ADVERSARIAL ROBUSTNESS THROUGH WIDE LOCAL MINIMA: A SIMPLE TRAINING TECHNIQUE

Anton S. Khritankov
Moscow Institute of Physics and Technology,
HSE University
Moscow, Russian Federation
akhritankov@hse.ru

Andrey S. Veprikov
Moscow Institute of Physics and Technology
Moscow, Russian Federation

Sergey A. Smirnov
Institute for Information Transmission Problems
Moscow, Russian Federation

ABSTRACT

Achieving adversarial robustness is a critical aspect of ensuring the security and reliability of machine learning models, particularly in applications where trustworthiness is paramount. This paper delves into the theoretical aspects and impact of width of the local minima and learning parameters on adversarial robustness in Deep Neural Networks (DNNs) for image classification tasks. Through our investigation of gradient learning methods, we identify that certain optimization parameters can enhance robustness without compromising prediction quality. Building on these findings, we introduce a novel adversarial defense technique aimed at improving the model's resilience against attacks.

1 INTRODUCTION

Adversarial robustness (Khamaiseh et al. 2022, Bountakas et al. 2023) of a machine learning model refers to the ability of the model to maintain its performance even when it is presented with intentionally crafted inputs designed to deceive or mislead the model. These inputs, known as adversarial examples, are purposefully perturbed data points that are indistinguishable to humans but can cause the model to make incorrect predictions.

Adversarial attacks on machine learning models can be used to exploit edge cases, leading to potential incorrect responses and security breaches. In this case robustness of the model protects from such manipulation of the responses. As a result, adversarial robustness is included as one of the important quality characteristics of trustworthiness of a machine learning system (Zhang & Li, 2019; Hu et al., 2021; Akhtar et al., 2021) so that the system behaves predictably and consistently even in the presence of adversarial inputs.

Adversarial training (Madry et al., 2017; Kannan et al., 2018) is one of popular methods to improve robustness. Such training involves augmenting the training data with adversarial examples generated during training with FGSM (Goodfellow et al., 2014), PGA (Zhu et al., 2024), MIM (Dong et al., 2018), ALP (Goodman et al., 2019). As a result, the model learns to better generalize and make accurate predictions (Bai et al., 2021) even on adversarial inputs. Adversarial detection methods (Cohen et al., 2020; Aldahdooh et al., 2022) help protect the model during inference. Feature squeezing reduces the precision of the inputs, defensive distillation methods smoothes out decision boundaries, input randomization makes more difficult to input adversarial examples, outlier detection methods (Deng et al., 2021) help identify adversarial inputs, activation clustering monitors the neuron activations in the model to detect the adversarial inputs (Zhang et al., 2020).

Adversarial robustness is commonly evaluated by comparing the performance of the model on clean data with the performance in presence of adversarial inputs (Zhu et al., 2024; Dong et al., 2018). Adversarial inputs can be produced by black-box methods, which only use inputs and outputs of the model during inference, and white-box methods that have access to the model structure and

parameters. Overall, white-box methods such as FGSM (Goodfellow et al., 2014), PGA (Zhu et al., 2024) produce more powerful adversarial examples with fewer interactions with the model.

In this paper, we investigate whether the properties of the loss function near the minimum found by the training procedure influence the robustness of the model, so that selecting a specific minimum from a range of many similar ones helps improve the adversarial robustness.

The approach we propose differs from the adversarial training in that we do not change nor augment the input training data, the loss function and the optimizer remain the same. It is applied during training only and as such differs from adversarial detection and evasion methods that protect the model during inference. As the proposed approach only concerns finding appropriate minima of the given loss function, it could be well combined with other defense methods thus further improving the robustness.

The rest of the paper is organized as follows. In the next Section 2, we briefly provide a background on adversarial defense methods and position the proposed approach among them, in Section 3 we describe the motivation and research methodology and provide a more formal explanation of the method in Section 4. An experimental evaluation on an image classification task on a popular MNIST dataset with FGSM attack is given in Section 5 and additional details in Appendix A.

2 BACKGROUND

Adversarial attacks are widely studied in the literature. There are several surveys (Khamaiseh et al., 2022; Liang et al., 2022; Bountakas et al., 2023) that propose similar classifications for attacks and defense methods. In this study we are concerned only with white-box attacks, that is when an adversary has complete knowledge about the structure and parameters of the predictive model, the optimization algorithm, the loss function and the training dataset (Biggio et al., 2013; Papernot et al., 2017; Khamaiseh et al., 2020). The goal of a white-box attack is to add a small perturbation ρ to the original item x to get an adversarial example $\hat{x} = x + \rho$, so that the model predicts an incorrect class or specific target class. We briefly cover some of the methods to obtain $\hat{x}$.

The untargeted C&W attacks (Carlini & Wagner, 2017) generate adversarial examples by solving the following optimization problem:

$$\min_{\rho} \|\rho\|_p + c \cdot g(x + \rho),$$
$$\text{s.t. } x + \rho \in [0; 1]^n,$$

where $\|\cdot\|_p$ is the L_p norm and g is penalty function to avoid solving a constrained problem $f(\theta, x + \rho) = \hat{y}$ (Carlini & Wagner, 2017). An example of such function could be $g(x) = 1 - L(x, \hat{y}; \theta)$ for a loss function L. A Fast Gradient Sign method (FGSM) is more computationally efficient (Goodfellow et al., 2014). For item x and a small parameter ε it solves the problem

$$\hat{x} = x - \varepsilon \cdot \text{sign}\left\{\nabla_X L(x, y; \theta)\right\},$$

Defense methods against adversarial attacks could be divided into three groups (Liang et al., 2022; Bountakas et al., 2023). Methods in the first group modify the structure of the predictive model to improve adversarial robustness. Papernot et al. (Papernot et al., 2016) introduced defensive distillation, that is transfer knowledge from a non-robust teacher neural network to a more simple but robust student network. Gradient regularization (Addepalli et al., 2020; Ma et al., 2020) penalizes slight variations in the input data during training, so that the model becomes immune to small perturbations.

Methods in the second group change the input data. In adversarial training, the training data is augmented with adversarial examples (Szegedy et al., 2013; Madry et al., 2018; Sinha et al., 2018), so that the learned model is more robust to the included types of attacks. Online defense methods remove the adversarial noise from the input data (Samangouei et al., 2018). For instance, PuVAE method (Hwang et al., 2019) denoises inputs with a pretrained variational autoencoder.

The third method group add another network to the given model to defend it from adversarial examples. Some of them detect and remove adversarial examples in the training data (Meng & Chen, 2017; Cohen et al., 2020). The integrated defense technique (He et al., 2017; Yu et al., 2019) splits the model into two parts. The first is public and it is the original model known to the attacker. The

second is private and combines a few neural networks unknown to the attacker. As a result a pure white-box attack is complicated.

