However, 1. we also tested WRM with $p = \infty$ and it performed poorly, and 2. we found that switching to an ℓ_2 IFGSM adversary did not substantially change Figure 1.a.

All iterative defenses and test-time adversaries were run for 15 iterations.

The network architecture was the same across all experiments except for the WRM experiments which swapped ReLU for ELU. We used the standard CNN from the Cleverhans tutorials: 8×8 filters, then 6×6 filters, then 5×5 filters, followed by a fully-connected layer and then softmax.

D Additional Experiments

In Figure 2 we repeat the experiment from Figure 1.a, except this time we test against IFGSM with respect to ℓ_2 norm. The same performance trend emerges, despite the reversed mismatch between pointwise norms. Specifically, though the model trained against WRM is the only one trained to be robust to ℓ_2 error and all others are targeted for ℓ_∞ , IFGSM performs best, followed by FGSM and our FWDRO, and WRM lags behind. Note that for WRM, $\gamma = 0.001$ was again selected as the best performing value for this test.

In Figure 3 we evaluate the stability of the FDRO training procedure as a function of batch size. Training is more stable when batch sizes are sufficiently large. Our intuition is that our BCD algorithm needs to assign a value ϵ_i to each training sample x_i ; the more ϵ_i 's it can update at once, the faster it will find an optimal assignment over the whole training set.

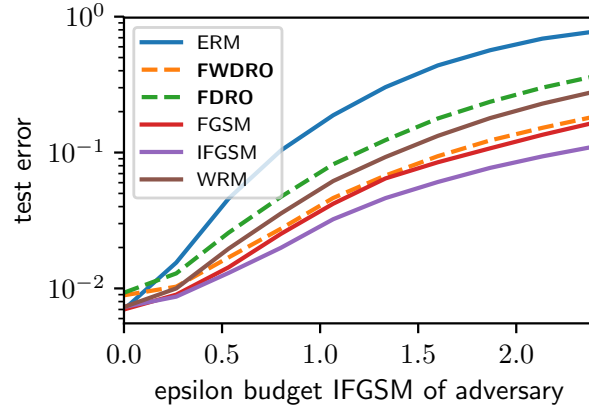


Figure 2: Test error for various defenses against ℓ_2 IFGSM adversary. Same as Figure 1.a but with an ℓ_2 adversary instead of ℓ_∞ .

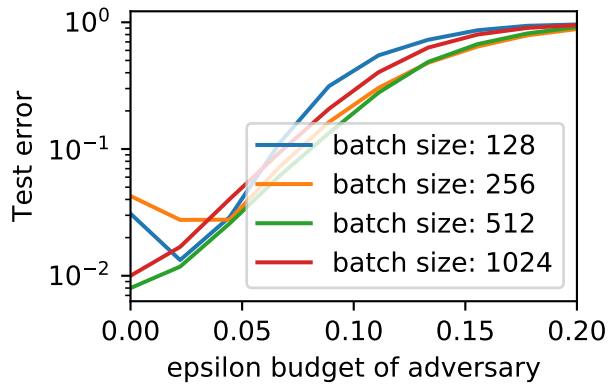


Figure 3: Robustness to IFGSM of FDRO for W_2 , trained with different batch sizes. FDRO training is less stable for smaller batch sizes, resulting in poor accuracy at test time on clean examples. As batch size increases, stability improves considerably, presumably because each block coordinate step is a better approximation of a full gradient step.