ADVERSARIAL TRAINING WITH VORONOI CONSTRAINTS

Anonymous authors
Paper under double-blind review

ABSTRACT

Adversarial examples are a pervasive phenomenon of machine learning models where seemingly imperceptible perturbations to the input lead to misclassifications for otherwise statistically accurate models. Adversarial training, one of the most successful empirical defenses to adversarial examples, refers to training on adversarial examples generated within a geometric constraint set. The most commonly used geometric constraint is an L_p-ball of radius ϵ in some norm. We introduce adversarial training with Voronoi constraints, which replaces the L_p-ball constraint with the Voronoi cell for each point in the training set. We show that adversarial training with Voronoi constraints produces robust models which significantly improve over the state-of-the-art on MNIST and are competitive on CIFAR-10.

1 INTRODUCTION

Deep learning at scale has led to breakthroughs on important problems in computer vision (Krizhevsky et al. (2012)), natural language processing (Wu et al. (2016)), and robotics (Levine et al. (2015)). Shortly thereafter, the interesting phenomena of *adversarial examples* was observed. A seemingly ubiquitous property of machine learning models where perturbations of the input that are imperceptible to humans reliably lead to confident incorrect classifications (Szegedy et al. (2013); Goodfellow et al. (2014)). What has ensued is a standard story from the security literature: a game of cat and mouse where defenses are proposed only to be quickly defeated by stronger attacks (Athalye et al. (2018)). This has led researchers to develop methods which are provably robust under specific attack models (Wong and Kolter (2018); Sinha et al. (2018); Raghunathan et al. (2018); Mirman et al. (2018)) as well as empirically strong heuristics (Madry et al. (2018); Zhang et al. (2019)). As machine learning proliferates into society, including security-critical settings like health care Esteva et al. (2017) or autonomous vehicles Codevilla et al. (2018), it is crucial to develop methods that allow us to understand the vulnerability of our models and design appropriate counter-measures.

Adversarial training has been one of the few heuristic methods which has not been defeated by stronger attacks. In this paper, we propose a modification to the standard paradigm of adversarial training. We replace the L_p-ball constraint with the Voronoi cells of the training data, which have several advantages detailed in Section 3. In particular, we need not set the maximum perturbation size ϵ as part of the training procedure. The Voronoi cells adapt to the maximum allowable perturbation size locally on the data distribution. We show how to construct adversarial examples within the Voronoi cells and how to incorporate Voronoi constraints into standard adversarial training. In Section 5 we show that adversarial training with Voronoi constraints gives state-of-the-art robustness results on MNIST and competitive results on CIFAR-10.

2 RELATED WORK

Adversarial training, the process of training on adversarial examples generated in L_p-balls around the training data, is a very natural approach to constructing robust models and was originally proposed by Goodfellow et al. (2014). Madry et al. (2018) formalized the adversarial training objective and highlighted the importance of a strong adversary for constructing adversarial examples in the inner training loop. Their approach to adversarial training, which utilized a projected gradient descent adversary, produced some of the first empirically robust models which were not later broken

by stronger attacks. There's was the *only* approach surveyed by Athalye et al. (2018) which was not either fully circumvented by Athalye et al. (2018) or in a later paper (Jalal et al. (2019)). More recently, the celebrated algorithm TRADES (Zhang et al. (2019)) has been proposed, which attempts to provide a principled way to trade off between robustness and natural accuracy. The analysis that inspires TRADES decomposes the robust error into two terms: natural error and error near the decision boundary. The yields an objective function with two terms, one which encourages accuracy and another which pushes the decision boundary away from the data distribution. Constructing a decision boundary that is far from the data distribution is explored in other heuristic works such as Ding et al. (2018); Jakubovitz and Giryes (2018); Hoffman et al. (2019). Our approach falls into this class of defenses and so we will compare exclusively against such defenses.