Properties of the local minima of the loss function have been studied in the context of adversarial training (Madry et al., 2018; Sinha et al., 2018). This study takes on a different way and relies on the stochasticity of the gradient during training instead of adding adversarial examples to the data. As a result the overall process and input data does not change. However, some theoretical findings could also be applied in this study.

The "flatness" of a local minimum is usually defined as $\max_{\|\delta\|_\infty \leq \varepsilon}\{L(f(x+\delta,\theta),y)\}$ (Stutz et al., 2021). The authors also show that achieving flat minima helps with the adversarial robustness (Stutz et al., 2021; Izmailov et al., 2018). They, however, do not propose any methods for avoiding sharp minima, and the relationship between the minimum on the model weights and the input data is not explored. Learning rate or batch size may better correlate with generalization than non-flatness or sharpness (Andriushchenko et al., 2023).

A Stochastic Averaging Method (SAM) (Foret et al., 2020) for finding flat minima looks for parameters that lie in neighbourhoods with uniformly low loss by solving a min-max optimization problem. Flooding method (Liu et al., 2022) is to correct the loss function to $\tilde{L} = |L - b| + b$, so that the model reaches sharp minima rarely. A role of the noise in SGD is important for generalization (Wu et al., 2022). The authors explore a special SGD noise structure: concentrated in sharp directions of the local landscape and proportional to the loss function.

Compared to the aforementioned groups of methods, the approach taken in this study does not easily fit into any of them. The practical value of the study is the training technique applied at the training stage, with the optimizer, the model architecture, and the loss function remain unchanged.

The question of why achieving flatter minima may lead to better generalization, adversarial robustness or both is studied empirically (Stutz et al., 2021; Izmailov et al., 2018; Foret et al., 2020; Andriushchenko et al., 2023). We propose our theoretical argument on the matter in this paper.

3 METHODOLOGY

Let us start with the motivation and rationale for this research and then briefly describe the research methodology. In a classification task the loss function $L(X, y; \theta)$ has many local minima with similar loss value but different boundary surfaces. Let us look for a "wide" local minima in terms of model parameters θ. We hypothesize that under certain conditions such minima also provide for "wider" gap between different classes for variations in the input data X.

The underspecified (D'Amour et al., 2022) learning pipelines, as commonly found in the industrial settings, may result in different trained models and thus provide an opportunity to improve adversarial robustness.

The main research question of this study can be stated as follows. Is it possible to improve the specification of a learning pipeline so that the training process results in a model whose parameters lie in the aforementioned local minima and thus have better adversarial robustness?

It seems that the simplest approach to select appropriate minima θ_0 without modifying the loss function, the pipeline and training data is to tune the hyperparameters of the learning algorithm.

We argue that if the training procedure utilizing an stochastic gradient descent or a similar algorithm converges with a higher learning rate and smaller mini-batch size near a local minimum, then the model parameters lie in a "wider" local minimum and therefore the trained model would be more robust to adversarial examples.

The overall research methodology is as follows. First, we formally define such "wide" local minima and explain why selection of such minima leads to better adversarial robustness. Second, we consider an image classification task on MNIST dataset with a convolution network model and solve it using Adam and SGD optimizer for a range of mini-batch size and learning rate parameters.

The image classification task on MNIST is frequently used to evaluate adversarial defense methods (Goodfellow et al., 2014; Carlini & Wagner, 2017; Kannan et al., 2018). Other defense techniques

have been investigated for this task and we have relevant references (Hwang et al., 2019; Zhu et al., 2024) to compare with.

For each set of hyper-parameters we measure adversarial robustness of the model during training as its prediction accuracy on a set of adversarial examples generated with FGSM for the test set. Better accuracy compared to other hyper-parameter set would mean higher adversarial robustness. We select the FGSM white-box gradient-based method to generate the adversarial examples because the method is well-known and often used in benchmarks on adversarial robustness (Madry et al., 2017; Cohen et al., 2020; Deng et al., 2021).

4 Learning Rate and Adversarial Robustness

Let us first formally define what a "wide" local minimum is. Recall, that a level set $V(c)$ of a real-valued function $\phi(x) : \mathbb{R}^d \to \mathbb{R}$ is a set where the function takes on a given constant value $\phi(x) = c$. Let x^* be a local minimum, $\phi(x^*) < c$, then the distance from x^* to the level set is $d(c) = \min_{x \in V(c)}\{\|x - x^*\|\}$.

Definition 1 (width of a minimum) *The width $w_x(x^*, c)$ of a local minimum at x^* of $\phi(x)$ with respect to x for some c is a function*

$$w_x(x^*, c) = \frac{d(c)}{c - \phi(x^*)}.$$

So that, larger $d(c)$ for given c corresponds to a wider minimum. We may write simply $w(c)$ or $w(x^*, c)$ instead of full $w_x(x^*, c)$.

Consider a loss function $L(X, y; \theta)$. If x^* is a tight local minimum with $w(x^*, c)$ smaller than for other minima, then a smaller perturbation $\|\delta\|_\infty$ is sufficient for an adversarial example $x^* + \delta$ to cause a prediction error. On the other hand, if the minimum is wider, larger perturbations are needed. The following Theorem 1 relates the width of the minimum to a Lipschitz constant of the function near the minimum.

Theorem 1 *For all subsets $\mathcal{S} \subset \mathbb{R}^d$ such that $\max_{x \in \mathcal{S}}\{\|x - x^*\|\} \leq d(c)$ it holds that*

1. *If ϕ is continuous on $\mathcal{S}$, then for all $x \in \mathcal{S}$ it holds that $|\phi(x)| \leq |c|$.*
2. *If ϕ is convex on $\mathcal{S}$, then for all $x \in \mathcal{S}$ it holds that*

$$\phi(x) - \phi(x^*) \leq \frac{1}{w(x^*, c)} \cdot \|x - x^*\|.$$

Let g^t be an estimator of the gradient $\nabla_\theta L(X, y; \theta)$ and γ_t be the learning rate parameter. Then, for a classical stochastic gradient descent (SGD) algorithm (Bengio et al., 2009; Danilova et al., 2022) the update is defined as follows

$$\theta^{t+1} = \theta^t - \gamma_t g^t,$$

The algorithm and its variants have been studied extensively with various loss functions $L(X, y; \theta)$. In a typical computer vision problem the loss usually has many minima and locally quasi-convex (Konnov, 2003; Ke & Kanade, 2007) nearby as SGD tends to avoid saddles.

Following the results (Hazan et al., 2015; Mertikopoulos et al., 2020) show that SGD converges to narrow local minima θ^* with smaller learning rates γ_t for $c < \epsilon$, where ϵ is the solution accuracy

$$\gamma_t \sim K_L^{-1} \sim w(\theta^*, c),$$

where K_L is the Lipschitz constant of the loss function L. According to the Theorem 1 it holds that $K_L \sim w(\theta^*, c)^{-1}$.

