Input Hessian Regularization of Neural Networks

Waleed Mustafa [1] **Robert A. Vandermeulen** [2] **Marius Kloft** [1]

Abstract

Regularizing the input gradient has shown to be effective in promoting the robustness of neural networks. The regularization of the input's Hessian is therefore a natural next step. A key challenge here is the computational complexity. Computing the Hessian of inputs is computationally infeasible. In this paper we propose an efficient algorithm to train deep neural networks with Hessian operator-norm regularization. We analyze the approach theoretically and prove that the Hessian operator norm relates to the ability of a neural network to withstand an adversarial attack. We give a preliminary experimental evaluation on the MNIST and FMNIST datasets, which demonstrates that the new regularizer can, indeed, be feasible and, furthermore, that it increases the robustness of neural networks over input gradient regularization.

1. Introduction

Allowing for the automation of many previously impossible tasks, deep learning has brought about tremendous advances in machine learning. However, the lack of robustness of deep neural networks (DNNs) has caused some concern regarding their deployment, particularly in security or safety-critical applications. One such concern is the existence of adversarial examples (Szegedy et al., 2013)—samples that have been imperceptibly modified to trick a deep learning system to behave incorrectly. Many methods have been proposed to generate such examples (Carlini & Wagner, 2017; Kurakin et al., 2016; Athalye et al., 2018; Moosavi-Dezfooli et al., 2015) or defend neural networks against them (Hein & Andriushchenko, 2017; Singla & Feizi, 2020; Moosavi-Dezfooli et al., 2019; Madry et al., 2017).

A line of research suggests that the regularization of input gradients helps promote robustness (Finlay & Oberman, 2019; Ross & Doshi-Velez, 2018; Hein & Andriushchenko, 2017). To better theoretically understand the robustness of neural networks against adversarial attacks, Hein & Andriushchenko (2017) introduced a theoretical bound on the robustness of a network in terms of the norm of its gradient around a test sample. A large (input) gradient at a sample implies that it is possible to change the network output by only slightly manipulating the input. Using this theoretical insight, they propose a defense in the form of a regularizer that promotes the model's gradient norms to be small. Their regularizer assumes that the network is locally well-approximated by a linear function. This approximation can be poor when a function is highly non-linear, as it is the case with neural networks. Second-order approximations can be more suitable here and improve robustness. This was empirically supported in Fawzi et al. (2017), where the authors showed that adversarial directions can highly correlate with high principal curvatures (i.e., Hessian eigenvectors that correspond to high eigenvalues).

In this paper we propose to use second-order derivatives to better approximate the behavior of the classifier around a sample. We prove guarantees in the form of lower bounds on the robustness of neural networks in terms of second-order derivatives. Motivated by our theoretical results, we propose *Cross-Hölder Regularization*, a second order regularizer that penalizes input Hessians with large eigenvalues. While a naive approach to computing the gradient of the largest absolute eigenvalue of a Hessian is computationally expensive, we introduce an efficient procedure to compute it. We present a preliminary empirical evaluation on the MNIST and FMNIST datasets, which showcases that our method can outperform gradient-based regularization on robustness to adversarial samples.

The paper is organized as follows. In the remainder of this section we introduce the notation. In Section 2 we introduce a new theory that relates adversarial robustness to second-order differential properties of neural networks. In Section 3 we introduce a new regularizer as well as a method for computing the gradient of the operator norm of a Hessian. Finally in Section 5 we present an empirical analysis. Note that related work is discussed in Appendix A.

Notation and Problem Setup Let $\mathcal{X} \subset \mathbb{R}^d$ be a feature space and $\mathcal{Y} = \{1, \dots, K\}$ a label space. Denote the train-

[1]Computer Science Department, TU Kaiserslautern [2]Machine Learning Group, TU Berlin. Correspondence to: Waleed Mustafa <mustafa@cs.uni-kl.de >.

Workshop on "Beyond first-order methods in ML systems" at the 37th International Conference on Machine Learning, Vienna, Austria, 2020.

ing set by $\{(x_i, y_i)\}_{i=1}^N$. We are interested in classifiers that are derived from a scoring function $\mathbf{f} : \mathcal{X} \to \mathbb{R}^K$ from which we classify via $C_\mathbf{f}(x) := \arg\max_i \mathbf{f}_i(x)$. Training is therefore achieved by selecting a scoring function from a family of functions $\mathcal{F}$ indexed by real-valued parameters $\theta \in \mathbb{R}^D$, with the goal to minimize the objective $\Sigma_{i=1}^N \ell(\mathbf{f}_\theta(x_i), y_i)$, where $\ell : \mathbb{R}^K \times \mathcal{Y} \to \mathbb{R}^+$ is a loss function. For the rest of the paper we use $\mathbf{f}(x)$ instead of $\mathbf{f}_\theta(x)$ to avoid notational clutter. For a matrix $H \in \mathbb{R}^{d\times d}$, the operator norm $\|H\|_{p\to q}$ is defined as: $\|H\|_{p\to q} := \sup\{\|Hx\|_q : \|x\|_p = 1\}$. Our goal is to train classifiers that are robust under adversarial attacks. That is, if a classifier $C_\mathbf{f}$ predicts a class for an input x, it should stay consistent for similar examples x'. Formally, for a *robustness level* $\epsilon > 0$, we require $C_\mathbf{f}(x) = C_\mathbf{f}(x+\delta)$, for $\|\delta\|_p \leq \epsilon$.

2. Robustness and Second Order Derivatives

In this section, we prove a theoretical result that relates the robustness of a classifier to the input Hessian operator norm. Our first result is akin to Theorem 1 in Hein & Andriushchenko (2017), where the authors proved that robustness is controlled by the norm input gradient of a margin loss in a local ℓ_p ball around it. We extend their theory to control the robustness of classifier by the Hessian operator norm in an ℓ_p ball around an input example. Therefore, we show an explicit connection between the robustness and curvature. This connection allows us to derive our second order regularizer. To avoid notational clutter, we present our result for the binary classifier case. First we review the result by Hein & Andriushchenko (2017).