The frequency with which heuristic defenses have been defeated by stronger attacks has led to a line of work on certifiable robustness, which can guarantee that there exists no perturbation within an L_p-ball of radius ϵ which causes the classifier to change its classification. One of the first works by Wong and Kolter (2018) proposed to approximate the set of possible activations of every L_∞-bounded perturbation by propagating upper and lower bounds for each activation through the network. These upper and lower bounds are used to construct a convex outer approximation to the set of possible activations in the final layer, and a linear program is used to certify that this convex approximation does not intersect the decision boundary. This initial work had several notable drawbacks, and several subsequent works have attempted to improve upon these initial results (Weng et al. (2018); Mirman et al. (2018); Gehr et al. (2018); Wong et al. (2018); Singh et al. (2019)). However the fundamental problems have remained: (1) these approaches do not scale to larger networks despite considerable effort, (2) they often depend crucially on the specific details of the architecture, and (3) the size of ϵ which can be certified is often considerably smaller than what we observe to be empirically robust. A different approach to certified robustness which addresses some of these concerns, called randomized smoothing (Lecuyer et al. (2018); Cohen et al. (2019)), has recently been proposed. Randomized smoothing leverages the ability of *any* classifier f to perform well on Gaussian noise to construct a new classifier g which is certifiably robust under adversarial L_2 perturbations. Unlike prior approaches to certified robustness, randomized smoothing is a simple approach which does not depend on the architecture details of the classifier. Its main drawback is that it is currently, and possibly fundamentally, limited to L_2. We also note that more recent work has combined randomized smoothing with adversarial training to produce even more certifiably robustness classifiers in L_2 (Salman et al. (2019)). Since the goal and limitations of these method are often different from heuristic approaches we do not compare our method against these approaches.

Finally there has been a long line of work on the theory of adversarial examples. Schmidt et al. (2018) explore the sample complexity required to produce robust models. They demonstrate a simple setting, a mixture of two Gaussians, in which a linear classifier with near perfect natural accuracy can be learned from a single sample, but *any* algorithm that produces *any* binary classifier requires $\Omega(\sqrt{d})$ samples to produce a robust classifier. Followup work by Bubeck et al. (2019) suggests that adversarial examples may arise from computational constraints. They exhibit pairs of distributions that differ only in a k-dimensional subspace, and are otherwise standard Gaussians, and show that while it is information-theoretically possible to distinguish these distributions, it requires exponentially many queries in the statistical query model of computation. We note that both of these constructions produce distributions whose support is the entirety of $\mathbb{R}^d$.

Additionally there is a line work that attempts to explain the pervasiveness of adversarial examples through the lens of high-dimensional geometry. The work of Gilmer et al. (2018) experimentally evaluated the setting of two concentric under-sampled 499-spheres embedded in $\mathbb{R}^{500}$, and concluded that adversarial examples occur on the data manifold. Shafahi et al. (2019) suggest that adversarial examples may be an unavoidable consequence of the high-dimensional geometry of data. Their result depends upon the use of an isopermetric inequality. The main drawback of these works, as well as the constructions in the previous paragraph, is that they assume that the support of the data distribution has full or nearly full dimension. We do not believe this to be the case in practice, instead we believe that the data distribution is often supported on a very low-dimensional subset of $\mathbb{R}^d$. This case is addressed in Khoury and Hadfield-Menell (2018), where they consider the problem of adversarial robustness in the case where data is drawn from a low-dimensional manifold embedded in $\mathbb{R}^d$. They highlight the role of co-dimension, the difference between the dimension of the embedding space and the dimension of the data manifold, as a key source of the pervasiveness of adversarial vulnerability. Said differently, it is the low-dimensional structure of features embed-

ded in high-dimensional space that contributes, at least in part, to adversarial examples. This idea is also explored in Nar et al. (2019), but with emphasis on the cross-entropy loss. We build on the work of Khoury and Hadfield-Menell (2018), specifically their results on adversarial training in high-codimensions, which make clear several drawbacks of the L_p-ball formulation. In Section 5.1 we show that our approach improves robustness in high-codimension settings.

3 Adversarial Examples from Voronoi Cells

Goodfellow et al. (2014) originally proposed adversarial training where adversarial examples were constructed inside of an L_p-ball of radius ϵ. The use of the L_p-ball was meant to represent a simple notion of similarity between two images, delaying the complicated question of what is an adversarial image in favor of a tractable research problem. However it was never meant to be the final say on the threat model of the adversary and recent work has begun to explore alternative adversaries (Kang et al. (2019); Hendrycks et al. (2019)).