As we can see, the width of a minimum $w(\theta^*, c)$ as defined in Definition 1 is directly related to the learning rate γ_t needed for the SGD algorithm to stay near the minimum θ^*. Such definition is easier to interpret than the classical flatness (Stutz et al., 2021).

Let M be the upper bound of the loss L near the minimizer θ^*. For SGD convergence, the size of the mini-batch has to be large enough to compensate for the noise in estimator g_t (Hazan et al., 2015),

$$B \sim M^2 \sim c^2 \sim \left(\frac{d(c)}{w(\theta^*, c)}\right)^2 \sim w(\theta^*, c)^{-2}.$$

According to Theorem 1 $M \sim c^2$, and according to Definition 1 $M \sim 1/w(\theta^*, c)$, if $d(c)$ is given.

Therefore, the width of the local minimum $w(\theta^*, c)$ an SGD algorithm converges to, depends on both mini-batch size B and the learning rate γ. While the exact relation between them is not yet established, we could expect SGD does not converge to a narrow minima when either learning rate γ is large or the mini-batch size B is small.

We now informally state the theoretical result, that justifies the training technique for adversarial robustness. The following Theorem 2 relates the weight robustness achieved during training with the input data robustness of a predictive model.

Theorem 2 (wide minima training) *Let θ^* be a minimizer to the loss $L(X, y; \theta) < \epsilon$ for a small ϵ and given X^*, y^* an SGD algorithm converges to. If the predictor $f(x; \theta)$ is a function of the inner product of its parameters $f(x; \theta) = f(x^T\theta)$, then the following holds around X^*, y^*, θ^*:*

1. *L is locally quasi-convex w.r.t. θ and w.r.t. X,*
2. *width w_θ w.r.t. to θ and the width w_X w.r.t. X have a common non-negative multiplier,*
3. *the larger the width w_θ, the larger the width w_X.*

The idea of the proof is to show that second derivatives of the loss with respect to X and θ have a common multiplier $\nabla^2_{zz} f = \partial^2 f / \partial z^2$, $z = X\theta$. If a minimum X^*, y^*, θ^* is global, then consider a small enough neighbourhood, in which L is locally quasi-convex, thereby $\nabla^2_{zz} f \geq 0$. Therefore, finding a wider minimum for L with respect to θ ensures smaller $\nabla^2_{zz} f$, so that the corresponding minimum with respect to θ is also wider.

Theorem 2 can be demonstrated in simple cases analytically. Let us estimate the width $w(x^*, c)$ and show that the larger the distance $d_\theta(c)$ with respect to model parameters θ only, the larger the distance with respect to the inputs $d_X(c)$. Let X be the feature matrix, y be the target vector in a binary classification task. Let L be the log-loss function for a single layer perceptron model with a sigmoid $p_i = \sigma(X_i^T\theta)$ activation function:

$$L(X, y; \theta) = -\frac{1}{n} \cdot \sum_{i=1}^{n} [y_i \ln(p_i) + (1 - y_i) \cdot \ln(1 - p_i)].$$

Then the first and the second partial derivatives of the loss L w.r.t. model parameters θ and inputs X are given by:

$$\nabla_\theta L = \frac{1}{n} X^T(p - y),\ \nabla^2_\theta L = \frac{1}{n} X^T D \cdot X,$$
$$\nabla_X L = \frac{1}{n}(p - y)\theta^T,\ \nabla^2_X L = \frac{1}{n} D \otimes \theta \cdot \theta^T,$$
$$\text{where } D = \text{diag}(p \odot (1 - p)).$$

Here, we use '$\odot$' symbol for the element-wise product of two vectors and '$\otimes$' for the outer product, so that $a \otimes b = \sum_{ij} a_i b_j$. Now, if we found a local minimum θ_0, we get $\nabla_\theta L = X^T(p - y)/n = 0$, then either of the following holds a) $p - y$ is in $\ker(X^T)$ or b) $p - y = 0$ and the minimum is global. If $p \approx y$, then it is a minimum $\nabla_\theta L \approx 0$.

Near a global minimum, $D = \text{diag}(p \odot (1 - p))$ and $p \in (0, 1)$, so that L is quasi-convex and $\nabla^2_\theta L$ is positive semi-definite with a maximum eigenvalue $\lambda > 0$. Eigenvalues for both $\nabla^2_X L$ and $\nabla^2_\theta L$ are proportional to elements of D.

In case of an overparameterized model $p - y \approx 0$ is reachable (Wu et al., 2018). Following the Taylor expansion of L near θ^*, the minimum distance $d_\theta(c)$ to the level set is given by

$$d_\theta(c) \approx \sqrt{\frac{2(c - L)}{\lambda}} \approx \sqrt{\frac{2c}{\lambda}}.$$