Theorem 1. *Let $x \in \mathbb{R}^d$ and $\mathbf{f} : \mathbb{R}^d \to \mathbb{R}^2$ be a binary-classifier scoring function with continuously differentiable components. Further let $\mathbf{f}_0(x) > \mathbf{f}_1(x)$ and define $f = \mathbf{f}_0 - \mathbf{f}_1$. For $\delta \in \mathbb{R}^d$ and p, q such that $\frac{1}{p} + \frac{1}{q} = 1$, if $\mathbf{f}_1(x+\delta) \geq \mathbf{f}_0(x+\delta)$, then the following bound holds*

$$\|\delta\|_p \geq \max_{R>0} \min \left\{ R, \frac{f(x)}{\max_{\|\gamma\|_p \leq R} \|\nabla f(x+\gamma)\|_q} \right\}. \tag{1}$$

The above bound implies that, if f is not steep around x (i.e. $\nabla f(\delta + x)$ is small), then one must move a large distance away from x to find an adversarial example (i.e. δ is large). This bound was shown to be tight in Hein & Andriushchenko (2017) when considering linear classifiers.

Note here that the dependence on curvature is only implicit. Indeed, if f is locally highly curved, then $\max_{\|\gamma\|_p \leq R} \|\nabla f(x+\gamma)\|_q$ is large. In order to design a curvature regularizer, we introduce the following result which makes the dependence on the curvature explicit.

Theorem 2. *Let $x \in \mathbb{R}^d$ and $\mathbf{f} : \mathbb{R}^d \to \mathbb{R}^2$ be a binary-classifier scoring function with continuously twice differentiable components. Further let $\mathbf{f}_0(x) > \mathbf{f}_1(x)$ and define $f = \mathbf{f}_0 - \mathbf{f}_1$. $H_f : \mathbb{R}^d \to \mathbb{R}^{d\times d}$ denotes the Hessian of f. For $\delta \in \mathbb{R}^d$ and p, q such that $\frac{1}{p} + \frac{1}{q} = 1$, if $\mathbf{f}_1(x+\delta) \geq \mathbf{f}_0(x+\delta)$, then the following bound holds:*

$$\|\delta\|_p \geq \max_{R>0} \min \left\{ R, \frac{f(x)}{\|\nabla f(x)\|_q + \frac{R}{2} K_R(f, x)} \right\}, \tag{2}$$

where $K_R(f, x) = \max_{\|\gamma\|_p \leq R} K(f, x+\gamma)$, for $K(f, x) := \|H_f(x)\|_{p\to q}$.

The proof of the above theorem is deferred to Appendix B. The theorem suggests that, to control the robustness of a classifier, we must control the local curvature around input points. Again, this requirement is on f, not the individual components $\mathbf{f}_i$. Therefore, the classifier components can have a large local curvature as far as the relative curvature is low, which is a less strict requirement. The dependence on the curvature is explicit in this result through the input Hessian operator norm of the margin function f locally around the input point. Since this bound is equivalent to the bound (1) in the case of linear scoring functions, it borrows its tightness.

In the next corollary, we extend our result to a multi-class classifier. It is straightforward to do so. If a classifier predicts a class t, then define $f^{(j)} = \mathbf{f}_t - \mathbf{f}_j$. We then apply theorem 2 to $f^{(j)}$, for $j \neq t$, and take the minimum of resulting lower bounds.

Corollary 1. *Let $x \in \mathbb{R}^d$ and $\mathbf{f} : \mathbb{R}^d \to \mathbb{R}^K$ be a multi-class classifier scoring function with continuously twice differentiable components. Further let $t = \arg\max_i \mathbf{f}_i(x)$ and define $f^{(j)} = \mathbf{f}_t - \mathbf{f}_j$ for $j \neq t$. For $\delta \in \mathbb{R}^d$ and p, q such that $\frac{1}{p} + \frac{1}{q} = 1$, if $\max_{j\neq t} \mathbf{f}_j(x+\delta) > \mathbf{f}_t(x+\delta)$, then $\|\delta\|_p$ is bounded below by:*

$$\max_{R>0} \min \left\{ R, \min_{j\neq t} \frac{f^{(j)}(x)}{\|\nabla f^{(j)}(x)\|_q + \frac{R}{2} K_R(f^{(j)}, x)} \right\}. \tag{3}$$

3. Cross Hölder Regularizer

In this section, we introduce a new technique for adversarial defending. The goal is to define a regularizer that improves the robustness of multi-class classifiers. This regularizer is based on (3) introduced in the last section. We start by deriving the regularization term to maximize the lower bound. Directly doing so is computationally impractical. Therefore, we reformulate it, to make it amenable for training.

Our goal is to train a classifier so that for an adversarial example, $x + \delta$, the norm of δ must be large, i.e. adversarial examples must have a large amount of distortion. To that end, we train our classifier in such a way that the right-hand

side of (3) is large at test samples. It suffices to maximize the term for each j independently, that is,

$$\max_{\theta} \frac{f^{(j)}(x)}{\left\|\nabla f^{(j)}(x)\right\|_q + \frac{R}{2} K_R(f,x)}. \tag{4}$$

It was argued in Hein & Andriushchenko (2017) that the cross entropy loss will naturally drive $f^{(j)}(x)$ to be large. Therefore we turn our attention to the denominator of (4). Minimizing the gradient term is standard (Drucker & Le Cun, 1992; Hein & Andriushchenko, 2017; Ross & Doshi-Velez, 2018). The Hessian term is, however, hard to minimize. The first issue is the maximization operator $\max_{\|\gamma\|_p \leq R}$, which makes evaluation of this term computationally expensive. To alleviate this issue, we assume that $\|H_{f^{(j)}}(x)\|_{p\to q} \approx \max_{\|\gamma\|_p \leq R} \|H_{f^{(j)}}(x+\gamma)\|_{p\to q}$. This can be achieved if we assume that the function f is locally well-approximated by a quadratic function.

Our final objective is thus to make $\left\|\nabla f^{(j)}(x)\right\|_q + \frac{R}{2}\|H_{f^{(j)}}(x)\|_{p\to q}$ small for all $j \neq y$, where we now treat R as a tuning parameter. Relaxing this regularizer a bit and taking a summation of all possible classes we arrive at our proposed training objective, where we enforce our network to be robust around training samples. We define our loss $L\left(\{x_i, y_i\}_{i=1}^n\right)$ to be

$$\begin{aligned} &\sum_{i=1}^{n} \ell(y_i, x_i) + \\ &\lambda_1 \sum_{i=1}^{n} \sum_{j=1, j\neq y_i}^{K} \left(\|\nabla f^{(j)}(x_i)\|_q + \lambda_2 K(f^{(j)}, x_i) \right), \end{aligned} \tag{5}$$

where $\ell(.,.)$ is the cross-entropy loss, and λ_1 and λ_2 are regularization parameters. The last two terms of (5) together form the *Cross-Hölder Regularizer*. Notice that when $\lambda_2 = 0$ we get the Cross-Lipschitz regularizer.