Khoury and Hadfield-Menell (2018) describe a number of issues associated with the use of L_p-balls. Their results are formalized in the manifold setting, where samples from each class are sampled from one of C *class manifolds* $\mathcal{M}_1, \ldots, \mathcal{M}_C$, and the *data manifold* $\mathcal{M} = \cup_{1 \leq j \leq C} \mathcal{M}_j$ is a k-dimensional manifold embedded in $\mathbb{R}^d$. When $p = 2$, they show that the L_2-balls centered on a dense sample of $\mathcal{M}$ covers a negligible fraction of the neighborhood around $\mathcal{M}$. Thus, when constructing adversarial examples in the inner training loop, the adversary is restricted to constructing adversarial examples in a negligible fraction of the neighborhood around the data manifold. This vulnerability increases with the *codimension* $d - k$ of $\mathcal{M}$. Furthermore they show that, for any p, a nearest neighbor classifier more effectively covers the neighborhood around $\mathcal{M}$ than a robust empirical risk minimization oracle, which outputs a classifier that is guaranteed to be correct in the L_p-balls centered on the data.

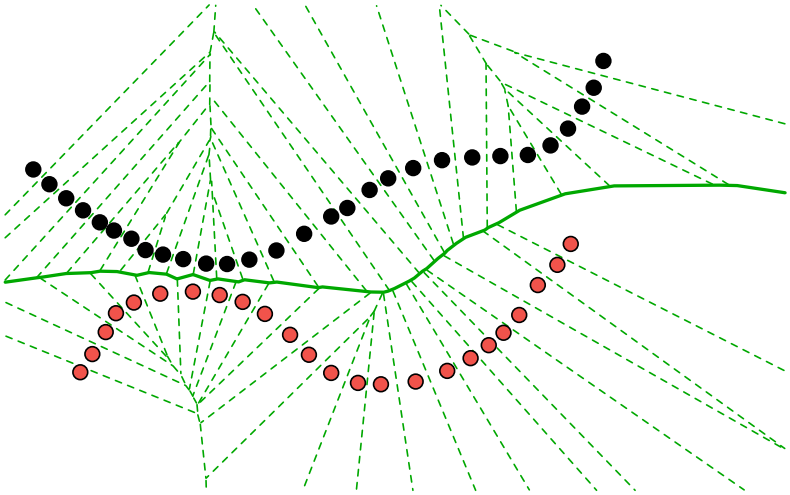

Figure 1: The Voronoi diagram for a dense sample drawn from a low-dimensional distribution with two classes, one in red and one in black. The Voronoi cells, shown in green, vary in size depending on how close a sample is to samples in the other class. The Voronoi edges that are adjacent to two samples from two different classes are shown in solid green, and approach a decision boundary which is as far from the data distribution as possible.

To remedy these shortcomings, we replace the L_p-ball constraint with a different geometric constraint, namely the Voronoi cell at each sample x, defined as

$$\mathrm{Vor}_p\, x = \{x' \in \mathbb{R}^d : \|x - x'\|_p \leq \|z - x'\|_p \; \forall z \in X \backslash \{x\}\}. \tag{1}$$

In words, the Voronoi cell $\mathrm{Vor}_p\, x$ of x is the set of all points in $\mathbb{R}^d$ that are closer to x than to any other sample in X. The Voronoi diagram is defined as the collection of Voronoi cells, and their

lower dimensional faces, for each sample in X. Figure 1 shows the Voronoi diagram for a dense sample from a dataset with two classes of data.

The Voronoi cell constraint has many advantages over the L_p-ball constraint. First the Voronoi cells *partition* the entirety of $\mathbb{R}^d$ and so the interiors of Voronoi cells generated by samples from different classes do not intersect. This is in contrast to L_p-balls which may intersect for sufficiently large ϵ. In particular the Voronoi cells partition the neighborhood around $\mathcal{M}$ and, for dense samples, are elongated in the directions normal to the data manifold (Dey (2007)). Thus the Voronoi cells are well suited for high codimension settings. Second, the size of the Voronoi cells adapts to the data distribution. A Voronoi cell generated by a sample which is close to samples from a different class manifold is smaller, while those further away are larger. See Figure 1. Thus we do *not* need to set a value for ϵ in the optimization procedure. The constraint naturally adapts to the largest value of ϵ possible locally on the data manifold. Note that the maximum perturbation size possible will often vary as we move along the data manifold, and cannot be captured by a single number which, by necessity, is upper bounded by the smallest distance to a different class. In summary, the Voronoi constraint gives the adversary the freedom to explore the entirety of the neighborhood around $\mathcal{M}$.