Batch Size		Learning Rate																	
		0.0003		0.0006		0.001		0.003		0.006		0.01		0.03		0.06		0.1	
		AVG	STD	AVG	STD	AVG	STD	AVG	STD	AVG	STD	AVG	STD	AVG	STD	AVG	STD	AVG	STD
1	epoch	117	1	116	3	41	18	4	5	5	3	0	1	0	1	0	0	0	1
	acc	**98.81**	0.09	**98.74**	0.07	97.52	0.35	94.70	1.47	69.04	33.26	39.98	39.69	11.14	0.48	10.92	0.59	10.72	0.58
	adv_acc	**87.33**	3.40	**96.70**	0.53	43.19	6.72	30.60	10.11	15.47	4.62	20.47	13.41	11.14	0.48	10.92	0.59	10.72	0.58
2	epoch	116	1	114	4	117	1	31	40	23	15	3	4	1	1	0	0	0	0
	acc	**98.84**	0.06	**98.78**	0.08	**98.64**	0.21	95.56	3.14	91.50	2.88	71.46	33.76	11.14	0.48	10.89	0.64	10.92	0.59
	adv_acc	63.91	7.13	**93.94**	0.41	**96.83**	0.37	32.95	13.27	22.32	5.92	21.25	10.76	11.14	0.48	10.89	0.64	10.92	0.59
4	epoch	117	2	114	4	115	3	34	40	57	42	23	28	1	1	2	1	0	1
	acc	**98.91**	0.07	**98.73**	0.13	**98.72**	0.08	97.06	0.69	89.52	5.88	73.53	35.26	22.71	26.01	11.14	0.48	10.72	0.58
	adv_acc	38.24	2.80	**80.62**	5.57	**95.51**	0.67	32.86	9.72	30.60	7.42	25.37	16.97	22.61	25.77	11.14	0.48	10.72	0.58
8	epoch	94	24	112	3	114	4	111	10	45	53	60	26	2	1	1	1	1	1
	acc	**98.83**	0.15	**98.81**	0.08	**98.69**	0.10	**98.16**	0.17	94.29	4.32	89.35	1.95	53.28	38.34	11.14	0.48	11.14	0.48
	adv_acc	28.24	3.52	54.98	3.84	**88.08**	1.55	78.91	11.67	33.83	8.28	54.19	23.63	16.90	14.15	11.14	0.48	11.14	0.48
16	epoch	88	20	108	3	117	1	114	3	68	43	78	22	4	3	2	1	2	1
	acc	**98.90**	0.05	**98.82**	0.22	**98.80**	0.10	**98.60**	0.16	96.62	0.70	94.54	2.38	41.94	41.92	11.14	0.46	10.72	0.57
	adv_acc	25.23	2.65	36.12	5.12	64.02	3.45	**90.93**	6.07	46.73	13.12	56.59	25.80	19.72	11.49	11.14	0.46	10.72	0.57
32	epoch	106	13	97	19	101	17	116	3	115	2	109	4	34	41	2	1	2	1
	acc	**98.81**	0.10	**98.88**	0.10	**98.78**	0.07	**98.69**	0.10	97.99	0.23	94.62	1.28	57.50	42.64	11.14	0.48	11.14	0.48
	adv_acc	25.94	3.09	32.85	3.68	41.51	3.13	93.16	2.18	72.56	6.52	70.24	22.27	33.79	25.75	11.14	0.48	11.14	0.48
64	epoch	99	17	90	25	93	19	116	2	115	3	113	3	24	14	1	1	1	2
	acc	**98.81**	0.07	**98.80**	0.10	**98.83**	0.07	**98.67**	0.08	**98.39**	0.16	97.46	0.34	90.69	4.63	11.14	0.48	10.93	0.58
	adv_acc	23.02	2.46	28.42	3.13	36.16	3.44	79.43	5.45	**82.93**	8.08	**82.25**	7.88	29.81	25.63	11.14	0.48	10.93	0.58
128	epoch	98	11	100	10	103	16	115	3	117	2	115	4	71	36	1	0	1	0
	acc	**98.81**	0.08	**98.79**	0.12	**98.83**	0.11	**98.71**	0.12	**98.60**	0.11	**98.12**	0.11	92.87	3.25	26.13	33.04	25.92	32.58
	adv_acc	18.33	3.56	25.03	4.51	30.86	4.31	59.12	5.30	**86.06**	4.99	**89.87**	3.90	44.96	29.18	10.46	2.00	10.55	1.80

Table 1: Accuracy on adversarial test set (FGSM) for Adam optimizer. Average (AVG) and approximate standard deviation (STD) over five runs. For each mini-batch size B and learning rate γ: epoch when the best adversarial accuracy is achieved (epoch), the corresponding neutral accuracy (acc) and the adversarial accuracy (adv_acc).

As we can see, larger $w_\theta(X^*, c)$ w.r.t θ gives smaller λ in D and if $p - y \approx 0$ for an overparameterized model, $\dim(\ker(X)) > 0$, then eigenvalues of $\nabla^2_X L$ are non-negative and also smaller, and L is quasi-convex near the given training data X^* and $w_X(X^*, c)$ is larger. Thus the model is expected to have higher adversarial robustness, and the higher $\dim(\ker(X))$, the more there are such θ^*.

Based on the findings, the proposed training technique could be stated as follows. By tuning the mini-batch size B and learning rate γ, find the convergency edge for the chosen optimizer. Adjust the parameters within the edge to maintain $f(X; \theta) - y \approx 0$ and $0 < L(X, y; \theta) < \epsilon$ as long as possible during training while still converging. Pick the snapshot of the model with the best accuracy and adversarial accuracy.

5 Experimental Results

We evaluate our theoretical hypotheses in a series of experiments. We consider an image classification task on a well-known MNIST dataset. Following the results of the previous sections, we specifically check that the bigger the step, and the smaller the mini-batch size, the higher would be the adversarial accuracy. The architecture of the neural network architecture is typical for the 2D image classification tasks: two convolutional layers, each followed by a ReLU activation function and a max pooling layer, two fully connected layers, the first with 100 output features followed by a ReLU activation, the final layer outputs logits for the target ten classes. No dropout layers or batch normalization applied during training.

We compare Adam and SGD optimizers as implemented in PyTorch. Both algorithms are run with a fixed mini-batch sizes B. For each set of parameters we run the experiment for at most 120 epochs for five times. During training, we save the model parameters θ^k every 1024 samples seen by the optimizer, so that we store $\theta^{1024*k-1}$ and θ^{1024*k}. After each epoch the model was evaluated on the test samples and on the adversarial samples derived from the same test samples by FGSM method for the currently available model parameters θ.

Experiments are run on a single HPC node with the following setup: CentOS 7, 2 x Intel Xeon E5 2680 v4 (2,40 GHz, 14 cores), 128 Gb RAM. The experiment source code is implemented in Python 3.12.4, IBM Adversarial Robustness Toolbox 1.18.1 (Nicolae et al., 2018), PyTorch 2.3.1 and GNU Parallel 20240722 (Tange, 2023).

At Table 1 we see that the Adam optimizer achieves maximum accuracy (acc) for different mini-batch sizes B and learning rates γ. Whereas, the adversarial accuracy (adv_acc) for a given batch