4. Training with Hessian Operator norm

To minimize the new objective (5), we will use stochastic gradient descent. Thus we need to evaluate the gradient of the objective. Obtaining the gradients needed for SGD is usually done through *automatic differentiation* (Griewank & Walther, 2008), as implemented in popular machine learning toolkits (e.g., Tensorflow (Abadi et al., 2016)). It is sufficiently efficient to compute the gradient of the loss and gradient norm. Computing the gradient of the operator norm of the Hessian is, however, very computationally inefficient for algorithmic differentiation. Obtaining the Hessian is very costly for large neural network architectures, let alone computing its operator norm and computing the gradient of the combined operations with respect to the model parameters.

We now describe how to efficiently calculate the gradient of the Hessian operator norm. In what follows we use H_θ to denote $H_{f^{(j)}}(x)$. Define a function $g : \mathbb{R}^d \times \mathbb{R}^D \to \mathbb{R}^+$ as

$$g(\theta, v) = \|H_\theta v\|_q, \tag{6}$$

where θ is the model parameters and $v \in \{v \in \mathbb{R}^d : \|v\|_p = 1\}$. Therefore, the operator norm is

$$\|H_\theta\|_{p\to q} = \max_{\|v\|_p = 1} g(\theta, v).$$

Let v^* be the vector attaining the maximum of (6). If v^* is a unique maximizer, then, by Danskin's theorem (Danskin, 2012), we have

$$\nabla_\theta \max_{\|v\|_p = 1} g(\theta, v) = \nabla_\theta g(\theta, v^*).$$

It was also shown in Madry et al. (2017) that, even if v^* is not a unique maximum, the direction $-\nabla_\theta g(\theta, v^*)$ is still a descent direction. Therefore it suffices to compute the gradient at the maximum v^* for training.

There still remains the problem of computing both v^* and $\nabla_\theta \|H_\theta v^*\|_q$. One can find v^* by optimizing (6). It is, however, again the problem of computing the Hessian matrix H_θ. Fortunately, the Hessian matrix appears above only multiplied by a vector. This allows us to use an efficient algorithm to obtain a computational graph for Hessian vector multiplication (Pearlmutter, 1994). With this algorithm one can use algorithmic differentiation to obtain both $\nabla_\theta \|H_\theta v^*\|_q$ and $\nabla_v \|H_\theta v\|_q$[1] with a computational expense on the same order as backpropagation (Griewank & Walther, 2008; Pearlmutter, 1994). In a nutshell, our approach reduces to solving

$$\begin{aligned} \min_{\theta} \max_{v_{ij}} &\sum_{i=1}^{n} \ell(y_i, x_i) + \\ &\lambda_1 \sum_{i=1}^{n} \sum_{j\neq y_i}^{K} \left(\|\nabla f^{(j)}(x_i)\|_q + \lambda_2 \|H_{f^{(j)}}(x_i) v_{ij}\|_q \right), \end{aligned}$$

for $\|v_{ij}\|_p = 1$. Algorithm 4 summarizes our training approach. The inner loop is used to estimate v^* which is then used in the last step to estimate the gradient of the objective. In our experiments, we found setting $T = 10$ and $o = 0.1$ in the algorithm suffices to get a reasonable approximation to the Hessian operator norm. It is worth noting that thanks to the hessian vector multiplication algorithm (Pearlmutter, 1994), computing the gradient enjoys a linear running time.

5. Experiments

In this section, we present some first experimental results for evaluating the adversarial robustness of models trained

[1] When $\|H_\theta v\|_q$ is differentiable as a function of v which is the case for $q = 2$.

Algorithm 1 Hessian Norm Regularization training

Input: Training data $\{(x_i, y_i)\}_{i=1}^N$, a model $\mathbf{f}$, initialized parameters θ, number of epochs J, operator norm optimization iterations T, hyper parameters λ_1, λ_2, and learning rates o and l.
for $j = 1$ **to** J **do**
  Sample a mini batch $\{(x_i, y_i)\}_{i=1}^b$ of size b.
  Initialize v_{ij}s i.i.d. from $\mathcal{N}(0, I)$ defined on $\mathbb{R}^d$, for $i \in 1, \ldots, b$ and $j \in 1, \ldots, K-1$.
  for $t = 1$ **to** T **do**
    Update v_{ij}s for each i and j using:

$$v_{ij} = v_{ij} + o * \nabla_{v_{ij}} \|H_{f^{(j)}}(x_i) v_{ij}\|_q$$
$$v_{ij} = \frac{v_{ij}}{\|v_{ij}\|_p}$$

  end for
  Update parameters θ by:

$$\theta = \theta + \sum_{i=1}^{b} \sum_{j \neq y_i}^{K} l * \nabla_\theta \Big(\ell(x_i, y_i) + \lambda_2 \|\nabla f^{(j)}(x_i)\|_q + \lambda_1 \lambda_2 \|H_{f^{(j)}}(x_i) v_{ij}\|_q \Big)$$

end for

with the Cross-Hölder regularizer. We test the robustness on two datasets, namely MNIST and FMNIST. As baseline defenses, we use the gradient based approach of Cross-Lipschitz (Hein & Andriushchenko, 2017). We also compare against adversarial training (Madry et al., 2017) as it is considered state of the art defense (Athalye et al., 2018). The detailed experimental setup and further results are presented in Appendix C.

Evaluating the robustness is a hard problem due to the high non-convexity of the attack optimization problem (Madry et al., 2017). Many defenses claim the robustness against weak attacks while they fail to be robust against harder attacks (Athalye et al., 2018; Carlini et al., 2019). Therefore, to evaluate the robustness of models trained with a given defense mechanism, it is important to use the strongest available attacks. We use the recommended PGD attack with multiple iterations and multiple random initializations (Carlini et al., 2019). Since, in adversarial training, an attack is used in the training procedure we use also a different attack in the evaluation phase (Carlini et al., 2019). We use two PGD attacks with 50 iterations and 10 random restarts. The first one uses the cross-entropy loss as objective, as in adversarial training. The second one uses the objective of Carlini and Wagner (CW) (Carlini et al., 2019; Athalye et al., 2018). As a robustness metric, we report on adversarial accuracy, that is, the accuracy after we have applied a given attack. We report the worst-case results of the two attacks, which is the minimum adversarial accuracy of the two attacks. We report the adversarial accuracy for different robustness levels ε. Figure 1 depicts the worst-case accuracy of the three defenses on the MNIST dataset. The figure shows that all three defenses are quite effective up to robustness level 4.0 after which Cross-Hölder outperforms Cross-Lipschitz. Figure 2

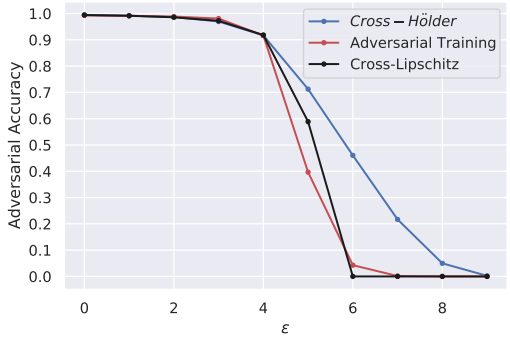


Figure 1. Adversarial accuracy versus robustness levels for MNIST dataset.