At each iteration of standard adversarial training, we must solve the inner optimization problem $\max_{\delta \in B(0,\epsilon)} L(x+\delta, y; \theta)$ to generate an adversarial example. Goodfellow et al. (2014) solve this problem using the fast gradient sign method (FGSM), while Madry et al. (2018) use projected gradient descent. To incorporate Voronoi constraints, at each iteration of the outer training loop we must solve the inner optimization problem

$$\begin{aligned} \underset{\hat{x}}{\text{maximize}} \quad & L(\hat{x}, y; \theta) \\ \text{subject to} \quad & \|x - \hat{x}\|_p - \|z - \hat{x}\|_p \leq 0 \; \forall z \in X - \{x\}. \end{aligned} \tag{2}$$

When $p = 2$ the Voronoi cells are convex and so we can project a point onto a Voronoi cell by solving a quadratic program. Thus we can solve Problem 2 using projected gradient descent, as in Madry et al. (2018). When $p \neq 2$ the Voronoi cells are not necessarily convex. In this setting there are many approaches, such as barrier and penalty methods, one might employ to approximately solve Problem 2 (Boyd and Vandenberghe (2004)).

However we found that the following heuristic is both fast and works well in practice. At each iteration of the outer training loop, for each training sample x in a batch, we generate adversarial examples by taking iterative steps in the direction of the gradient starting from x. Instead of projecting onto a constraint after each iterative step, we instead check if any of the Voronoi constraints of x shown in Equation 1 are violated. If no constraint is violated we perform the iterative update, otherwise we simply stop performing updates for x. Figure 2 illustrates the procedure.

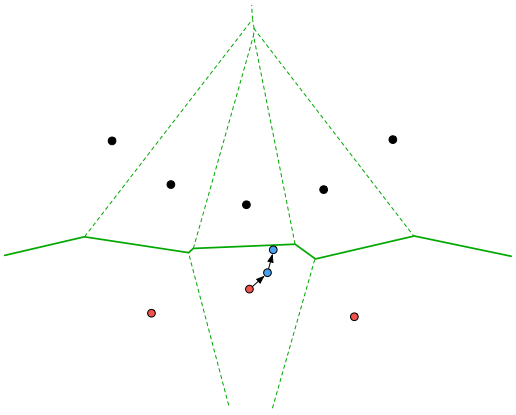

Figure 2: To construct an adversarial example within a Voronoi cell, we repeatedly take steps in the direction of the gradient of the loss, shown in blue. After each iteration we check if any of the Voronoi constraints are violated. We take the last iteration before a constraint is violated as our adversarial example.

Problem 2 has $n-1$ constraints, one for each sample in $X \backslash \{x\}$. In practice however very few samples contribute to the Voronoi cell of x. Even fewer contribute to the faces of the Voronoi cell that are shared by samples in different classes, as shown in Figure 1. At each iteration, we perform a nearest neighbor search query to find the m nearest samples to x in each other class. That is we search for $m(C-1)$ samples where C is the number of classes. We do not impose constraints from samples in the same class as x; there is no benefit to restricting the adversary's movement with the neighborhood around the class manifold of x. In our experiments we set $m = 10$.

4 Adversarial Training with Voronoi Constraints

Madry et al. (2018) formalize adversarial training by introducing the robust objective

$$\min_{\theta} \mathbb{E}_{(x,y)\in\mathcal{D}} \left[\max_{\hat{x}\in B(x,\epsilon)} L(\hat{x}, y; \theta) \right] \tag{3}$$

where $\mathcal{D}$ is the data distribution and B is a L_p-ball centered at x with radius ϵ. Their main contribution was the use of a strong adversary which used projected gradient descent to solve the inner optimization problem.

To incorporate Voronoi constraints, we replace the L_p-ball constraint in Equation 3 with the Voronoi cell at x. That is, we formalize the adversarial training objective as

$$\min_{\theta} \mathbb{E}_{(x,y)\in\mathcal{D}} \left[\max_{\hat{x}\in \mathrm{Vor}_p\, x} L(\hat{x}, y; \theta) \right], \tag{4}$$

where we use the optimization procedure described in Section 3 to solve the inner optimization problem.