Batch Size		Learning Rate																	
		0.003		0.006		0.01		0.03		0.06		0.1		0.3		0.6		1	
		AVG	STD	AVG	STD	AVG	STD	AVG	STD	AVG	STD	AVG	STD	AVG	STD	AVG	STD	AVG	STD
1	epoch	47	19	117	1	48	8	1	2	1	1	0	1	0	0	1	1	2	1
	acc	**98.90**	0.12	97.87	0.38	96.53	1.36	66.09	31.50	11.14	0.46	11.14	0.48	10.72	0.57	10.47	0.50	10.43	0.52
	adv_acc	55.19	2.91	**89.48**	2.01	82.27	2.52	31.63	15.20	11.14	0.46	11.14	0.48	10.72	0.57	10.47	0.50	10.43	0.52
2	epoch	35	6	41	11	115	2	16	7	0	0	1	1	0	0	0	0	1	1
	acc	**98.92**	0.09	**98.88**	0.11	**98.30**	0.21	95.53	0.82	84.31	8.39	11.14	0.46	10.92	0.59	10.72	0.57	10.47	0.50
	adv_acc	40.94	4.44	53.45	3.08	**83.80**	3.80	67.99	3.97	37.36	6.22	11.14	0.46	10.92	0.59	10.72	0.57	10.47	0.50
4	epoch	45	5	35	4	33	3	97	10	12	4	1	1	1	1	0	0	0	0
	acc	**98.89**	0.09	**98.93**	0.07	**98.93**	0.15	96.78	0.71	96.42	1.22	90.44	4.84	11.14	0.48	10.92	0.59	10.72	0.58
	adv_acc	33.58	2.03	39.44	2.78	48.60	3.89	**89.26**	1.12	71.90	4.99	43.61	8.11	11.14	0.48	10.92	0.59	10.72	0.58
8	epoch	78	15	47	10	37	3	75	38	86	9	25	8	2	1	1	1	0	0
	acc	**98.84**	0.13	**98.87**	0.11	**98.86**	0.12	**98.71**	0.27	97.29	1.02	96.69	1.01	11.14	0.46	11.14	0.48	10.92	0.59
	adv_acc	27.71	2.64	33.45	3.04	38.81	3.75	66.06	8.72	**89.38**	1.23	77.04	4.52	11.14	0.46	11.14	0.48	10.92	0.59
16	epoch	103	9	74	3	51	13	30	6	100	35	115	1	4	3	2	1	1	1
	acc	**98.72**	0.09	**98.77**	0.13	**98.83**	0.19	**98.80**	0.13	**98.60**	0.17	97.77	0.25	94.98	1.16	11.14	0.46	11.14	0.48
	adv_acc	22.57	2.84	27.36	2.51	32.99	1.50	43.05	3.15	70.23	7.32	**89.87**	1.59	54.90	4.52	11.14	0.46	11.14	0.48
32	epoch	45	61	103	11	87	19	39	7	35	6	41	8	44	14	2	1	2	1
	acc	60.26	35.63	**98.79**	0.10	**98.78**	0.12	**98.87**	0.09	**98.89**	0.15	**98.85**	0.07	97.64	0.36	11.14	0.46	11.14	0.46
	adv_acc	17.99	5.37	22.13	2.78	27.53	2.95	35.43	1.52	44.67	4.19	55.18	6.14	**80.06**	6.51	11.14	0.46	11.14	0.46
64	epoch	0	0	44	60	108	8	72	19	41	10	31	4	112	6	21	28	1	1
	acc	40.88	16.53	60.36	35.45	**98.36**	0.59	**98.83**	0.15	**98.86**	0.15	**98.80**	0.06	**98.34**	0.14	45.62	46.93	11.14	0.46
	adv_acc	17.02	7.32	18.23	5.14	21.61	3.09	29.00	1.30	33.98	1.56	42.44	2.22	81.70	2.36	40.16	39.46	11.14	0.46
128	epoch	2	1	0	0	65	59	99	12	66	24	43	4	35	5	50	48	1	1
	acc	37.47	14.20	40.95	16.54	71.57	37.34	**98.79**	0.09	**98.90**	0.06	**98.85**	0.15	**98.93**	0.09	80.31	38.57	11.35	0.00
	adv_acc	17.48	6.06	17.02	7.22	16.27	6.51	24.41	2.35	29.28	1.56	32.69	2.26	49.63	7.58	40.40	31.97	11.35	0.00

Table 2: Accuracy on adversarial test set (FGSM) for SGD optimizer. Average (AVG) and approximate standard deviation (STD) over five runs. For each mini-batch size B and learning rate γ: epoch when the best adversarial accuracy is achieved (epoch), the corresponding neutral accuracy (acc) and the adversarial accuracy (adv_acc).

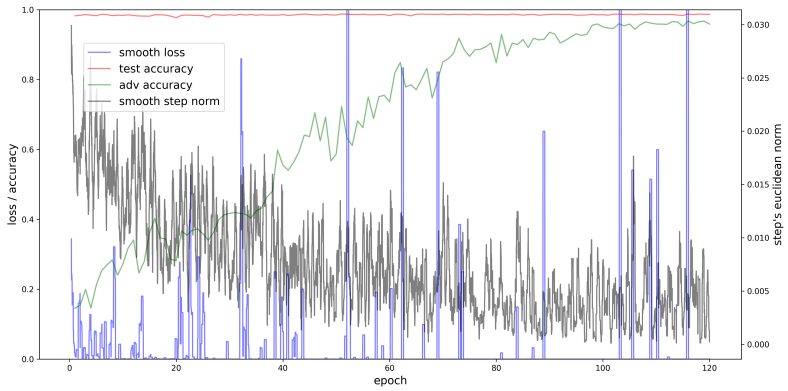

Figure 1: Learning curves for Adam ($\gamma = 0.001$, $B = 2$). The loss on the training data (blue), the step norm $\|\theta^k - \theta^{k-1}\|$ (black), accuracy on the adversarial examples derived from the test set using FGSM (green), accuracy on the clean test data (red).

size is high only for the most one or two highest learning rates while the training still converges. The highest adversarial accuracy is attained for $B = 2$ and $\gamma = 0.001$. Note that when learning rate is high, the variation in the adversarial accuracy between runs is also very large, so that the standard deviation (STD) is not meaningful.

At Table 2 we see that the SGD optimizer achieves the same high accuracy on clean data (acc) than that of Adam. The adversarial accuracy for some batch size B is only high when the learning rate γ is the maximum while the training still converges. Training with lower learning rates still achieves good accuracy, but not the adversarial accuracy. The highest adversarial accuracy is achieved for a range of batch size and learning rates, lower than that of Adam optimizer, however. Also note, that compared to the Adam optimizer, the variation in the adversarial accuracy is much lower.

Let us now take a look at the learning curves. At Fig. 1 the learning curve (blue) shows that the zero loss is achieved in the first three to five epochs and then the training continues with loss mostly zero so that the model parameters may still experience a random walk (Ishida et al., 2020). The step norm $\|\theta^k - \theta^{k-1}\|$ (black) gradually decreases while the accuracy on clean data remains the same (red) and accuracy on adversarial data (green) grows and reaches maximum near the end of the training.

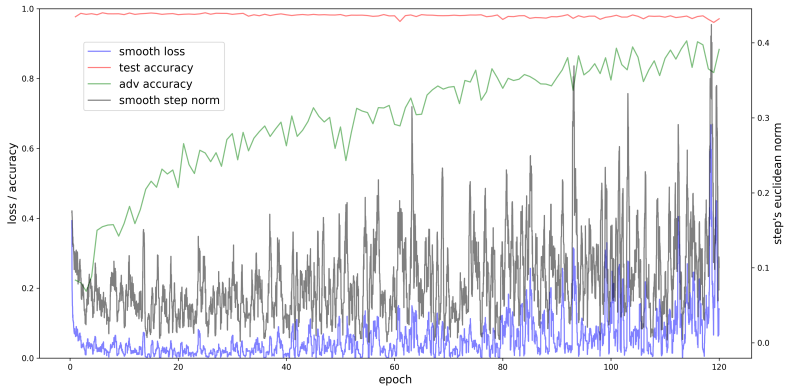

Figure 2: Learning curves for SGD ($\gamma = 0.1$, $B = 16$). The loss on the training data (blue), the step norm $\|\theta^k - \theta^{k-1}\|$ (black), accuracy on the adversarial examples derived from the test set using FGSM (green), accuracy on the clean test data (red)

A somewhat similar picture can be seen at Fig. 2 for the SGD optimizer. The loss (blue) reaches very low values quickly during the first five to ten epochs. Due to the stochasticity of the gradient and the fixed learning rate, the loss does not get very close to zero for the SGD optimizer. The step norm (black) remains almost the same for the duration of the training, while the training eventually tends to diverge as the epochs grow. The adversarial accuracy (green) grows faster at first than that of Adam optimizer, but the maximum accuracy is still lower.