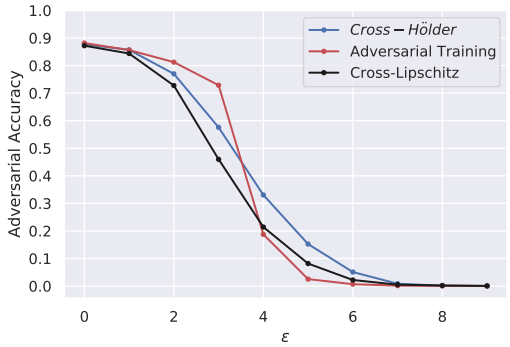


Figure 2. Adversarial accuracy versus robustness levels fashion MNIST dataset.

shows the worst case adversarial accuracy on the FMNIST dataset. Again Cross-Hölder outperforms Cross-Lipschitz but is inferior to adversarial training up to robustness level 4.0. Experiments on FMNIST showed the effectiveness of Cross-Hölder regularizer over Cross-Lipschitz while being competitive with adversarial training.

6. Conclusion

In this paper we present a method for an input gradient regularization of neural networks, which we call *Cross-Hölder regularization*. We have proven both theoretically and empirically that this regularization is useful for increasing the resilience of deep neural networks against adversarial attacks. In future work, we wish to analyze the proposed regularization algorithm on further datasets.

Acknowledgement

Marius Kloft acknowledges support by the German Research Foundation (DFG) award KL 2698/2-1 and by the Federal Ministry of Science and Education (BMBF) awards 01IS18051A and 031B0770E. Robert A. Vandermeulen acknowledges support by the Berlin Institute for the Foundations of Learning and Data (BIFOLD) sponsored by the German Federal Ministry of Education and Research (BMBF). The experiments were partly executed on the high performance cluster "Elwetritsch II" at the TU Kaiserslautern (TUK) which is part of the "Alliance for High Performance Computing Rheinland-Pfalz"(AHRP). We kindly acknowledge the support of the RHRK especially when using their DGX-2.

References

Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., Kudlur, M., Levenberg, J., Monga, R., Moore, S., Murray, D. G., Steiner, B., Tucker, P. A., Vasudevan, V., Warden, P., Wicke, M., Yu, Y., and Zhang, X. Tensorflow: A system for large-scale machine learning. In *OSDI*, 2016.

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

Bietti, A., Mialon, G., Chen, D., and Mairal, J. A kernel perspective for regularizing deep neural networks. *arXiv preprint arXiv:1810.00363*, 2018.

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 *Joint European conference on machine learning and knowledge discovery in databases*, pp. 387–402. Springer, 2013.

Brendel, W., Rauber, J., and Bethge, M. Decision-based adversarial attacks: Reliable attacks against black-box machine learning models. *arXiv preprint arXiv:1712.04248*, 2017.

Carlini, N. and Wagner, D. A. Towards evaluating the robustness of neural networks. *2017 IEEE Symposium on Security and Privacy (SP)*, pp. 39–57, 2017.

Carlini, N., Katz, G., Barrett, C., and Dill, D. L. Ground-truth adversarial examples. *CoRR*, abs/1709.10207, 2017. URL http://arxiv.org/abs/1709.10207.

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

Cissé, M., Bojanowski, P., Grave, E., Dauphin, Y., and Usunier, N. Parseval networks: Improving robustness to adversarial examples. In *ICML*, 2017.

Danskin, J. M. *The theory of max-min and its application to weapons allocation problems*, volume 5. Springer Science & Business Media, 2012.

Drucker, H. and Le Cun, Y. Improving generalization performance using double backpropagation. *IEEE Transactions on Neural Networks*, 3(6):991–997, 1992.

Fawzi, A., Moosavi-Dezfooli, S.-M., Frossard, P., and Soatto, S. Classification regions of deep neural networks. *arXiv preprint arXiv:1705.09552*, 2017.

Finlay, C. and Oberman, A. M. Scaleable input gradient regularization for adversarial robustness. *arXiv preprint arXiv:1905.11468*, 2019.

Goodfellow, I. J., Shlens, J., and Szegedy, C. Explaining and harnessing adversarial examples. *CoRR*, abs/1412.6572, 2014.

Griewank, A. and Walther, A. *Evaluating derivatives: principles and techniques of algorithmic differentiation*, volume 105. Siam, 2008.

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

Hein, M. and Andriushchenko, M. Formal guarantees on the robustness of a classifier against adversarial manipulation. In *NIPS*, 2017.

Kannan, H., Kurakin, A., and Goodfellow, I. Adversarial logit pairing. *arXiv preprint arXiv:1803.06373*, 2018.

Kurakin, A., Goodfellow, I. J., and Bengio, S. Adversarial machine learning at scale. *CoRR*, abs/1611.01236, 2016.

Madry, A., Makelov, A., Schmidt, L., Tsipras, D., and Vladu, A. Towards deep learning models resistant to adversarial attacks. *CoRR*, abs/1706.06083, 2017.

Meng, D. and Chen, H. Magnet: A two-pronged defense against adversarial examples. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, CCS '17, pp. 135–147, New York, NY, USA, 2017. Association for Computing Machinery. ISBN 9781450349468. doi: 10.1145/3133956.3134057. URL https://doi.org/10.1145/3133956.3134057.

Moosavi-Dezfooli, S.-M., Fawzi, A., and Frossard, P. DeepFool: a simple and accurate method to fool deep neural networks. *2016 IEEE Conference on Computer Vision and Pattern Recognition*

(CVPR), pp. 2574–2582, jun 2015. ISSN 10636919. doi: 10.1109/CVPR.2016.282. URL http://ieeexplore.ieee.org/document/7780651/ http://arxiv.org/abs/1511.04599.