5 Experiments

Datasets. Khoury and Hadfield-Menell (2018) introduce a synthetic dataset, PLANES, to investigate how the codimension (low-rank features) of a dataset influences robustness. The PLANES dataset consists of two 2-dimensional planes, the first in the $x_d = 0$ and the second in $x_d = 2$. The first two axis of both planes are bounded as $-10 \leq x_1, x_2 \leq 10$, while $x_3 = \ldots = x_{d-1} = 0$. The training set is sampled at the vertices of a regular grid with side length $\sqrt{2}$, and the test set at the centers of the grid cubes. This sampling is chosen so that the L_2-balls of radius 1 cover the 2-dimensional planes, and so a classifier that does well inside these balls also has perfect natural accuracy. The spacing along the axis x_d is chosen so the maximum perturbation size is 1. The codimension of this dataset is $d-2$. We also evaluate on MNIST and CIFAR-10.

Models. Our controlled experiments on synthetic data consider a fully connected network with 1 hidden layer, 100 hidden units, and ReLU activations. We set the learning rate for Adam (Kingma and Ba (2015)) as $\alpha = 0.1$. Our experimental results are averaged over 20 retrainings. For a fair comparison to adversarial training, our experiments on MNIST and CIFAR-10 use the same model architectures as in Madry et al. (2018). We train the MNIST model using Adam for 100 epochs and the CIFAR-10 model using SGD for 250 epochs.

Attacks. On MNIST we apply 300-step projected gradient descent (PGD), with step sizes $\{0.05, 0.07, 0.1, 0.15, 0.17, 0.2\}$. On CIFAR-10 we apply 20-step PGD with step sizes $\{2.0, 3.0, 4.0\}$. For both datasets we also apply the fast gradient sign method (FGSM) Goodfellow et al. (2014) to uncover possible gradient masking as recommended in Athalye et al. (2018). We evaluate these attacks *per sample*, meaning that if any attack successfully constructs an adversarial example for a sample x at a specific ϵ, it reduces the robust accuracy of the model at that ϵ.

Accuracy measures. We plot the robust classification accuracy as a function of ϵ, for each of our datasets. Since one of the primary advantages of Voronoi constraints is that we do not need to set ϵ, we need a measure of robustness that considers the total robustness of the model. Thus we report the *normalized area under the curve* (NAUC) defined as

$$\mathrm{NAUC}(\mathrm{acc}) = \frac{1}{\epsilon_{\max}} \int_0^{\epsilon_{\max}} \mathrm{acc}(\epsilon)\, d\epsilon, \tag{5}$$

where $\text{acc} : [0, \epsilon_{\max}] \to [0, 1]$ measures the classification accuracy and $\epsilon_{\max}$ is the largest perturbation considered. Note that NAUC $\in [0, 1]$ with higher values corresponding to more robust models.

Implementation Details. Constructing adversarial examples within the Voronoi cells, as described in Section 3, requires a nearest neighbor search query to find the m nearest samples to x in each other class. When the dataset remains constant throughout the course of training, this search can be performed once before training begins and reused at each iteration. However when the dataset is augmented during training, as in the case of data augmentation on CIFAR-10, the nearest neighbor search query must be computed at each iteration. Since this computation is performed on the CPU, we create 16 threads, each with a copy of a k-d tree, which constantly pull mini-batches of samples from a queue and perform nearest neighbor queries. With 16 threads running in parallel, the bottleneck for training became the construction of adversarial examples on the GPU, and so adversarial training with Voronoi constraints ran in time similar to standard adversarial training.

5.1 Adversarial Training in High Codimensions

Khoury and Hadfield-Menell (2018) showed that as the codimension of the Planes dataset increases, the adversarial training approach of Madry et al. (2018) with training $\epsilon = 1$ became less robust. They suggested that this was because the L_2-balls with radius 1 around the dataset covered an increasingly smaller fraction of the neighborhood around the data manifold.

Figure 3 shows that replacing the L_2 ball constraint with the Voronoi cells improves robustness in high codimension settings, on average. In codimension 10 (Figure 3 (Left)), our approach achieves NAUC of 0.99, while Madry's approach achieves NAUC of 0.94. In codimension 500 (Figure 3 (Right)), our approach achieves NAUC of 0.92, while Madry's approach achieves NAUC of 0.87.