The adversarial accuracy that could be achieved by an optimizer depends on of the size of the model. At Fig. 3 we show the best adversarial accuracy during training while varying the number of the output features in the first connected layer from the original 100. During the most runs for large models, the model achieved the accuracy more than 0.98 on the clean test data. Adversarial accuracy increases as the number of parameters grows. When the model is too large, third layer output dimension equals 1000 with batch size $B = 16$, the loss and gradient reach zero during training. In this case, the training stops and adversarial accuracy stops growing. It is possible to increase the accuracy by reducing the batch size, thereby introducing more noise into the gradient.

In order to minimize threats to internal validity, that is that the measurements are actually results of the predicted effect, we take the following steps. First, we study two different optimizers with different learning regimes. Second, we repeat the same experiment five times to limit confoundness on unknown factors. Third, we run full factorial experiments so that no parameter values are left

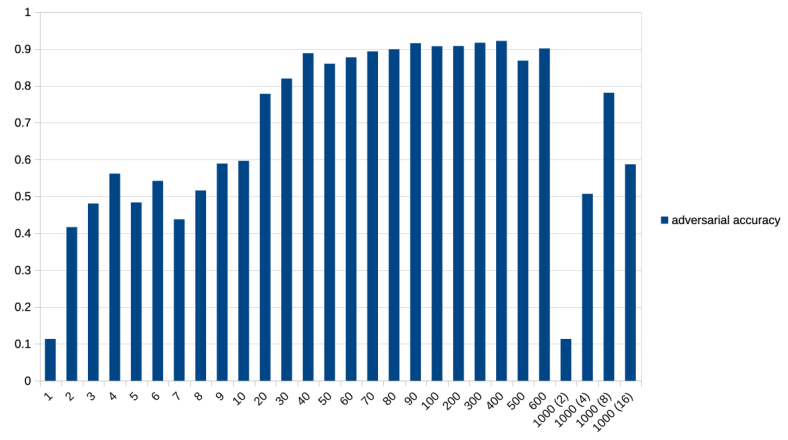

Figure 3: Adversarial accuracy for SGD with $\lambda = 0.1$ and $B = 16$, varying the number of parameters in a fully connected layer of the model. For layer size 1000, the B is given in parentheses.

out. Threats to external validity still remain, as we consider only a single image classification task and $\|\cdot\|_\infty$ perturbations via the FGSM method. However, this task is often used in benchmarking the attack and defense methods. As we can see in the literature, better results on this task often correspond to better results in others.

6 Discussion and Conclusion

We study the adversarial robustness problem, we investigate whether a specifically tuned learning regimes could improve the adversarial robustness. Theoretically, our results are based on the findings that so called "wide" local minima with respect to model parameters are also provide for wide with respect to input data. We derive this property in a basic case of a single-layer network. We also relate the width of the minimum with the Hessian norm and Lipschitz constants for the loss function.

Compared to the earlier results on flat minima, we are able to relate training for wide minima with adversarial robustness both theoretically and empirically.

In the experiments we demonstrate that at least for some neural network models just training with high learning rates and small mini-batch sizes long enough results in the state-of-the-art adversarial robustness earlier achieved only by data augmentation, modifying the loss function, distillation and other complex methods.

For the image classification problem at hand, a much more complex PuVAE (Hwang et al., 2019) and BPFC (Addepalli et al., 2020) methods give around 81% accuracy for a similar predictive model in the $\|\cdot\|_\infty$ setting. While the proposed method with the Adam and SGD optimizers achieve 96% and 89% correspondingly for the same task and the same FGSM ($\varepsilon = 0.3$) attack without sacrifing the prediction quality.

As a consequence, the proposed technique could be used as a basic baseline to compare with, that is, adversarial defense methods should demonstrate better performance than just finding a good "wide" minimum in the original loss function.

Future research could be driven towards reducing the number of steps needed to achieve maximum robustness. In the experiments, the SGD algorithm tends to attain the robustness quicker, while the Adam algorithm generally achieves slightly higher adversarial accuracy. Smarter learning schedules may also take into account that accuracy on the adversarial data depend on the width of the minimum, so that the range of parameters where high adversarial robustness is achieved is larger.

Another important finding is that when reporting results on adversarial robustness, the authors should either use reproducible runs, either obtain reliable statistical estimates as the results greatly depend on particular minimum θ and random seed values.

Finally, the ability of the model to generalize and its adversarial robustness seem to be related, so that training predictive models after reaching full accuracy prior to near-zero loss on the training data improves both.

Reproducibility

Authors would provide the source code and the experiment results on a reasonable request. The source data and other dependencies are available under open-source license already.

Acknowledgments

This work has been carried out using the computing resources of the federal collective usage center Complex for Simulation and Data Processing for Mega-science Facilities at NRC "Kurchatov Institute", `http://ckp.nrcki.ru/`.

References

Sravanti Addepalli, Vivek BS, Arya Baburaj, Gaurang Sriramanan, and R Venkatesh Babu. Towards achieving adversarial robustness by enforcing feature consistency across bit planes. In *Proceed-*

ings of the IEEE/CVF conference on computer vision and pattern recognition, pp. 1020–1029, 2020.

Naveed Akhtar, Ajmal Mian, Navid Kardan, and Mubarak Shah. Advances in adversarial attacks and defenses in computer vision: A survey. *IEEE Access*, 9:155161–155196, 2021.

Ahmed Aldahdooh, Wassim Hamidouche, Sid Ahmed Fezza, and Olivier Déforges. Adversarial example detection for dnn models: A review and experimental comparison. *Artificial Intelligence Review*, 55(6):4403–4462, 2022.

Maksym Andriushchenko, Francesco Croce, Maximilian Müller, Matthias Hein, and Nicolas Flammarion. A modern look at the relationship between sharpness and generalization. In *Proceedings of the 40th International Conference on Machine Learning*, pp. 840–902, 2023.