Moosavi-Dezfooli, S.-M., Fawzi, A., Uesato, J., and Frossard, P. Robustness via curvature regularization, and vice versa. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 9078–9086, 2019.

Papernot, N., McDaniel, P. D., Wu, X., Jha, S., and Swami, A. Distillation as a defense to adversarial perturbations against deep neural networks. *2016 IEEE Symposium on Security and Privacy (SP)*, pp. 582–597, 2016.

Papernot, N., McDaniel, P., Goodfellow, I., Jha, S., Celik, Z. B., and Swami, A. Practical black-box attacks against machine learning. In *Proceedings of the 2017 ACM on Asia conference on computer and communications security*, pp. 506–519, 2017.

Pearlmutter, B. A. Fast Exact Multiplication by the Hessian. *Neural Computation*, 6(1):147–160, jan 1994. ISSN 0899-7667. doi: 10.1162/neco.1994.6.1.147. URL http://www.mitpressjournals.org/doi/10.1162/neco.1994.6.1.147.

Ramachandran, P., Zoph, B., and Le, Q. V. Searching for activation functions. *CoRR*, abs/1710.05941, 2017.

Ross, A. S. and Doshi-Velez, F. Improving the adversarial robustness and interpretability of deep neural networks by regularizing their input gradients. In *Thirty-second AAAI conference on artificial intelligence*, 2018.

Singla, S. and Feizi, S. Curvature-based robustness certificates against adversarial examples, 2020. URL https://openreview.net/forum?id=Skgq1ANFDB.

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

Tramèr, F., Kurakin, A., Papernot, N., Goodfellow, I., Boneh, D., and McDaniel, P. Ensemble adversarial training: Attacks and defenses. *arXiv preprint arXiv:1705.07204*, 2017.

Tu, L. W. *An Introduction to Manifolds*. Universitext. Springer New York, New York, NY, 2011. ISBN 978-1-4419-7399-3. doi: 10.1007/978-1-4419-7400-6. URL http://link.springer.com/10.1007/978-1-4419-7400-6.

Zhang, H., Yu, Y., Jiao, J., Xing, E. P., Ghaoui, L. E., and Jordan, M. I. Theoretically principled trade-off between robustness and accuracy. *arXiv preprint arXiv:1901.08573*, 2019.

A. Related Work

Since the work of (Szegedy et al., 2013) and (Biggio et al., 2013), neural network models have been shown to be vulnerable to small input perturbations. Many elaborate attacks have been introduced since then (Carlini & Wagner, 2017; Brendel et al., 2017; Moosavi-Dezfooli et al., 2015). As a response, many defenses—training algorithms that harden the attack problem—have also been devised (Madry et al., 2017; Papernot et al., 2016; Hein & Andriushchenko, 2017). In this section, we give a brief overview of the prominent attacks and related defenses.

Attacks can be classified into *black-box* (Brendel et al., 2017) and *white-box* (Carlini & Wagner, 2017) attacks. In a white-box attack, the attacker has full knowledge of the classifier used (e.g., the neural network architecture and its parameters). On the other hand, black-box attacks can only query a classifier but do not have the full knowledge of the model. Both types of attacks have the same goal: they are trying to solve the problem (Szegedy et al., 2013; Carlini & Wagner, 2017):

$$\min_{\delta} \|\delta\|_p \quad \text{s.t. } C(x) \neq C(x+\delta), \tag{7}$$

or alternatively (Madry et al., 2017; Goodfellow et al., 2014)

$$\max_{\delta} \ell(x+\delta, y) \quad \text{s.t. } \|\delta\|_p \leq \epsilon. \tag{8}$$

The adversarial attack problem is usually not convex due to the constraint in the problem (7) and the objective of (8).

In Madry et al. (2017) the authors argue that projected gradient descent (PGD) with multiple steps provides a good solution to the problem. They showed that, when PGD is run with different random restarts, the optimized objective values seem to concentrate. They then concluded that PGD does not get stuck in a bad local minimum. Therefore, they advised using PGD with multiple steps to craft adversarial examples and to evaluate the robustness of classifiers. The authors of Carlini & Wagner (2017) attempt to solve the problem (7) by a Lagrangian relaxation. They do not specify a robustness level ϵ constraint to the attack. However, they minimize the distance of the crafted adversarial example to the input while adding a margin like loss to derive the constraint of (7) to be satisfied. A combination of the two approaches was used by the authors of Athalye et al. (2018). In it the authors used the PGD algorithm to solve a relaxed objective of (7).

A more realistic setting for practical attacks is the black-box setting (Papernot et al., 2017; Brendel et al., 2017), in which the attacker can access the attacked model only through queries. In Papernot et al. (2017) the authors proposed to query the attacked model with random samples and use the returned label to train a local replica model. They then craft adversarial examples on the replica in a white-box fashion. Those examples are then shown to transfer to the attacked model, that is, they are also misclassified under the black-box model. The authors of Brendel et al. (2017) proposed a heuristic search for adversarial examples. Given an input example, they generate a random example with a different class. Starting from it, they sample a random direction to move according to a *proposal* distribution. This distribution is designed such that moving in its sampled directions, the distance between the original and current point is reduced. After each move, the new point is checked if it still has a different class. If yes, the point is accepted, otherwise, it is rejected and a new sample direction is sampled. After a large number of iterations, they show that they could craft adversarial samples that could fool industrial deployed systems.

As more attacks are devised, many defense mechanisms have been proposed. Defenses can be characterised into data augmentation (Madry et al., 2017; Kannan et al., 2018), regularization based (Bietti et al., 2018; Cissé et al., 2017) and detection (Meng & Chen, 2017; Grosse et al., 2017).