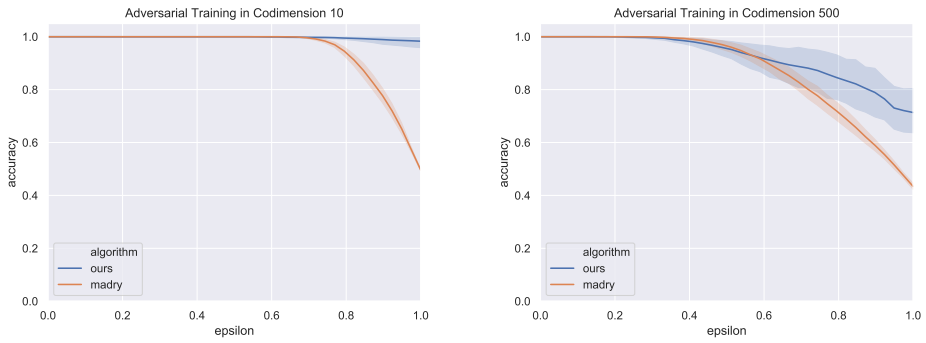

Figure 3: Adversarial training Voronoi constraints offers improved robustness in high codimension (10, 500) over standard adversarial training, on average.

5.2 MNIST and CIFAR-10

To explore the performances of adversarial training with Voronoi constraints on more realistic datasets, we evaluate on MNIST and CIFAR-10 and compare against the robust pretrained models of Madry et al. (2018).[1][2]. We include the recently proposed Jacobian regularization algorithm of Hoffman et al. (2019) with $\lambda_{\text{jr}} = 1.0$ as an additional baseline.

Figure 4 (Left) shows that our model maintains near identical robustness to the Madry model on MNIST up to $\epsilon = 0.3$, after which our model *significantly* outperforms the Madry model. The Madry model was explicitly trained for $\epsilon = 0.3$ perturbations. We emphasize that one advantage of our approach is that we did not need to set a value for the maximum perturbation size ϵ. The Voronoi cells adapt to the maximum size allowable locally on the data distribution. Our model maintains 76.3% accuracy at $\epsilon = 0.4$ compared to 2.6% accuracy for the Madry model. Furthermore our

[1] https://github.com/MadryLab/mnist_challenge
[2] https://github.com/MadryLab/cifar10_challenge

model achieves NAUC of 0.81, while the Madry model achieves NAUC of 0.67, an improvement of 20.8% and over the baseline. To our knowledge, this is the most robust MNIST model to L_∞ attacks.

Figure 4 (Right) shows the results of our approach on CIFAR-10. Both our model and the Madry model achieve NAUC of 0.29. However our approach trades natural accuracy for increased robustness against larger perturbations. This tradeoff is well-known and explored in Tsipras et al. (2019); Ilyas et al. (2019).

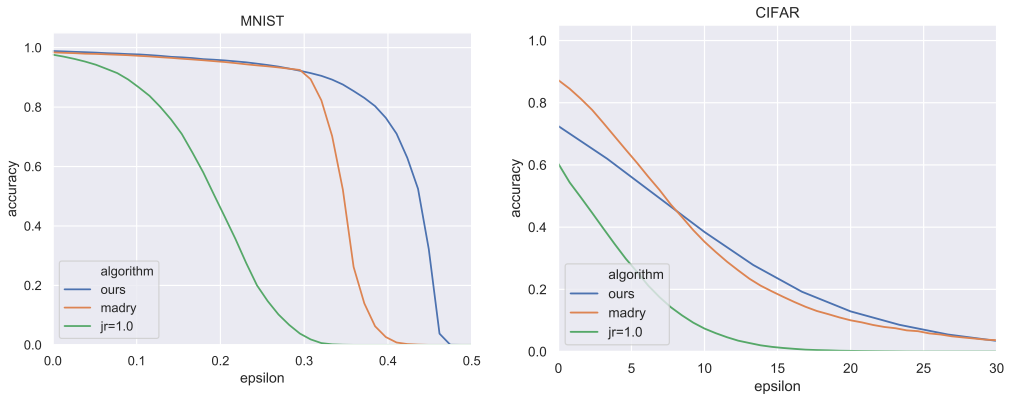

Figure 4: **Left:** Adversarial training with Voronoi constraints on MNIST. Our model has NAUC 0.81 and high classification accuracy after $\epsilon = 0.3$. In particular, our model maintains 76.3% accuracy at $\epsilon = 0.4$, compared to 2.6% accuracy for the Madry model. **Right:** On CIFAR-10, both models achieve NAUC of 0.29, but our model trades natural accuracy for robustness to larger perturbations.