Tao Bai, Jinqi Luo, Jun Zhao, Bihan Wen, and Qian Wang. Recent advances in adversarial training for adversarial robustness. *arXiv preprint arXiv:2102.01356*, 2021.

Yoshua Bengio et al. Learning deep architectures for ai. *Foundations and trends® in Machine Learning*, 2(1):1–127, 2009.

Battista Biggio, Igino Corona, Davide Maiorca, Blaine Nelson, Nedim Šrndić, Pavel Laskov, Giorgio Giacinto, and Fabio Roli. Evasion attacks against machine learning at test time. In *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2013, Prague, Czech Republic, September 23-27, 2013, Proceedings, Part III 13*, pp. 387–402. Springer, 2013.

Panagiotis Bountakas, Apostolis Zarras, Alexios Lekidis, and Christos Xenakis. Defense strategies for adversarial machine learning: A survey. *Computer Science Review*, 49:100573, 2023.

Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *2017 ieee symposium on security and privacy (sp)*, pp. 39–57. Ieee, 2017.

Gilad Cohen, Guillermo Sapiro, and Raja Giryes. Detecting adversarial samples using influence functions and nearest neighbors. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 14453–14462, 2020.

Alexander D'Amour, Katherine Heller, Dan Moldovan, Ben Adlam, Babak Alipanahi, Alex Beutel, Christina Chen, Jonathan Deaton, Jacob Eisenstein, Matthew D Hoffman, et al. Underspecification presents challenges for credibility in modern machine learning. *Journal of Machine Learning Research*, 23(226):1–61, 2022.

Marina Danilova, Pavel Dvurechensky, Alexander Gasnikov, Eduard Gorbunov, Sergey Guminov, Dmitry Kamzolov, and Innokentiy Shibaev. Recent theoretical advances in non-convex optimization. In *High-Dimensional Optimization and Probability: With a View Towards Data Science*, pp. 79–163. Springer, 2022.

Zhijie Deng, Xiao Yang, Shizhen Xu, Hang Su, and Jun Zhu. Libre: A practical bayesian approach to adversarial detection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 972–982, 2021.

Yinpeng Dong, Fangzhou Liao, Tianyu Pang, Hang Su, Jun Zhu, Xiaolin Hu, and Jianguo Li. Boosting adversarial attacks with momentum. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 9185–9193, 2018.

Pierre Foret, Ariel Kleiner, Hossein Mobahi, and Behnam Neyshabur. Sharpness-aware minimization for efficiently improving generalization. *arXiv preprint arXiv:2010.01412*, 2020.

Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

Dou Goodman, Xingjian Li, Ji Liu, Dejing Dou, and Tao Wei. Improving adversarial robustness via attention and adversarial logit pairing. *arXiv preprint arXiv:1908.11435*, 2019.

Elad Hazan, Kfir Levy, and Shai Shalev-Shwartz. Beyond convexity: Stochastic quasi-convex optimization. *Advances in neural information processing systems*, 28, 2015.

Warren He, James Wei, Xinyun Chen, Nicholas Carlini, and Dawn Song. Adversarial example defense: Ensembles of weak defenses are not strong. In *11th USENIX workshop on offensive technologies (WOOT 17)*, 2017.

Yupeng Hu, Wenxin Kuang, Zheng Qin, Kenli Li, Jiliang Zhang, Yansong Gao, Wenjia Li, and Keqin Li. Artificial intelligence security: Threats and countermeasures. *ACM Computing Surveys (CSUR)*, 55(1):1–36, 2021.

Uiwon Hwang, Jaewoo Park, Hyemi Jang, Sungroh Yoon, and Nam Ik Cho. Puvae: A variational autoencoder to purify adversarial examples. *IEEE Access*, 7:126582–126593, 2019.

Takashi Ishida, Ikko Yamane, Tomoya Sakai, Gang Niu, and Masashi Sugiyama. Do we need zero training loss after achieving zero training error? In *International Conference on Machine Learning*, pp. 4604–4614. PMLR, 2020.

Pavel Izmailov, Dmitrii Podoprikhin, Timur Garipov, Dmitry Vetrov, and Andrew Gordon Wilson. Averaging weights leads to wider optima and better generalization. *arXiv preprint arXiv:1803.05407*, 2018.

Harini Kannan, Alexey Kurakin, and Ian Goodfellow. Adversarial logit pairing. *arXiv preprint arXiv:1803.06373*, 2018.

Qifa Ke and Takeo Kanade. Quasiconvex optimization for robust geometric reconstruction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(10):1834–1847, 2007.

Samer Y Khamaiseh, Izzat Alsmadi, and Abdullah Al-Alaj. Deceiving machine learning-based saturation attack detection systems in sdn. In *2020 IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN)*, pp. 44–50. IEEE, 2020.

Samer Y Khamaiseh, Derek Bagagem, Abdullah Al-Alaj, Mathew Mancino, and Hakam W Alomari. Adversarial deep learning: A survey on adversarial attacks and defense mechanisms on image classification. *IEEE Access*, 10:102266–102291, 2022.

Igor V Konnov. On convergence properties of a subgradient method. *Optimization Methods and Software*, 18(1):53–62, 2003.

Hongshuo Liang, Erlu He, Yangyang Zhao, Zhe Jia, and Hao Li. Adversarial attack and defense: A survey. *Electronics*, 11(8):1283, 2022.

Qin Liu, Rui Zheng, Bao Rong, Jingyi Liu, ZhiHua Liu, Zhanzhan Cheng, Liang Qiao, Tao Gui, Qi Zhang, and Xuanjing Huang. Flooding-X: Improving BERT's resistance to adversarial attacks via loss-restricted fine-tuning. In Smaranda Muresan, Preslav Nakov, and Aline Villavicencio (eds.), *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 5634–5644, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.acl-long.386. URL https://aclanthology.org/2022.acl-long.386.

Avery Ma, Fartash Faghri, Nicolas Papernot, and Amir-massoud Farahmand. Soar: Second-order adversarial regularization. *arXiv preprint arXiv:2004.01832*, 2020.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=rJzIBfZAb.

Dongyu Meng and Hao Chen. Magnet: a two-pronged defense against adversarial examples. In *Proceedings of the 2017 ACM SIGSAC conference on computer and communications security*, pp. 135–147, 2017.

Panayotis Mertikopoulos, Nadav Hallak, Ali Kavis, and Volkan Cevher. On the almost sure convergence of stochastic gradient descent in non-convex problems. *Advances in Neural Information Processing Systems*, 33:1117–1128, 2020.

Maria-Irina Nicolae, Mathieu Sinn, Minh Ngoc Tran, Beat Buesser, Ambrish Rawat, Martin Wistuba, Valentina Zantedeschi, Nathalie Baracaldo, Bryant Chen, Heiko Ludwig, et al. Adversarial robustness toolbox v1. 0.0. *arXiv preprint arXiv:1807.01069*, 2018.