Perhaps the most widely used defense mechanism is a data augmentation method known as *adversarial training* (Madry et al., 2017). In adversarial training one generates adversarial examples using an attack and includes these examples in the training set. It was proposed concurrently with the introduction of adversarial examples in Szegedy et al. (2013). The proposed attack method in Szegedy et al. (2013) was, however, slow and hence not suitable as an inner loop in the training procedures. A fast attack was proposed in Goodfellow et al. (2014) to enhance adversarial training. It was, however, later shown that such a fast method is inadequate to promote real robustness (Tramèr et al., 2017) as authors introduce the masking gradient phenomena. Masking gradient or more generally obfuscating gradients (Athalye et al., 2018) is a term coined to describe the situation where a defense causes the input gradient of a model to be non-informative. This prevents weak attacks from altering example label but stronger attacks like PGD would still be successful (Athalye et al., 2018). When a strong attack used in adversarial training, real robustness can be achieved (Athalye et al., 2018; Madry et al., 2017). A theoretical motivation to adversarial training was introduced in Madry et al. (2017) where the authors viewed it as solving a robust optimization problem

$$\min_{\theta} \mathbb{E}[\max_{\|\delta\|_p \leq \epsilon} \ell(\mathbf{f}(x+\delta), y)].$$

They then utilize Danskin's theorem (Danskin, 2012) to argue that one can solve the inner maximization with PGD and substitute solution δ^* to solve the outer minimization. This argument, however, requires the inner maximization to be solved which can not be guaranteed due to the non-concavity of the problem. Adversarial training with a PGD

has withstood a lot of tests (Athalye et al., 2018) and is therefore considered the state of the art. Different flavors of adversarial training have also been proposed. For example in Kannan et al. (2018), the authors proposed to use the distance between the *logits* of clean and adversarial examples as a regularization term. Similarly, Zhang et al. (2019) proposed to use a *cross-entropy* loss between the predicted class probability of the model on a clean example and on adversarial examples generated based on that clean example.

Another class of defenses is that of regularization based approaches. In this approach, one identifies a property of the model and relates it to adversarial vulnerability and designs a regularization term to reduce such a property. For example, in Bietti et al. (2018); Cissé et al. (2017) the authors proposed to regularize the Lipschitz constant of the prediction model by reducing the operator norms of the weight matrices. The norm of the input gradient has been proposed to use as a regularizer to defend against adversarial examples (Hein & Andriushchenko, 2017; Ross & Doshi-Velez, 2018; Finlay & Oberman, 2019). Interestingly in Hein & Andriushchenko (2017), the authors derived a principled approach to defense. They theoretically derived a relationship between the norm of the adversarial noise and the maximum norm of the input gradient locally around an input example. They then proposed to regularize the norm of the gradient at the input point to promote robustness. Since the regularizer is designed to derive the norm of the gradient to be small at the input point and not the ball around it, it might be less powerful as there is an implicit assumption that the model is locally linear around the input. In our work, we propose a second order approach which provides a better local approximation of a model. We follow the principled approach of Hein & Andriushchenko (2017) by first deriving a relationship between adversarial robustness and the maximum operator norm of the Hessian matrix in a ball around an input example. Motivated by this result, we propose a regularizer to derive the operator norm of the Hessian to be small and thus promote robustness.

Related to our proposed method is the work of Moosavi-Dezfooli et al. (2019) and Singla & Feizi (2020). In Moosavi-Dezfooli et al. (2019), the authors experimentally noticed that adversarial training reduces the curvature of the loss function of neural networks, where they define curvature as the maximum magnitude of Hessian eigenvalues. They further proposed to regularize the curvature to induce robustness to the trained models. They used two main approximations in their regularizer. First, they used a finite difference method to compute a Hessian vector multiplication and therefore their proposed regularizer is simply a difference of gradients regularization. In our work, we used an exact and efficient Hessian vector multiplication algorithm (Pearlmutter, 1994). Second, the authors estimated the Hessian operator norm by the norm of the Hessian multiplied by the gradient assuming that the gradient and the vector associated with the largest eigenvalue of the Hessian are almost parallel. We experimentally show that it is not a tight lower bound on the operator norm of the Hessian. Therefore, we use an optimization approach to estimate the true operator norm of the Hessian. On the other hand Singla & Feizi (2020) proposes a certificate, a computable lower bound on the norm of adversarial noise, on adversarial robustness based on the curvature. They provided an upper bound on the operator norm of the input Hessian of the model and proposed to use it as a regularizer. The upper bound however is independent of the input example and therefore not tight for many inputs as we show in the experiments section that the Hessian operator norm distribution has a large tail on real inputs let alone pathological ones.

B. Proof of Theorem 2

In this section, we give the proof of theorem 2. We restate the theorem below and give its proof.

Theorem 3. *Let $x \in \mathbb{R}^d$ and $\mathbf{f} : \mathbb{R}^d \to \mathbb{R}^2$ be a binary-classifier scoring function with continuously twice differentiable components. Further let $\mathbf{f}_0(x) > \mathbf{f}_1(x)$ and define $f = \mathbf{f}_0 - \mathbf{f}_1$. $H_f : \mathbb{R}^d \to \mathbb{R}^{d\times d}$ denotes the Hessian of f. For $\delta \in \mathbb{R}^d$ and p, q such that $\frac{1}{p} + \frac{1}{q} = 1$, if $\mathbf{f}_1(x+\delta) > \mathbf{f}_0(x+\delta)$, then the following bound holds:*

$$\|\delta\|_p \geq \max_{R>0} \min \left\{ R, \frac{f(x)}{\|\nabla f(x)\|_q + \frac{R}{2} K_R(f,x)} \right\},$$

where $K_R(f,x) = \max_{\|\gamma\|_p \leq R} K(f, x+\gamma)$, for $K(f,x) := \|H_f(x)\|_{p\to q}$.

Proof. By Taylor theorem with remainder (Tu, 2011) we have,

$$f(x+\delta) = f(x) + \langle \nabla f(x), \delta \rangle + \int_0^1 \delta^T H_f(x+t\delta)\delta(1-t)dt,$$

Given that $\mathbf{f}_1(x+\delta) \geq \mathbf{f}_0(x+\delta)$, we get $f(x+\delta) \leq 0$ and hence,

$$f(x) \leq \langle -\nabla f(x), \delta \rangle + \int_0^1 -\delta^T H_f(x+t\delta)\delta(1-t)dt.$$

By applying Hölder inequality, monotonicity of integration

and the definition of the operator norm, we get,

$$\begin{aligned}
f(x) \leq & \langle -\nabla f(x), \delta \rangle + \int_0^1 -\delta^T H_f(x+t\delta)\delta(1-t)dt \\
\leq & \|\nabla - f(x)\|_q \|\delta\|_p \\
& + \int_0^1 \|\delta\|_p \|H_f(x+t\delta)\delta\|_q (1-t)dt \\
\leq & \|\nabla - f(x)\|_q \|\delta\|_p \\
& + \int_0^1 \|\delta\|_p^2 \|H_f(x+t\delta)\|_{p\to q}(1-t)dt \\
= & \|\nabla f(x))\|_q \|\delta\|_p \\
& + \|\delta\|_p \int_0^1 \|H_f(x+t\delta)\|_{p\to q}\|\delta\|_p (1-t)dt.
\end{aligned}$$