5.3 Increasing the Radius of the Norm Ball Constraint

A natural approach to improving the robustness of models produced by the adversarial training paradigm of Madry et al. (2018) is to simply increase the maximum allowable perturbation size ϵ of the norm ball constraint. As shown in Figure 5, increasing the size of ϵ to 0.4, from the 0.3 with which Madry et al. (2018) originally trained, and training for only 100 epochs produces a model which exhibits significantly worse robustness in the range $[0, 0.3]$ than the pretrained model. If we increase the number of training epochs to 150, the approach of Madry et al. (2018) with $\epsilon = 0.4$ produces a model with improved robustness in the range $[0.3, 0.4]$, but that still exhibits the sharp drop in accuracy after 0.4. Additionally the model trained with $\epsilon = 0.4$ for 150 epochs performs worse than both the pretrained model and our model in the range $[0, 0.3]$. Our model achieves NAUC 0.81, while the model trained with $\epsilon = 0.4$ for 150 epochs achieves NAUC 0.76. We emphasize that our approach does not require us to set ϵ, which is particularly important in practice where the maximum amount of robustness achievable may not be known a-priori.

6 Conclusions

The L_p-ball constraint for describing adversarial perturbations has been a productive formalization for designing robust deep networks. However, the use of L_p-balls has significant drawbacks in high-codimension settings and leads to sub-optimal results in practice. Adversarial training with Voronoi constraints improves robustness by giving the adversary the freedom to explore the neighborhood around the data distribution.

References

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

S. Boyd and L. Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

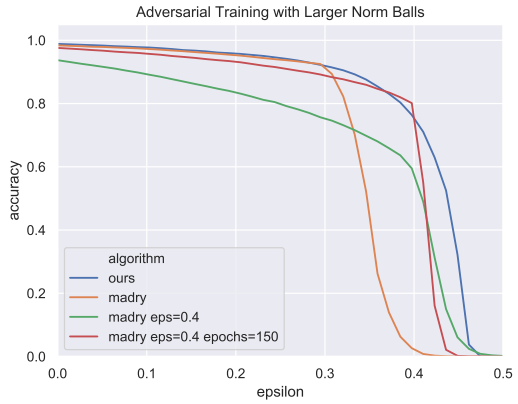

Figure 5: The adversarial training of Madry et al. (2018) with $\epsilon = 0.4$ (shown in green) produces a model with significantly reduced robustness in the range $[0, 0.3]$. Increasing the number of epochs to 150, the resulting model (shown in red) does exhibit improved robustness in the range $[0.3, 0.4]$, at the expense of some robustness in the range $[0, 0.3]$ and still exhibits a sharp drop in accuracy after 0.4. The purple model achieves NAUC of 0.76, while our model achieves NAUC 0.81.

S. Bubeck, Y. T. Lee, E. Price, and I. P. Razenshteyn. Adversarial examples from computational constraints. In *ICML*, 2019.

F. Codevilla, M. Müller, A. Dosovitskiy, A. López, and V. Koltun. End-to-end driving via conditional imitation learning. In *ICRA*, 2018.

J. M. Cohen, E. Rosenfeld, and J. Z. Kolter. Certified adversarial robustness via randomized smoothing. In *ICML*, 2019.

T. K. Dey. *Curve and Surface Reconstruction: Algorithms with Mathematical Analysis*. Cambridge University Press, 2007.

G. W. Ding, Y. Sharma, K. Y. C. Lui, and R. Huang. Max-margin adversarial (MMA) training: Direct input space margin maximization through adversarial training. *CoRR*, abs/1812.02637, 2018.

A. Esteva, B. Kuprel, R. A. Novoa, J. Ko, S. M. Swetter, H. M. Blau, and S. Thrun. Dermatologist-level classification of skin cancer with deep neural networks. *Nature*, 2017.

T. Gehr, M. Mirman, D. Drachsler-Cohen, P. Tsankov, S. Chaudhuri, and M. Vechev. AI2: Safety and robustness certification of neural networks with abstract interpretation. In *IEEE Symposium on Security and Privacy*, 2018.

J. Gilmer, L. Metz, F. Faghri, S. S. Schoenholz, M. Raghu, M. Wattenberg, and I. J. Goodfellow. Adversarial spheres. *CoRR*, abs/1801.02774, 2018. URL `http://arxiv.org/abs/1801.02774`.