Nicolas Papernot, Patrick McDaniel, Xi Wu, Somesh Jha, and Ananthram Swami. Distillation as a defense to adversarial perturbations against deep neural networks. In *2016 IEEE symposium on security and privacy (SP)*, pp. 582–597. IEEE, 2016.

Nicolas Papernot, Patrick McDaniel, Ian Goodfellow, Somesh Jha, Z Berkay Celik, and Ananthram Swami. 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.

Pouya Samangouei, Maya Kabkab, and Rama Chellappa. Defense-gan: Protecting classifiers against adversarial attacks using generative models. *arXiv preprint arXiv:1805.06605*, 2018.

Aman Sinha, Hongseok Namkoong, and John Duchi. Certifiable distributional robustness with principled adversarial training. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=Hk6kPgZA-.

David Stutz, Matthias Hein, and Bernt Schiele. Disentangling adversarial robustness and generalization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6976–6987, 2019.

David Stutz, Matthias Hein, and Bernt Schiele. Relating adversarially robust generalization to flat minima. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 7807–7817, 2021.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

Ole Tange. Gnu parallel 20240722 ('assange'), July 2023. URL https://doi.org/10.5281/zenodo.12789352. GNU Parallel is a general parallelizer to run multiple serial command line programs in parallel without changing them.

Lei Wu, Chao Ma, et al. How sgd selects the global minima in over-parameterized learning: A dynamical stability perspective. *Advances in Neural Information Processing Systems*, 31, 2018.

Lei Wu, Mingze Wang, and Weijie Su. The alignment property of sgd noise and how it helps select flat minima: A stability analysis. *Advances in Neural Information Processing Systems*, 35:4680–4693, 2022.

Yueyao Yu, Pengfei Yu, and Wenye Li. Auxblocks: defense adversarial examples via auxiliary blocks. In *2019 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–8. IEEE, 2019.

Chongzhi Zhang, Aishan Liu, Xianglong Liu, Yitao Xu, Hang Yu, Yuqing Ma, and Tianlin Li. Interpreting and improving adversarial robustness of deep neural networks with neuron sensitivity. *IEEE Transactions on Image Processing*, 30:1291–1304, 2020.

Jiliang Zhang and Chen Li. Adversarial examples: Opportunities and challenges. *IEEE transactions on neural networks and learning systems*, 31(7):2578–2593, 2019.

Guanghui Zhu, Mengyu Chen, Chunfeng Yuan, and Yihua Huang. Simple and efficient partial graph adversarial attack: A new perspective. *IEEE Transactions on Knowledge and Data Engineering*, 2024.

A ADVERSARIAL EXAMPLES FOR ROBUST MODELS

In this section we provide more details for the experiments. At Fig. 4 we show the adversarial accuracy curves for the FGSM method for $\epsilon \in [0, 1]$ for SGD and Adam optimizers. The increased robustness of the neural network trained with tuned parameters is evident (orange) compared to early-stopping (blue).

We compare adversarial examples generated by FGSM at $\epsilon = 0.3$ for the models trained with SGD and Adam optimizers. As the Adam optimizer finds sharp minima with early-stopping, the corresponding adversarial example is "off-manifold" (Stutz et al., 2019). Whereas examples for the models trained with SGD optimizer resemble the original item more closely.

From these examples we may conclude that adversarial examples produced by FGSM for predictive models with high adversarial robustness are more likely resemble the original items thus, possibly, reducing the label noise and relating the adversarial robustness with the generalization.

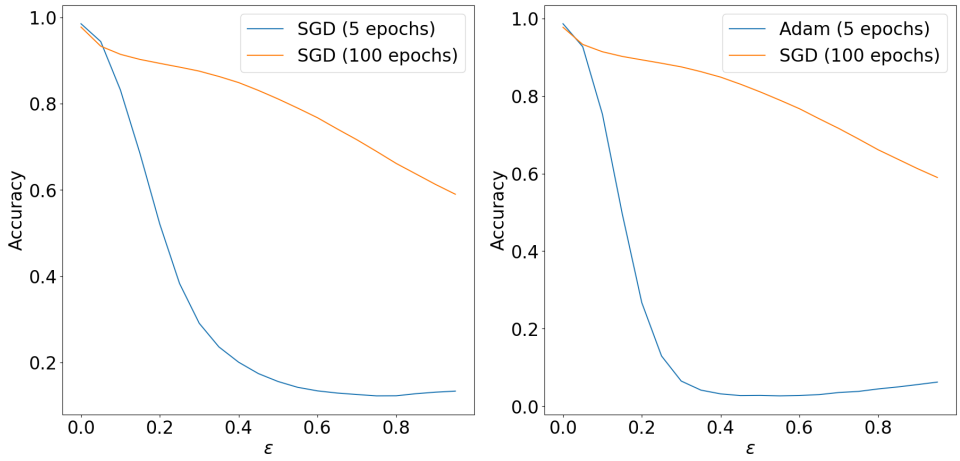

Figure 4: Adversarial accuracy as a function of ϵ (FGSM). Left: SGD optimizer ($\gamma = 0.1$, $B = 16$) after 5 epochs (blue) and the same optimizer after 100 epochs (orange). Right: Adam optimizer ($\gamma = 0.001$, $B = 16$) after 5 epochs (blue), and SGD optimizer ($\gamma = 0.1$, $B = 16$) after 100 epochs (orange).

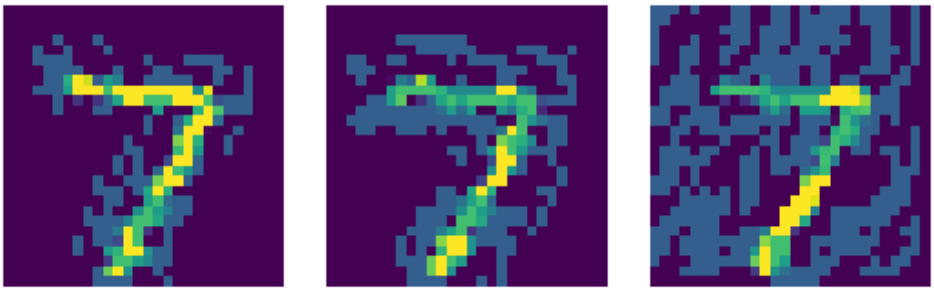

Figure 5: Adversarial examples for $\epsilon = 0.3$ (FGSM). Left: SGD optimizer ($\gamma = 0.1$, $B = 16$) after 100 epochs. Middle: SGD optimizer ($\gamma = 0.1$, $B = 16$) after 5 epochs. Right: Adam optimizer $\gamma = 0.001$, $B = 16$) after 5 epochs.