For a fixed $R > 0$, assume that $\|\delta\|_p \leq R$, then by monotonicity of integration we have,

$$\begin{aligned}
& \int_0^1 \|H_f(x+t\delta)\|_{p\to q}\|\delta\|_p(1-t)dt \\
& \leq \int_0^1 \max_{\|\gamma\|_p \leq R} \|H_f(x+\gamma)\|_{p\to q}\|\delta\|_p(1-t)dt \\
& = K_R(f,x)R\int_0^1 (1-t)dt \\
& = \frac{R}{2}K_R(f,x).
\end{aligned}$$

Substituting in (9) we get,

$$\begin{aligned}
f(x) &\leq \|\nabla f(x)\|_q\|\delta\|_p + \|\delta\|_p \frac{R}{2}K_R(f,x) \\
&\leq \|\delta\|_p \left(\|\nabla f(x))\|_q + \frac{R}{2}K_R(f,x) \right).
\end{aligned}$$

Thus we have that

$$\|\delta\|_p \geq \frac{f(x)}{\|\nabla f(x)\|_q + \frac{R}{2}K_R(f,x)}.$$

Since we restricted $\|\delta\|_p$ to be at most R, the last bound is only correct when the left-hand side is less than R. So we restrict the lower bound to be at most R. That is,

$$\|\delta\|_p \geq \min\left\{ R, \frac{f(x)}{\|\nabla f(x)\|_q + \frac{R}{2}K_R(f,x)} \right\}.$$

Maximizing of R we get,

$$\|\delta\|_p \geq \max_{R>0} \min\left\{ R, \frac{f(x)}{\|\nabla f(x))\|_q + \frac{R}{2}K_R(f,x)} \right\}.$$

□

C. Experiments

In this section, we give a detailed experimental setup and results. We begin by investigating the estimation of the Hessian operator norm and compare this to the norm of the Hessian multiplied by the gradient to the Hessian operator norm. We then present experiments where we evaluate adversarial robustness on both the MNIST and FMNIST datasets using our adversarial defense versus gradient based methods and adversarial training.

C.1. Hessian Operator Norm Estimation

In this experiment we investigate the Hessian operator norm approximation presented in Moosavi-Dezfooli et al. (2019). The authors propose to estimate the Hessian[2] operator norm with the norm of the Hessian multiplied by the gradient, we denote this quantity by $|Hg|$. We compare this to the true Hessian operator norm computed by gradient ascent optimization that we denote by $|Hv^*|$. We train a convolution neural network with the architecture below on MNIST dataset with no defense. Then for each data point on the evaluation set we compute both $|Hg|$ and $|Hv^*|$ and plot the histogram (see figure 3). Figure 3a shows that the histogram of $|Hv^*|$ is shifted to the right and has a heavier tail. There is however a large overlap between the two histograms. We then investigate whether this overlap correspond to $|Hg|$ being a good estimate or not. Figure 3b depicts the histogram of the difference $|Hv^*| - |Hg|$ where we notice that it has a mode at around 5 with a heavier right tail. Therefore, we conclude that $|Hg|$ is not a tight lower bound and hence it is better to use the true operator norm for regularizing the Hessian. On the other hand, we see from the heavy tail of the histogram of $|Hv^*|$ that any global bound on the Hessian operator norm is not tight for most of the input examples and therefore using it as regularizing term as in (Singla & Feizi, 2020) is restrictive, especially when the true operator norm can be efficiently used as in our approach.

C.2. Adversarial Robustness Evaluation

In this section, we evaluate the adversarial robustness of models trained with the Cross-Hölder regularizer. We test the robustness on two datasets, namely MNIST and FMNIST.

C.2.1. Setup

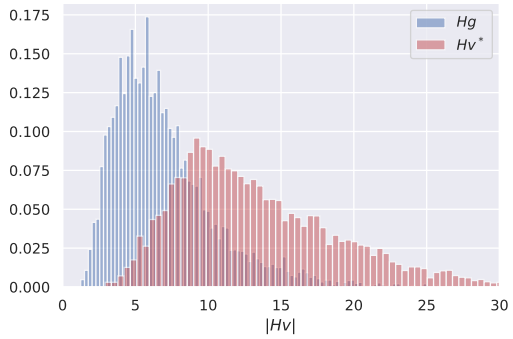
As baseline defenses, we use the gradient based approach of Cross-Lipschitz (Hein & Andriushchenko, 2017). We also compare against adversarial training (Madry et al., 2017) as it is considered state of the art defense (Athalye et al., 2018). We used the same network architecture for both the MNIST

[2]The Hessian is computed on $f^{(j)}(x)$, where j is the index of the second largest logit.

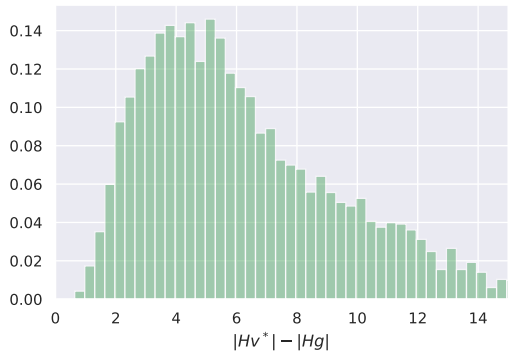
and FMNIST datasets which is the same architecture used in Carlini et al. (2017). It consists of 4 convolution layers with 3×3-sized filters and the respective filter maps sizes of (32,32,64,64), followed by two fully connected layers of size 200 connected to the logits layer with size 10. We introduced two changes to the architecture to ensure that it is twice differentiable. First, we replaced the ReLU activation function with SWISH $x\sigma(x)$ (Ramachandran et al., 2017), where σ is the sigmoid function. Second, max-pooling was replaced by strided convolution every second convolutional layer.

Robustness Evaluation Evaluating the robustness is a hard problem due to the high non-convexity of the attack optimization problem (Madry et al., 2017). Many defenses claim robustness against weak attacks while they fail to be robust against harder attacks (Athalye et al., 2018; Carlini et al., 2019). Therefore it is important to use the strongest available attacks to evaluate the robustness of models trained with a given defense mechanism. We use the recommended PGD attack with multiple iterations and multiple random initializations (Carlini et al., 2019). Since in adversarial training an attack is used in the training procedure it is recommended to use a different attack in the evaluation phase (Carlini et al., 2019). Therefore we use two iterative PGD attacks with 50 iterations and 10 random restarts. The first is using cross entropy loss as objective, the same as adversarial training. The second is using the Carlini and Wagner (CW) objective (Carlini et al., 2019; Athalye et al., 2018). We also report the worst case results of the two attacks, which is the minimum adversarial accuracy of the two attacks. The robustness metric we report is adversarial accuracy, the accuracy after we have applied a given attack. We report adversarial accuracy for different robustness levels.