I. J. Goodfellow, J. Shlens, and C. Szegedy. Explaining and harnessing adversarial examples. In *ICLR*, 2014.

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

J. Hoffman, D. A. Roberts, and S. Yaida. Robust learning with jacobian regularization. *CoRR*, abs/1908.02729, 2019.

A. Ilyas, S. Santurkar, D. Tsipras, L. Engstrom, B. Tran, and A. Madry. Adversarial examples are not bugs, they are features. *CoRR*, abs/1905.02175, 2019.

D. Jakubovitz and R. Giryes. Improving DNN robustness to adversarial attacks using jacobian regularization. In *ECCV*, 2018.

A. Jalal, A. Ilyas, C. Daskalakis, and A. G. Dimakis. The robust manifold defense: Adversarial training using generative models. *CoRR*, abs/1712.09196, 2019.

D. Kang, Y. Sun, D. Hendrycks, T. Brown, and J. Steinhardt. Testing robustness against unforeseen adversaries. *arXiv preprint arXiv:1908.08016*, 2019.

M. Khoury and D. Hadfield-Menell. On the geometry of adversarial examples. *CoRR*, abs/1811.00525, 2018.

D. Kingma and J. Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015.

A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012.

M. Lecuyer, V. Atlidakis, R. Geambasu, D. Hsu, and S. Jana. Certified robustness to adversarial examples with differential privacy. *CoRR*, abs/1802.03471, 2018.

S. Levine, N. Wagener, and P. Abbeel. Learning contact-rich manipulation skills with guided policy search. In *ICRA*, 2015.

A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu. Towards deep learning models resistant to adversarial attacks. In *ICLR*, 2018.

M. Mirman, T. Gehr, and M. T. Vechev. Differentiable abstract interpretation for provably robust neural networks. In *ICML*, 2018.

K. Nar, O. Ocal, S. S. Sastry, and K. Ramchandran. Cross-entropy loss and low-rank features have responsibility for adversarial examples. *CoRR*, abs/1901.08360, 2019.

A. Raghunathan, J. Steinhardt, and P. Liang. Certified defenses against adversarial examples. In *ICLR*, 2018.

H. Salman, G. Yang, J. Li, P. Zhang, H. Zhang, I. Razenshteyn, and S. Bubeck. Provably robust deep learning via adversarially trained smoothed classifiers. In *NeurIPS*, 2019.

L. Schmidt, S. Santurkar, D. Tsipras, K. Talwar, and A. Madry. Adversarially robust generalization requires more data. In *NIPS*, 2018.

A. Shafahi, W. R. Huang, C. Studer, S. Feizi, and T. Goldstein. Are adversarial examples inevitable? In *ICLR*, 2019.

G. Singh, T. Gehr, M. Püschel, and M. T. Vechev. An abstract domain for certifying neural networks. *PACMPL*, 2019.

A. Sinha, H. Namkoong, and J. Duchi. Certifying some distributional robustness with principled adversarial training. In *ICLR*, 2018.

C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. J. Goodfellow, and R. Fergus. Intriguing properties of neural networks. *CoRR*, abs/1312.6199, 2013. URL `http://arxiv.org/abs/1312.6199`.

D. Tsipras, S. Santurkar, L. Engstrom, A. Turner, and A. Madry. Robustness may be at odds with accuracy. In *ICLR*, 2019.

T. Weng, H. Zhang, H. Chen, Z. Song, C. Hsieh, L. Daniel, D. S. Boning, and I. S. Dhillon. Towards fast computation of certified robustness for relu networks. In *ICML*, 2018.

E. Wong and J. Z. Kolter. Provable defenses against adversarial examples via the convex outer adversarial polytope. In *ICML*, 2018.

E. Wong, F. R. Schmidt, J. H. Metzen, and J. Z. Kolter. Scaling provable adversarial defenses. In *NeurIPS*, 2018.

Y. Wu, M. Schuster, Z. Chen, Q. V. Le, M. Norouzi, W. Macherey, M. Krikun, Y. Cao, Q. Gao, K. Macherey, J. Klingner, A. Shah, M. Johnson, X. Liu, L. Kaiser, S. Gouws, Y. Kato, T. Kudo, H. Kazawa, K. Stevens, G. Kurian, N. Patil, W. Wang, C. Young, J. Smith, J. Riesa, A. Rudnick, O. Vinyals, G. Corrado, M. Hughes, and J. Dean. Google's neural machine translation system: Bridging the gap between human and machine translation. *CoRR*, abs/1609.08144, 2016. URL http://arxiv.org/abs/1609.08144.

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