Hyperparameters We implement three defenses that depend on several hyperparameters. Adversarial training has the parameter ε, the robustness level used for the training attack. Cross-Hölder has the parameters λ_1 and λ_2. We choose the hyperparameters by leaving out 5% of the data outside of the training set. We employ a grid search to select the hyperparameter on the left out evaluation set. The criteria to select the winning hyperparameter is for it to achieve at least 99% accuracy on the clean evaluation set in case of MNIST and 88% in the case of FMNIST. We then select the hyperparameter by selecting the best sum of adversarial accuracy on robustness levels 3.0 and 4.0. The selected parameters for adversarial training were $\epsilon = 5.0$ on MNIST and $\epsilon = 4.0$ on FMNIST. For Cross-Lipschitz, we selected $\lambda_1 = 0.2$ for MNIST and $\lambda_1 = 0.15$ for FMNIST. Finally for Cross-Hölder, we select $\lambda_1 = 0.02$, $\lambda_2 = 0.5$ for MNIST and $\lambda_1 = 0.2$, $\lambda_2 = 0.5$ for FMNIST.



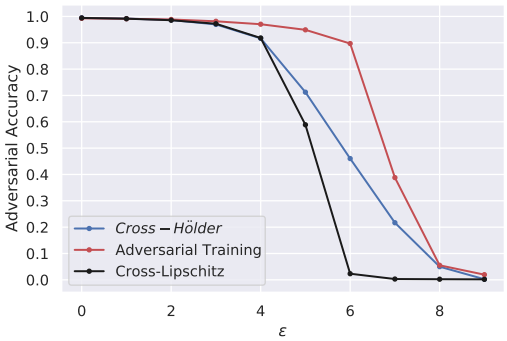
(a)



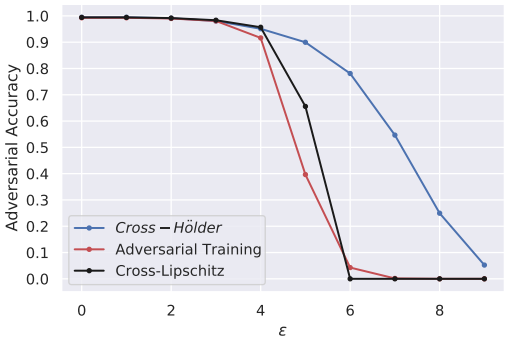
(b)

Figure 3. Hessian operator norm estimation comparison.

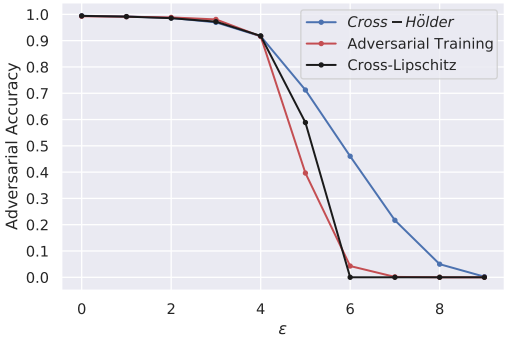
Results Figure 1 shows the result of adversarial evaluation on the MNIST dataset. Figure 4a shows the result of applying the PGD attack with the cross entropy loss. The figure shows that all three defenses are quite effective up to robustness level 3.0 after which we see that both Cross-Lipschitz and Cross-Hölder accuracies decrease with Cross-Lipschitz decreases faster. This was expected as in adversarial training the same attack was used in training. Figure 4b shows the results of applying PGD with CW loss. On this attack Cross-Hölder behaves better than both adversarial training and Cross-Lipschitz after robustness level 4.0. Figure 5 shows the result of the worst case attack which again shows that Cross-Hölder outperforms both Cross-Lipschitz and adversarial training after robustness level 4.0. On MNIST, our experiments suggest that Cross-Hölder regularizer outperforms both Cross-Lipschitz and adversarial training.



(a) PGD on cross entropy loss.

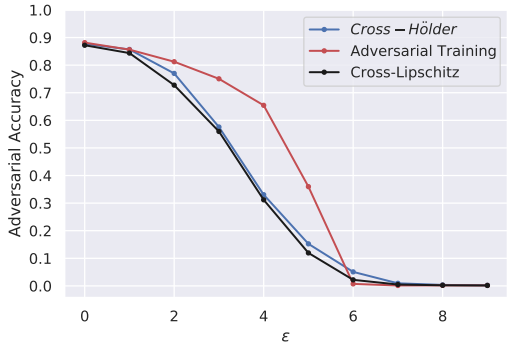


(b) PGD on CW loss.

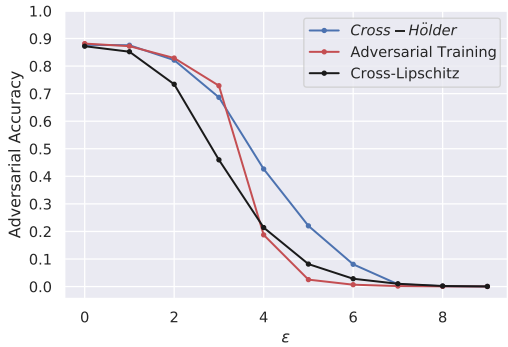


(c) Worst case attack.

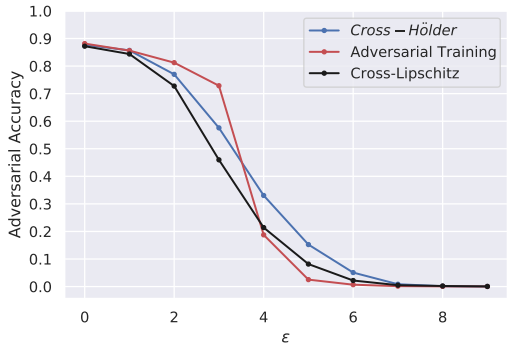
Figure 4. Adversarial accuracy versus robustness levels for MNIST dataset.



(a) PGD on cross entropy loss.



(b) PGD on CW loss.



(c) Worst case attack.

Figure 5. Adversarial accuracy versus robustness levels fashion MNIST dataset.